Aaron K. Baughman
Jiang Gao
Jia-Yu Pan
Valery A. Petrushin *Editors*

Multimedia Data Mining and Analytics

Disruptive Innovation

Springer

Multimedia Data Mining and Analytics

Aaron K. Baughman · Jiang Gao
Jia-Yu Pan · Valery A. Petrushin
Editors

Multimedia Data Mining and Analytics

Disruptive Innovation

Springer

Editors
Aaron K. Baughman
IBM Corp.
Durham, NC
USA

Jia-Yu Pan
Google Inc.
Mountain View, CA
USA

Jiang Gao
Nokia Inc.
Sunnyvale, CA
USA

Valery A. Petrushin
4i, Inc.
Carlsbad, CA
USA

ISBN 978-3-319-14997-4 ISBN 978-3-319-14998-1 (eBook)
DOI 10.1007/978-3-319-14998-1

Library of Congress Control Number: 2014959196

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

In recent years, disruptive developments in computing technology, such as large-scale and mobile computing, has accelerated the growth in volume, velocity, and variety of multimedia data while enabling tantalizing analytical processing potential. During the last decade, multimedia data mining research extended its scope to cover more data modalities and shifted its focus from analysis of data of one modality to multi-modal data, from content-base search to concept-base search, and from corporate data to social networked communities data. Ubiquity of advanced computing devices such as smart phones, tablets, e-book readers, networked gaming platforms, which serve both as data producers and ideal personalized delivery tools, brought a wealth of new data types including geographical aware data, and personal behavioral, preference and sentiment data. Developments in networked sensor technology allow enriched behavioral personal data that include physiological and environmental data that can be implemented to build deep, intrinsic, and robust models.

This book reflects on the major focus shifts in multimedia data mining research and applications toward networked social communities, mobile devices, and sensors. Vast amount of multimedia are produced, shared, and accessed everyday in various social platforms. These multimedia objects (images, videos, texts, tags, sensor readings, etc.) represent rich, multifaceted recordings of human behavior in the networked society, which lead to a range of important social applications, such as consumer behavior forecasting for business to optimize advertising and product recommendations, local knowledge discovery to enrich customer experience (e.g., for tourism or shopping), detection of emergent news events and trends, etc. In addition to techniques for mining single media items, all these applications require new methods for discovering robust features and stable relationships among the content of different media modalities and users, in a dynamic, social context rich, and likely noisy environment.

Mobile devices with multimedia sensors, such as cameras and geographic location sensors (GPS), have further integrated multimedia into people's daily lives. New features, algorithms, and applications for mining multimedia data collected with mobile devices enable the accessibility and usefulness of multimodal data in peoples' daily lives. Examples of such applications include personal assistants, augmented reality systems, social recommendations, entertainment, etc.

In addition to the research topic mentioned above, this book also includes chapters devoted to privacy issues in multimedia social environments, large-scale biometric data processing, content and concept-based multimedia search, advanced algorithms for multimedia data representation, processing, and visualization.

This book is mostly based on extended and updated papers presented at the Multimedia Data Mining Workshops held in conjunction with Association of Computing Machinery (ACM) Special Interest Group Knowledge Discovery and Data Mining (SIGKDD) Conferences in 2010–2013. The book also includes several invited chapters. The editors recognize that this book cannot cover the entire spectrum of research and applications in multimedia data mining but provides several snapshots of some interesting and evolving trends in this field.

The editors are grateful to the chapter authors whose efforts made this book possible and organizers of the ACM SIGKDD Conferences for their supports. We also thank Dr. Farhan Balush for sharing his LaTex expertise that helped to unify the chapters.

We thank the Springer-Verlag employees Wayne Wheeler, who supported the book project, and Simon Rees, who helped with coordinating the publication and editorial assistance.

Durham, NC, September 2014 Aaron K. Baughman
Sunnyvale, CA Jiang Gao
Mountain View, CA Jia-Yu Pan
Carlsbad, CA Valery A. Petrushin

Contents

Part I Introduction

1 Disruptive Innovation: Large Scale Multimedia Data Mining 3
Aaron K. Baughman, Jia-Yu Pan, Jiang Gao
and Valery A. Petrushin

Part II Mobile and Social Multimedia Data Exploration

2 Sentiment Analysis Using Social Multimedia 31
Jianbo Yuan, Quanzeng You and Jiebo Luo

3 Twitter as a Personalizable Information Service 61
Mario Cataldi, Luigi Di Caro and Claudio Schifanella

4 Mining Popular Routes from Social Media 93
Ling-Yin Wei, Yu Zheng and Wen-Chih Peng

5 Social Interactions over Location-Aware Multimedia Systems 117
Yi Yu, Roger Zimmermann and Suhua Tang

6 In-house Multimedia Data Mining 147
Christel Amato, Marc Yvon and Wilfredo Ferré

7 Content-Based Privacy for Consumer-Produced Multimedia 157
Gerald Friedland, Adam Janin, Howard Lei, Jaeyoung Choi
and Robin Sommer

Part III Biometric Multimedia Data Processing

8 Large-Scale Biometric Multimedia Processing 177
Stefan van der Stockt, Aaron K. Baughman and Michael Perlitz

9 Detection of Demographics and Identity in Spontaneous Speech and Writing 205
Aaron Lawson, Luciana Ferrer, Wen Wang and John Murray

Part IV Multimedia Data Modeling, Search and Evaluation

10 Evaluating Web Image Context Extraction 229
Sadet Alcic and Stefan Conrad

11 Content Based Image Search for Clothing Recommendations in E-Commerce 253
Haoran Wang, Zhengzhong Zhou, Changcheng Xiao and Liqing Zhang

12 Video Retrieval Based on Uncertain Concept Detection Using Dempster–Shafer Theory 269
Kimiaki Shirahama, Kenji Kumabuchi, Marcin Grzegorzek and Kuniaki Uehara

13 Multimodal Fusion: Combining Visual and Textual Cues for Concept Detection in Video 295
Damianos Galanopoulos, Milan Dojchinovski, Krishna Chandramouli, Tomáš Kliegr and Vasileios Mezaris

14 Mining Videos for Features that Drive Attention 311
Farhan Baluch and Laurent Itti

15 Exposing Image Tampering with the Same Quantization Matrix 327
Qingzhong Liu, Andrew H. Sung, Zhongxue Chen and Lei Chen

Part V Algorithms for Multimedia Data Presentation, Processing and Visualization

16 Fast Binary Embedding for High-Dimensional Data 347
Felix X. Yu, Yunchao Gong and Sanjiv Kumar

17 Fast Approximate *K*-Means via Cluster Closures 373
Jingdong Wang, Jing Wang, Qifa Ke, Gang Zeng and Shipeng Li

18 Fast Neighborhood Graph Search Using Cartesian Concatenation 397
Jingdong Wang, Jing Wang, Gang Zeng, Rui Gan, Shipeng Li and Baining Guo

19 Listen to the Sound of Data 419
Mark Last and Anna Usyskin (Gorelik)

Author Index 447

Subject Index 449

Contributors

Sadet Alcic Department of Databases and Information Systems, Institute for Computer Science, Heinrich-Heine-University of Duesseldorf, Duesseldorf, Germany

Christel Amato IBM France Laboratory, Bois Colombes Cedex, France

Farhan Baluch Research and Development Group, Opera Solutions, San Diego, CA, USA

Aaron K. Baughman IBM Corporation, Research Triangle Park, NC, USA

Mario Cataldi LIASD, Department of Computer Science, Université Paris 8, Paris, France

Krishna Chandramouli Division of Enterprise and Cloud Computing, VIT University, Vellore, India

Lei Chen Department of Computer Science, Sam Houston State University, Huntsville, TX, USA

Zhongxue Chen Department of Epidemiology and Biostatistics, Indiana University Bloomington, Bloomington, IN, USA

Jaeyoung Choi International Computer Science Institute, Berkeley, CA, USA

Stefan Conrad Department of Databases and Information Systems, Institute for Computer Science, Heinrich-Heine-University of Duesseldorf, Duesseldorf, Germany

Luigi Di Caro Department of Computer Science, University of Turin, Turin, Italy

Milan Dojchinovski Web Engineering Group, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic; Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, Prague, Czech Republic

Wilfredo Ferré IBM Integrated Health Services, Bois Colombes Cedex, France

Luciana Ferrer Speech Technology and Research Laboratory (STAR), SRI International, Menlo Park, CA, USA

Gerald Friedland International Computer Science Institute, Berkeley, CA, USA

Damianos Galanopoulos Centre for Research and Technology Hellas, Information Technologies Institute, Thermi-Thessaloniki, Greece

Rui Gan School of Mathematical Sciences, Peking University, Beijing, China

Jiang Gao Technologies, Nokia, Inc, Sunnyvale, CA, USA

Yunchao Gong Facebook AI Research, Menlo Park, CA, USA

Marcin Grzegorzek Pattern Recognition Group, University of Siegen, Siegen, Germany

Baining Guo Microsoft, Beijing, Haidian District, China

Laurent Itti Department of Computer Science, Psychology and Neuroscience Graduate Program, University of Southern California, Los Angeles, CA, USA

Adam Janin International Computer Science Institute, Berkeley, CA, USA

Qifa Ke Microsoft, Sunnyvale, CA, USA

Tomáš Kliegr Division of Enterprise and Cloud Computing, VIT University, Vellore, India

Kenji Kumabuchi Graduate School of System Informatics, Kobe University, Nada Kobe, Japan

Sanjiv Kumar Google Research, New York, NY, USA

Mark Last Department of Information Systems Engineering, Ben-Gurion University of the Negev, Marcus Family Campus, Beersheva, Israel

Aaron Lawson Speech Technology and Research Laboratory (STAR), SRI International, Menlo Park, CA, USA

Howard Lei International Computer Science Institute, Berkeley, CA, USA

Shipeng Li Microsoft, Beijing, Haidian District, China

Qingzhong Liu Department of Computer Science, Sam Houston State University, Huntsville, TX, USA

Jiebo Luo Department of Computer Science, University of Rochester, Rochester, NY, USA

Vasileios Mezaris Centre for Research and Technology Hellas, Information Technologies Institute, Thermi-Thessaloniki, Greece

John Murray Computer Science Laboratory, SRI International, Menlo Park, CA, USA

Jia-Yu Pan Google Inc., Mountain View, CA, USA

Wen-Chih Peng National Chiao Tung University, Hsinchu, Taiwan

Michael Perlitz IBM Corporation, Herndon, VA, USA

Valery A. Petrushin Research and Development, 4i, Inc., Carlsbad, CA, USA

Claudio Schifanella Department of Computer Science, University of Turin, Turin, Italy

Kimiaki Shirahama Pattern Recognition Group, University of Siegen, Siegen, Germany

Robin Sommer International Computer Science Institute, Berkeley, CA, USA

Stefan van der Stockt IBM Corporation, Johannesburg, South Africa

Andrew H. Sung School of Computing, The University of Southern Mississippi, Hattiesburg, MS, USA

Suhua Tang Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu, Tokyo, Japan

Kuniaki Uehara Graduate School of System Informatics, Kobe University, Nada Kobe, Japan

Anna Usyskin (Gorelik) Department of Information Systems Engineering, Ben-Gurion University of the Negev, Marcus Family Campus, Beersheva, Israel

Haoran Wang Brain-Like Computing and Machine Intelligence Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Jing Wang Key Laboratory on Machine Perception, Peking University, Beijing, China

Jingdong Wang Microsoft, Beijing, Haidian District, China

Wen Wang Speech Technology and Research Laboratory (STAR), SRI International, Menlo Park, CA, USA

Ling-Yin Wei Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan

Changcheng Xiao Brain-Like Computing and Machine Intelligence Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Quanzeng You Department of Computer Science, University of Rochester, Rochester, NY, USA

Felix X. Yu Department of Electrical Engineering, Columbia University, New York, NY, USA

Yi Yu School of Computing, National University of Singapore, Singapore, Singapore

Jianbo Yuan Department of Computer Science, University of Rochester, Rochester, NY, USA

Marc Yvon IBM Human Centric Solutions Center, Bois Colombes Cedex, France

Gang Zeng Key Laboratory on Machine Perception, Peking University, Beijing, China

Liqing Zhang Brain-Like Computing and Machine Intelligence Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Yu Zheng Microsoft Research Asia, Beijing, China

Zhengzhong Zhou Brain-Like Computing and Machine Intelligence Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Roger Zimmermann School of Computing, National University of Singapore, Singapore, Singapore

Part I
Introduction

Chapter 1
Disruptive Innovation: Large Scale Multimedia Data Mining

Aaron K. Baughman, Jia-Yu Pan, Jiang Gao and Valery A. Petrushin

Abstract This chapter gives an overview of multimedia data processing history as a sequence of disruptive innovations and identifies the trends of its future development. Multimedia data processing and mining penetrates into all spheres of human life to improve efficiency of businesses and governments, facilitate social interaction, enhance sporting and entertainment events, and moderate further innovations in science, technology and arts. The disruptive innovations in mobile, social, cognitive, cloud and organic based computing will enable the current and future maturation of multimedia data mining. The chapter concludes with an overview of the other chapters included in the book.

1.1 Introduction

Multimodal, hyper-dimensional, and ultimately multimedia data is the digital vehicle that captures and augments the human experience. The human senses of touch, smell, taste, hearing, and vision are stimulated by multimedia. High definition cameras, biometric devices, audio acquisition, odor characterization and etc. capture bands of information through the lens of a human.

A.K. Baughman (✉)
IBM Corporation, 3039 Cornwallis Road,
Research Triangle Park, NC 27709, USA
e-mail: baaron@us.ibm.com

J.-Y. Pan
Google Inc., 1600 Amphitheatre Parkway,
Mountain View, CA 94043, USA
e-mail: jypan@google.com

J. Gao
Technologies, Nokia, Inc, 200 South Mathilda Avenue, Sunnyvale, CA 94086, USA
e-mail: gao.new@gmail.com

V.A. Petrushin
Research and Development, 4i, Inc., 6256 Citracado Circle,
Carlsbad, CA 92009, USA
e-mail: vapetr3@gmail.com

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_1

Disruptive innovation is the catalyst for changes that enable technological integration into everyday life. The computing backbone that supports multimedia data mining is undergoing technological disruption with trillions of interconnected devices that produce large volumes of data consumable by mathematical algorithms and statistical tools within a cloud computing environment. The accelerating data avalanche is gaining unimpeded momentum that is producing an overwhelming volume and density of information. Specifically, a growing and required component of today's corpora of information is multimedia data. The fabric of an instrumented, interconnected, and intelligent human experience is stitched together by multimedia analytics.

Within the knowledge discovery and data mining community and as evidenced by the success of the previous decade of the Multimedia Data Mining (MDM) workshops, there is an increasing interest in new techniques and tools that can detect and discover patterns in multimedia data. Latest research within MDM describes multimedia information as a digital capsule, which is ubiquitous, rich, artful, and empirical. Entertainment venues, businesses, sporting events, social networks, governments, academia, and the imagination produce and consume multimedia information. Multimedia value and markets are not created from sustained innovation but rather disruptive innovation. In addition, mobile, social, cognitive, cloud and organic based computing will enable the current and future maturation of multimedia data mining.

1.2 Multimedia Disruptive Innovation

Technological change and advancement is fueled by both imagination and empirical design around an overall problem statement. The top down approach begins with imagining the future within the constraints of a business problem. Approaches such as the Walt Disney Imagineering "yes and" are helpful in expanding and encouraging creativity. The "yes and" technique builds upon previous ideas, no matter how outrageous an assertion is. The creativity box is expanded by serendipity, or by the discovery of an ingenious idea within the known and unknown, with the imagination force multiplier. As the process continues, the imagined realities are inputs into the innovation stage. Empirical constraints are applied to each idea such that the innovation can be turned into reality. The conversation from imagination to innovation produces a business impact.

Alternatively, a business goal or desire can be established a priori. After the business constraints are defined, the innovation constraints are defined such as human capital, natural resources, geography, etc. With the bottom level boxes of impact and innovation defined, imagination can be unleashed for within-the-box thinking. The imagination stages use outputs from both the innovation and impact stages to filter creative thought.

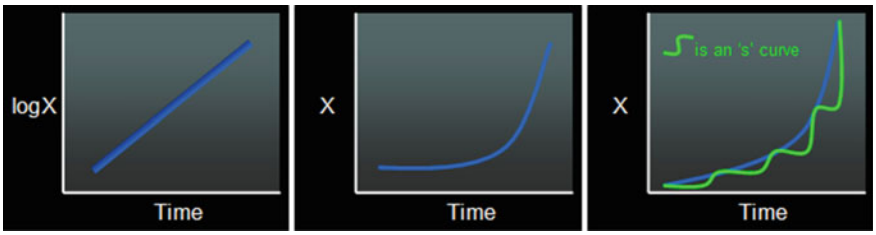


Fig. 1.1 Depiction of disruptive innovations with S-curves

The set of stages Imagine, Innovate, and Impact are represented by the notation i^3. The top down approach imagines the future, innovates to achieve the ideas, and watches the impact of the ideas throughout society and the world. In the other direction, a required impact for society to thrive is defined, innovative techniques are assembled, and ideas are imagined to create a desirable environment. The successful completion of i^3 produces disruptive innovations. Figure 1.1 depicts the phenomenon of disruptive innovation: the S-curve. Each disruptive innovation within a domain is defined by an S-curve, which accelerates the domain's capability along the x-axis. An exponential growth line results when all of the S-curves are curve fitted [1].

The use and combination of multimedia data such as images, sound, movies, vibration, and smell within the world around us, provide the i^3process with a problem statement. How can multimedia data be used to create a safer, sustainable, collaborative, and engaging world? Multimedia Disruptive Innovation is the result from a plurality of possibilities. Within this book, "Multimedia Data Mining and Analytics: Disruptive Innovation", we present some of the leading ideas within the multimedia field. Many of the chapters and sections come from the presentations at the Multimedia Data Mining Workshop held jointly with the Association of Computing Machinery (ACM) SIGKDD Knowledge Discovery and Data Mining Conferences in the previous five years.

1.3 Examples of Multimedia Disruptive Innovations

Disruptive innovation adopts cutting edge technology and ideas that enable new and novel applications to sustain exponential growth. A disruptive innovation increases long term productivity and changes the way people experience and live daily life. In this section, we discuss several disruptive innovations in multimedia, namely, (1) effective human-computer interfaces that increase productivity, (2) new life experiences from world digitization, and (3) ubiquitous multimedia information that facilitates people's life.

1.3.1 Effective Human-Computer Interfaces

Although typing has been the most effective way for a human to interact with a computer, it has never been the most convenient way for a human user. The most natural way for a human to express oneself is through talking and body gesture. Despite many years of research and engineering efforts on speech recognition and gesture understanding, it is not until the past one or two years that commercial products provide effective human-computer interfaces that human can interact with computing devices in a natural fashion. Currently, services such as Siri on iPhone or "Google Now" on Android phones, have been able to understand speech commands from human users with high accuracy.

Large data sets of human speech that are available for training speech recognition models are one of the reasons that effective speech recognition are available today [2]. These large datasets of human speech come in various forms and quality. Some of these data sets are high quality, professionally made radio or podcast programs. There are also mid-quality videos such as lectures from university courses and conferences, as well as, a vast collection of user-posted materials on video-sharing websites such as YouTube. The professionally made data allows researchers to build systems with high recognition accuracy. However, the mid-quality and low-quality videos provides training samples that are less formal and more conversational which can make recognition systems more successful in interacting with ordinary users in daily tasks.

In addition, the availability of large data sets allow the use of novel algorithms to build speech recognition models. In particular, the big data sets allow the use of deep neural networks, which have been shown to outperform previous speech recognition systems [3].

1.3.2 New Life Experiences from the Digitized World

Advances in multimedia recording devices and post-processing algorithms have been gradually digitizing the life experience of humans. The digitization of the physical world and life experience not only facilitates the recording and sharing of life experiences, but also has profoundly changed people's lives in many ways. One example of such change is the Internet services that provide detail maps and even 3D models of the physical world, which have allowed human users to experience the world without being at the physical location.

Internet map services, such as Google Maps and Google StreetView, provide a virtual experience of the physical world [4]. Maps with the 3D model and 360-degree imagery of a location allows a user a good impression of a location, without traveling to the actual place. With such convenience, a user can now check out a travel destination when planning for a vacation. A house buyer can inspect the look of a property and its neighborhood when making purchase decisions. Businesses and governments can also take advantages of this geographical information in planning and forming strategies.

1.3.3 Ubiquitous Multimedia Information Facilitates Individuals' Lives

If having large and informative collections of multimedia information is the foundation on which multimedia disruptive innovations are from, being able to make such information ubiquitously available to everyone at any time is the catalyst of these disruptive innovations. Smart personal devices with Internet access allows a user to access all kinds of multimedia services on the Web, and human life has evolved and transformed.

Currently, multimedia information has become an element of decision making. Before buying a product, a user checks the appearance, price, and customer reviews of the item, as well as information of other competing products. When planning for a vacation, a user inspects the facility and the location of a hotel before making a reservation. When looking for a restaurant on the road, a user can locate nearby restaurants, review comments from friends or previous customers, and checks the menus. In education, multimedia materials (video lectures, slides, interactive homework, and so on) made available by the Massive Open Online Course (MOOC) initiatives have given many more students, no matter where they are, access to high-quality education and can have large impacts on society.

1.4 Multimedia Data Mining S-Curves

Over the last several decades, a few key disruptive innovations had significant impact to the multimedia data mining field. The first contributor to multimedia data mining was the evolution of the Internet. In the 1960s, the Defense Advanced Research Projects Agency (DARPA) awarded several contracts to construct packet network systems to send data between computational devices across disperse geographical locations. The network was called Advanced Research Projects Agency Network (ARPANET), which implemented Transmission Control Protocol (TCP)/Internet Protocol (IP). Charley Kline sent the first message from UCLA to a computer at the Stanford Research Institute (SRI). After several decades of network technology maturation, Tim Berners invented the Hypertext Transfer Protocol (HTTP) and coined the World Wide Web (WWW) [5]. HTTP provides the foundation for data communication over the WWW. The Internet when combined with the WWW and HTTP enabled the possibility to quickly retrieve large collections of documents containing text, images, videos, and audio. The sharing and access to multimedia information over the WWW, which is still considered a research area today, propelled the entire field of multimedia retrieval.

A second boost within the field of multimedia was the development of digital cameras. The introduction of digital cameras with video capability has exponentially multiplied the amount of multimedia content year over year. In the 1990s, digital cameras became affordable and functional for everyday consumers. In fact, one of the pioneers of photography, Eastman Kodak, filed for bankruptcy in January of 2012 in part because the company did not embrace digital camera technology. The portable digital camera helped pave the way to today's 60 % contribution of multimedia to all content [6].

As consumers purchased digital cameras, social media sites began offering services to share photographs. For example, Flickr was created in 2004 to host videos and images. The service has been an open and accessible goldmine for multimedia data collection. The site enabled users to tag photographs while also extracting geolocations, when available, from headers of data in exif format. Shortly thereafter in 2005, YouTube focused efforts on allowing users to freely share and comment about videos. Flickr and YouTube provided the foundation to provide open datasets to evaluate techniques and algorithms within the multimedia field.

The next S-Curve occurred with the mass adoption of smart phones. In 2007, Apple Inc., introduced the iPhone while in 2008 an Andriod operating system phone, HTC Dream (T-Mobile G1), was released as a consumer product [7, 8]. By 2011, Facebook became the largest photograph host in part due to the integration of cameras into mobile phones. The photo aggregator service, Pixable, had over 100 billion photos from Facebook by the middle of 2011 [9]. Users could easily take a picture, video, or sound clip with their smart phone and upload to a social media site. A few staggering stats are that three billion Facebook photo uploads are made per month and 72 hours of video are uploaded to YouTube every minute [6]. Perhaps more importantly than the increase in multimedia volume was the addition of metadata for an image that was acquired by a smart phone meter such as instant geolocation and accelerometer readings. The adoption of the smart phone into every aspect of life has paved the way to an endless number of apps developed to interpret multimedia data.

Currently, the field is experiencing another technological disruption, depth cameras or contextual systems. An example of a depth camera or system is the Kinect. The Kinect enables users to interact with a digital medium by gestures, facial expressions, sound or movements. The technology has opened a line of research that integrates multimedia and augments virtual spaces. Quite possibly in the future, the next S-Curve is forming with wearable computing technology. Google Glass has made wearable computing a reality by going on sale to the general public in May 15, 2014. Wearable computing is evolving a new line of research called egocentric video analysis and summarization [10, 11].

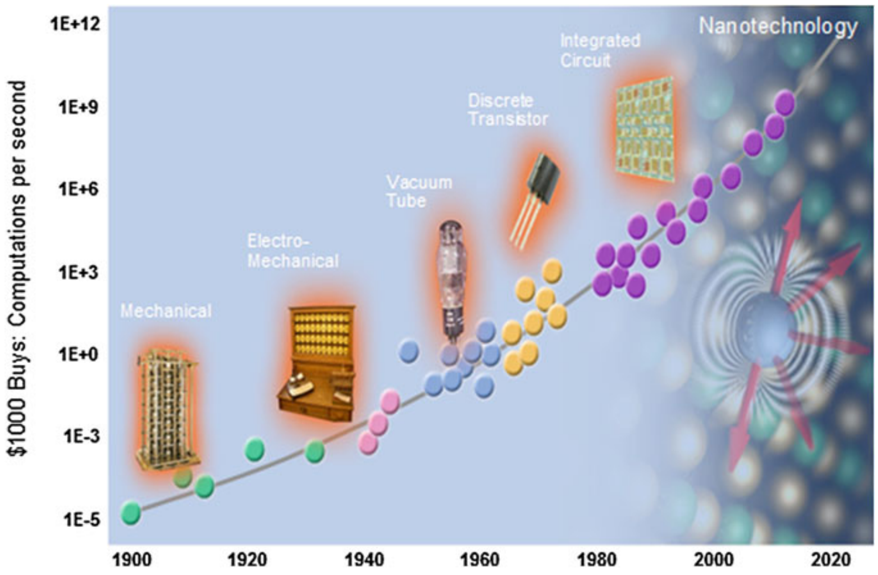


Fig. 1.2 Depiction of Moore's Law with respect to processing power

1.5 Moore's Law

Moore's law is a famous and at times infamous curve that shows that computing power will double every 18 months [12]. Figure 1.2 shows a log-scale curve of the computations per second that $1,000 could buy over a 120 year period. As predicted by Moore's Law, the curve is linear in log scale. In addition, the historical exponential growth of computations will continue into the future with the development of organic computing as described below.

In the 1900s, mechanical devices were used to compute. For example, spaghetti sort encoded numbers onto uncooked strands of pasta. The lengths of the pasta were then sorted by a machine and decoded such that the original data was sorted [13]. The mechanical computing paradigm produced 1E-5 computations per second for a $1,000. In the 1930s and 40s, previous breakthroughs in physics that paired electrical and magnetism together produced the electromechanical computing paradigm. Electrical charge could move a switch, which resulted in binary gates. By 1940, $1,000 would buy 1E-3 computations per second. The next shift occurred in the 1960s with the vacuum tubes. Electrical current could be amplified within a vacuum to active switches in a computer. Mass producing the machines was a challenge. In the vacuum tube paradigm, $1,000 produced one computation per second.

By the 1980s, the discrete transistor was developed. Transistors could be individually packaged and required detailed soldering. Much like the vacuum tube, mass producing the transistors was extremely difficult. The number of computations per second for $1,000 reached 1,000 (or 1E3). The next phase produced the integrated circuits, which were perfected by companies such as Intel and AMD. Semiconductor material is used to create silicon wafers. The technological advancement earned Jack S. Kilby a Nobel Prize award in physics. The integrated circuit disruption enabled the number of computations per second to approach 1E9 for $1,000. In the future, nanotechnology and organic computing can sustain the technological progress that is required for multimedia data.

The exponential growth in computing capability depicted in Figs. 1.2, 1.4 and 1.5 is critical for multimedia applications. Sites such as *Facebook, Twitter, Flickr, LinkedIn, Pinterest, Google Plus+, Tumblr, Instagram, VK* and *Meetup* allow users to post multimedia time capsules. The continuing progress of computing permits the processing and storage of complex data on large distributed cloud centers. The growth of mobile computing increases the velocity of multimedia data acquisition and upload to social media sites. To keep pace with multimedia proliferation, the exponential growth of computing technology is an enabler for multimedia data mining. Complex data representations that cause a curse of dimensionality within algorithms are reduced. In addition, large-scale multimedia data mining is possible with large cloud computing plexes.

1.6 Data Law

As the inverse of Moore's Law, the multimedia data law or data law in general asserts that the cost of acquiring data is exponentially decaying with the progress of technology. Figure 1.3 depicts the curve of the multimedia data law. Data is one of the drivers for technological improvement. Before embedded devices and smart phones, the acquisition of data was extremely labor intensive and costly such as the use of punch cards. Other input devices such as keyboards, mice, tablets, and mobile phones are on the continuum of human assisted information acquisition. The frontier of data acquisition is automatic without much if any human intervention.

A new value of multimedia data is created as information that is acquired automatically and seamlessly. Sensory devices such as eye gaze tracker, heart rate variability monitors, physical location tracker, automatic speech and sound recognition, and etc. are current and future technology enablers. Miniaturization of sensors such as cameras and microphones enable computational systems to provide contextual computing. As production of bio-electronic devices will follow the Moore's Law, the cost of data will plummet as sensors automatically acquire information. Accelerating the movement towards zero cost and Open Data.

Open Data is defined as: "A piece of data or content is open if anyone is free to use, reuse, and redistribute it—subject only, at most, to the requirement to attribute and/or share-alike."[1] The climax of open data began in 2004 with the Organization for Economic Cooperation and Development (OECD), which represents most of the developed countries in the world, signed a declaration that all publically funded

[1] http://opendefinition.org/.

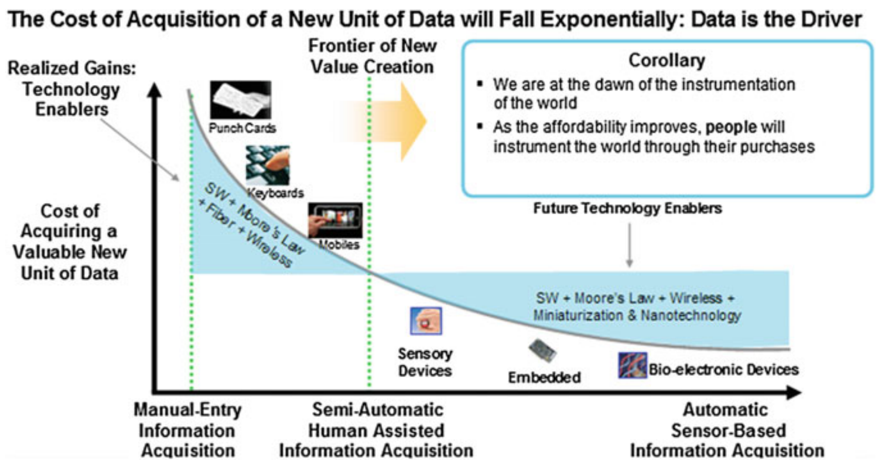

The Cost of Acquisition of a New Unit of Data will Fall Exponentially: Data is the Driver
Frontier of New Value Creation
Realized Gains: Technology Enablers
Corollary
We are at the dawn of the instrumentation of the world
As the affordability improves, people will instrument the world through their purchases
Punch Cards
Keyboards
Mobiles
SW + Moore's Law + Fiber + Wireless
Future Technology Enablers
Cost of Acquiring a Valuable New Unit of Data
SW + Moore's Law + Wireless + Miniaturization & Nanotechnology
Sensory Devices
Embedded
Bio-electronic Devices
Manual-Entry Information Acquisition
Semi-Automatic Human Assisted Information Acquisition
Automatic Sensor-Based Information Acquisition

Fig. 1.3 A depiction of the Data Law, which is the inverse of Moore's Law

archive data should be public [14]. The concept of Open Government was embraced and many academic works and commercial companies began leveraging the free and available government data [15]. The eight principles of open government include[2]:

- Data Must Be Complete.
- Data Must Be Primary and published as collected from the source.
- Data Must Be Timely to preserve the value of the data.
- Data Must Be Accessible so that the widest range of users can access the data.
- Data Must Be Machine Processable to enable algorithm consumption.
- Access Must Be Non-Discriminatory whereby data is accessible by anyone.
- Data Formats Must Be Non-Proprietary where an entity does not have exclusive control.
- Data Must Be License Free so that data restrictions do no exist.

As a well rounded benefit to all, the general public has increased transparency towards their publicly funded government, governments find cost efficiencies by providing free data instead of building service providers, and economic growth occurs as small businesses developed new products as innovative systems of engagement were developed. The United States alone has published over 11,193 datasets from federal agencies and states. The impact of the open data is estimated to have the potential to generate more than $3 trillion a year in diverse sectors such as education, energy, consumer products, health and finance. Clearly, data is a natural resource. In parallel, the scientific community is supporting Open Data called Open Science. A search on the IEEE or ACM libraries with the keywords "Open Data" results in hundreds of innovative papers.

[2] http://www.data.gov/.

Within multimedia data mining, many different data sets are available for experimentation. The MediaEval Datasets support Open Science by providing multimedia open data within speech, audio, visual content, context, users, and tags. For example, MediaEval has provided Spoken Web Search 2013, Violent Scenes Detection 2013, Geographical Placing set, Fashion 10,000, Social Event Detection, Annotated Music, Boredom Detection and etc. The United States National Institutes of Standards and Technology[3] (NIST) provides several types of data sets. Since 2001, NIST has sponsored digital video retrieval (TRECVID) to encourage research in automatic indexing, object recognition, segmentation, and semantic reasoning with large video datasets. In addition, NIST provides fingerprint, mugshot, and facial databases. Other popular people oriented databases include Carnegie Mellon University Pose, Illumination, and Expression (PIE) of humans and Columbia University Public Figures Face Database (PubFig). Several spoken or speech related datasets include University of Pennsylvania's TIMIT Acoustic-Phonetic Continuous Speech Corpus and several data sets from the Massachusetts Institute of Technology including the Negotiation DataSet, Group Polarization DataSet, Speed Dating DataSet, and the Conversational Interest DataSet. Over 121 multimedia datasets that were acquired by diverse devices are listed and referenced on a computer vision open data website [16].

1.7 Moore's Law Meets the Multimedia Data Law

The Moore's Law curve showed in Fig. 1.4 depicts exponential growth on a linear scale and linear growth on a log scale. Throughout history, key technological events produced disruption. The S-curves shown in the Fig. 1.1 depict disruptive innovation that sustained the exponential growth. The regression of the S-curves produces the exponential relationship of computing progress.

For example, the semiconductor sector experienced several S-curves or disruptive innovations. In the 1990s, bipolar silicon capability allowed the persistence of both charge and state [17]. The next S-curve occurred with the Aluminum and Copper CMOS. The technology enabled the integrated circuit on a silicon wafer. In the early 2000s the semiconductor industry was again disrupted by using copper as a conductor over aluminum and copper. By 2005 and leading into 2010, silicon on insulator technology used layers of silicon-insulator-silicon substrates for better computing performance. Leading into 2015, the maturation of embedded Dynamic Random Access Memory (DRAM) enables the placement of memory on the chips themselves. The S-curve pattern continues with the maturation of chip architecture. In the early 1970s, scalar processing was the simplest kind of computing that processed one datum at a time. By the 2000s, superscalar computing brought about parallelism or instruction level parallelism with a single processor. A few years later, multicore processors were introduced that allowed a single computing component with two

[3] http://www.nist.gov/.

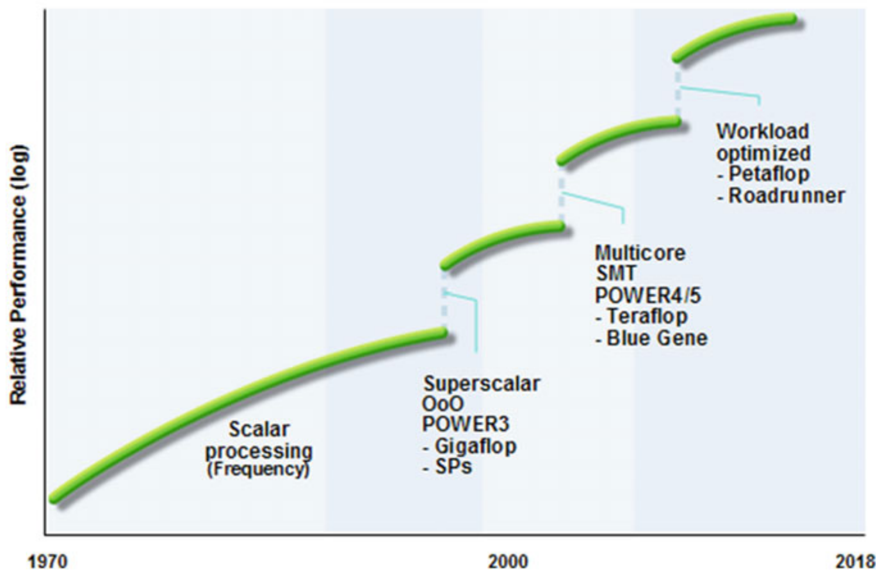


Fig. 1.4 Moore's Law within the context of chip architecture

or more processors to share the same bus. The next jump in computing architecture produces systems that bundle together hardware and software to handle large-scale data processing otherwise known as Big Data.

The Moore's Law and the Data Law are working together to accelerate the possibilities of multimedia data mining. Quite staggering, multimedia data makes up 60 % of Internet traffic, 70 % of available unstructured data and 70 % of mobile phone traffic. In addition, over 100 million photos per day are uploaded to Facebook while over 72 hours of video are uploaded to YouTube every minute [6]. The cost of computing power and of acquiring data for a monetary unit is decreasing. The combination of access to cheap and powerful cloud resources and multimedia data should thrust forward multimedia research.

1.8 Multimedia Technology Drivers

1.8.1 Organic Computing and Nano Systems

As described above, multimedia data will be a large driver for the maturation of computational systems. The miniaturization of data acquisition devices minimizes the cost of data. As such, nanosystems such as systems on a chip, photonics, quantum computing and the DNA transistor are key technological drivers to help simplify complex systems while enabling large-scale multimedia data processing. In 2001, an autonomic computing manifesto was released that asserted the software industry was in a complexity crisis where computing systems were beyond the comprehension of humans. As more devices interconnect to heterogeneous environments, pervasive computing and the Internet of things cascade into a web of entanglement. In [18] it is asserted that the solution to the problem was autonomic computing.

Autonomic computing is a type of computing where systems manage themselves and have a deep relationship to the metaphor of the human immune system and natural systems. Such systems have the following properties [18]:

- Self-configuration: Heuristics define automated configuration of components and systems.
- Self-healing: Software and hardware problems are automatically detected, diagnosed, and repaired.
- Self-optimizing: Hyperparameters or Metaparameters are constantly adjusted to increase performance gains.
- Self-protection: Systems can predict and prevent malicious attacks.

The paper [20] defines the field of Artificial Immune Systems (AIS) and applies the problem solving of the human immune system to biological inspired digital immune systems. The authors examined AIS case studies such as autonomous navigation, computer network security, job-shop scheduling and data analysis. Concurrently and before AIS, groups of researchers began developing natural system algorithms. Previous works developed and designed genetic algorithms and evolutionary computing that mimics the process of natural selection [21, 22]. In 1992, Robert Collins studied and published a dissertation on "Studies in Artificial Life" whereby his goal was to produce biological realism [23]. In addition, biological scientists contributed to natural computing systems by studying the behaviors of animals such as ants [24]. Academic programs such as the Department of Integrative Biology at Berkeley study the influences of structure and function on ecology, biology and the evolution of organisms. Computer scientists hoist natural science discoveries into algorithms [25, 26].

Following both function and structure of nature, nanosystems are bioinspired computing paradigms that have organic properties. Many of the architectures do not follow the conventional von Neumann architecture [27]. The systems have any combination of self-* properties [28]:

- Self-organizing: Within a system of systems or Internet of things, can automatically divide and conquer and orchestrate function towards solution.
- Self-configuration: The ability to setup system parameters.
- Self-optimization: Increase the efficiency to solve problems.
- Self-healing: Capability to recover from catastrophic events or malfunctioning parts.
- Self-explaining: Maintain a sense of self awareness such that the system can introspect and describe itself for humans.

- Context-awareness: Understand the operational ecosystem and can describe the context to humans.

Other definitions include amorphous systems whereby a system does not have a definitive shape or form but maintains an adaptive function within a specific ecosystem. The combination of both function and structure is powerful. For example, within DNA computing, computers leverage the properties of biological DNA to assemble answers to problems encoded in DNA strands. Solutions to problems such as the Traveling Salesman can be instantaneously computed with hydrogen bonding by the Watson-Crick property [29]. Within a gram of DNA, 700 terabytes of data can be encoded [30]. Within one human cell, over 6,000,000,000 rungs of DNA are present. If the base pairs were as far apart as the rungs on a real ladder, the ladder would be halfway to the moon. The density of DNA material within a drop of water enables quadrillions of computations to occur instantaneously.

Over the last decade, a lot of work has been published within the field of organic computing. Several works propose organic computing architectures, principles and frameworks [31–33]. Currently, we are within the infancy of applying both the function and structure of organic computing to multimedia data mining. Only the function of organic computing has been applied to video analysis tasks designed around the self-* framework [34]. The popular ACM Genetic and Evolutionary Computation Conference (GECCO) generally attracts evolutionary generated music papers and workshops. In the future, encoding in a qubit, DNA strand, light, molecule, or etc. is the first step towards the empirical validation of function and structure of organic computing. Potentially, we could look at ourselves as organic left and right brain intelligent computing beings that understand diverse multimedia data: sound, light, touch, taste, and smell. By examining computing through the lens of biology and understanding natural biological processes, we will continue the exponential growth found in Fig. 1.2.

1.8.2 Large Scale Computing

As multimedia data grows in density, large cloud computing infrastructures are needed to turn the information into insights. From 2004 to 2020, high performance computing (HPC) has moved from a high of several petaflops to potentially zettaflops. For example, in 2004, IBM's Blue Gene L achieved 0.3PF. followed by Blue Gene P that reached 1PF. In the early 2010s, IBM Roadrunner was the first HPC system to achieve 10PF. Commercially available systems are predicted to reach thousands of peta flops in 2015. Figure 1.5 presents the growth of the theoretical maximum number of floating point operations that the most powerful computing clusters can achieve within a given year.[4]

Large scale computing clusters has evolved into Cloud computing. Cloud computing technology is a term that refers to distributed computing platforms connected

[4] http://www.top500.org/lists/, July 22, 2014.

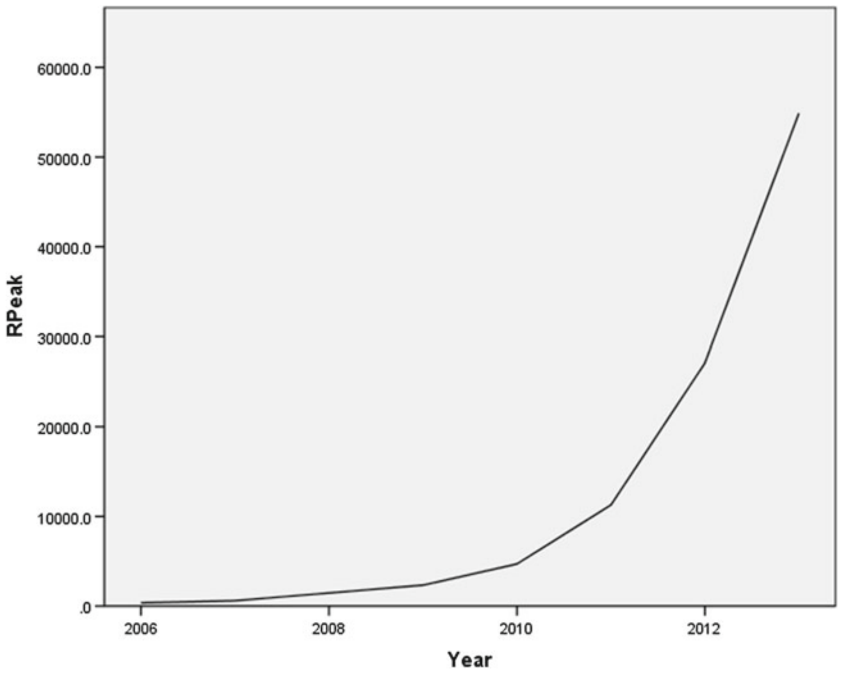


Fig. 1.5 The growth of the maximum number of floating point operations

by any number of networks. Within cloud, simple interfaces abstract users from complex infrastructures that form powerful computing clusters. Cloud computing technology is shifting toward autonomic behavior that is both reactive and predictive, or some combination thereof [19]. In addition, cloud computing is defined as a Software Defined Environment (SDE) and providing Infrastructure as a Service (IaaS). Projects such as OpenStack and OpenNebula are turning infrastructure into code [35].

Within the multimedia community, recipes and cookbooks can be written to create configured machines or logical partitions that support multimedia data processing. Full cluster nodes can be converged on demand or autonomously to optimally leverage the power of the cloud for multimedia data mining. The field of multimedia SDE is within its infancy.

1.8.3 Big and Fast Data Analytics

Multimedia Data is becoming the core component of Big Data. The newly formed Big Data Working Group at the NIST defines Big Data as the inability of traditional data architectures to handle new data [36]. Traditionally, Big Data is broken into 4-V's:

Variety or data within diverse formats, Velocity or the rate of flow of information, Veracity or the quality of data and Volume or the amount of data. Per Sec. 1.6, two additional V's can be added to the definition of Big Data to describe traits of Open Data: Visible or the data should be open to anyone and Value or provides analytical gain. Multimedia data is a form factor of Big Data whereby it is highly diverse, in large volume, processed at varying speeds, has varying degrees of openness and veracity. The culmination of the 6-V's of data are a significant driver of information system architecture.

Several paradigms of computing enable the processing of multimedia Big Data. Large volumes of data that need deep analytics require intense computational resources. Such data can be processed at rest and within a latent environment. These types of system typically split data between hundreds of cores and push algorithms to the data. In many cases, the splitting of data is done by a map function and the merging of individual processed elements is completed by a reduce function. The underlying system automatically parallelizes the processing among all available cores. The MapReduce paradigm handles machine failures, job scheduling, and parallelization [37]. The MapReduce paradigm is implemented in a widely known and recognized open source Hadoop system. Many commercial offerings such as Cloudera, IBM InfoSphere BigInsights, Amazon Elastic Map Reduce, Cloudspace, Pangool, Hortonworks and etc. have been released to ease the introduction of MapReduce into information systems. In addition, machine learning packages such as Apache Mahout leverages Hadoop to parallelize machine learning processes.

Data in motion can be streamed through analytical pipelines for real time processing. High velocity and instantaneous computations can be completed as data is accumulated. Quick analytics can produce more data that is pushed to downstream processing. Systems such as Aurora, STREAM, and Borealis were early streaming databases rooted from the academic community [38–41]. Streams processing has some roots in memory data processing on large cluster computing platforms [42]. Several open source and commercial products are now available to support stream processing such as DataTorrent, IBM InfoSphere Streams, Emblocsoft, HStreaming, and Apache Spark.

A third paradigm, data on demand, streams latent data through analytical pipelines for quick computations. On February 14–16, 2011, the DeepQA project produced a system called Watson that competed and won on the game show Jeopardy!. The machine had to answer questions within 2–3 s. The team developed Apache Unstructured Information Management Architecture (UIMA) to support deep Natural Language Processing (NLP) algorithms on a large amount of data [43]. Pieces of information and the accumulation of evidence were pushed through the analytical pipelines to support real time response. Furthermore, many computing architectures combine both data at rest and in motion data architectures.

1.8.4 Thinking Machines

Multimedia data affords computing with the opportunity to learn from the world in ways humans interact with the environment. Decades of research and work have lead to Cognitive Computing. The new paradigm of computing will enable systems to learn from multimedia data, enhance humans cognitive ability to understand multimedia data, and will naturally interact with humans through their senses of sight, taste, touch, hear, and smell. Figure 1.6 shows the evolution of thinking machines.

The earliest computer was the abacus followed by the stunningly complex Antikythera. Because of the complexity, the Antikythera was not replicated until 1400 AD in Europe. Circa the 1600s, Napier's rods only used addition to support multiplication. A paper was published in Rabdologia that described how to use Napier's rods. Babbage's machines followed Napier's rods. Babbage created two types of machines [17]. The first was a Difference Engine that used only arithmetic addition to solve problems. In 1834, the second Babbage machine was called an Analytical Engine, which was a programmable computing engine.

Next, the Elementron Numerical Integrator And Computer (ENIAC) was developed and announced in circa 1945 as the first general purpose digital computing device that was capable of being reprogrammed [44]. Of note, the ENIAC was Turing-complete. The ENIAC contained over 17,000 vacuum tubes. Unfortunately, when one tube malfunctioned, an entire tube board had to be replaced.

In 1964, International Business Machines (IBM) announced the introduction of the System/360. The System/360 was the first mainframe computer that was designed for general purposes and separated architecture and implementation . The System/360 was very successful with a 2/3rds market attainment [45]. Computing innovation began to accelerate with the introduction of the Integrated Circuit. In 1971, Intel released The Intel 4004 that was a general purpose 4-bit computer on 4 chips with a width of 10,000 nm [46]. The development of the Integrated Chip (IC), sets the stage for scientists and engineers to develop "Thinking Machines".

In 1958, Simon and Newell said that "… within ten years a digital computer will be the world's chess champion…". The two founders of Artificial Intelligence were wrong about the date but correct with their prediction. In 1997, Deep Blue beat Gary Kasparov, a chess grandmaster and former World Chess Champion, in chess 3.5 to 2.5 [47]. Alan Turing also posed the Imitation Test where a human and computer were indistinguishable in games such as chess or question and answer challenges [48]. On February 14–16, 2011, a massively parallel and probabilistic question and answer system named Watson, beat both Brad Rutter, the largest money winner on Jeopary!, and Ken Jennings, the record holder for the longest winning streak [43]. Hayes and Ford were critiques of the Turing and Imitation test. They claimed that Turing's measures for AI were much too restrictive and would produce an "artificial con artist" [49]. However, IBM's Watson is being put to work through commercial domain adaptation that included content, functional, and training adaptation [50]. Perhaps the next disruptive innovation within the evolution of "Thinking Machines" is the use of cognitive chips and biologically inspired systems as described above.

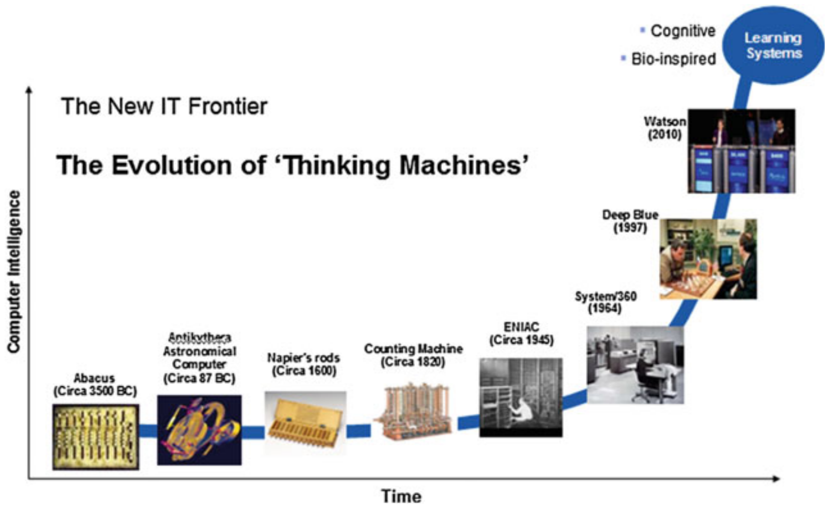


Fig. 1.6 The progression of intelligent computing

The boundaries of multimedia computational intelligence, reasoning under uncertainty, probabilistic reasoning, pattern recognition, machine learning, and etc. can be viewed through the proceedings of popular conferences such as:

- ACM Knowledge Discovery and Data Mining (KDD).
- ACM Multimedia (ACMMM).
- IEEE Computer Vision and Pattern Recognition (CVPR).
- IEEE International Conference on Computer Vision (ICCV).
- International Conference on Machine Learning (ICML).

Scientific journals evaluate papers and scientific works to share discovery to researchers, teachers and practitioners. The following journals provide a leading edge pulse within the fields of algorithms and multimedia:

- IEEE Transactions on Pattern Analysis and Machine Intelligence.
- IEEE Transactions on Evolutionary Computation.
- IEEE Transaction on Multimedia.
- Foundations and Trends in Machine Learning.
- ACM Transactions on Knowledge Discovery from Data.

We are very excited with the emerging trends of the use of multimedia data to enhance the physical world and to create immersive environments. Everyday new mobile applications are developed to interpret images, sounds, location and accelerometer information. Social networks use images, sound and video to interconnect people. New haptic interfaces are being developed to engage users within systems. Cognitive computing is learning and interacting with human usage. Systems of systems are both creating and consuming data. We live within an exciting era of disruption innovation!

1.9 Overview of the Book's Contents

During recent years, the focus of Multimedia Data Mining became wider and shifted to new data sources. Besides the traditional research that focuses on algorithms for improving the content and concept based multimedia search, feature representation and selection, new research directions include processing data from social networks, mobile devices, and sensors that provide multimedia data enriched with subjective, environmental and location-aware information. Such richness of data sources on one hand allows creating more sophisticated applications, but on the other hand increases the risk of privacy threat.

The book consists of five parts. The first part is an introduction, which includes the chapter that you are reading now. It gives a historical overview of how the technological innovations driven by information processing needs create positive feedback loops to expedite technological progress. It also shows the trends of technology development in the future and overviews the content of the chapters included within the book.

The second part is devoted to the rapidly developing field of mobile and social multimedia data processing and exploration. It includes six chapters. Chapter 2 "Sentiment Analysis Using Social Multimedia" by Jianbo Yuan, Quanzeng You and Jiebo Luo deals with sentiment analysis and opinion mining, which is a rapidly developing area of research with numerous applications almost to every possible domain, from consumer products, services, health care, and financial services to social events and political elections. The authors present latest works on topics of sentiment analysis based on both textual data and visual data. They introduce *Sentribute*, a novel image sentiment analysis framework based on middle level attributes and eigenface expression attributes. The chapter also presents a new study aimed at analyzing the sentiment changes of Twitter users by processing both visual and textual multimedia data.

Chapter 3 "Twitter as a Personalizable Information Service" by Mario Cataldi, Luigi Di Caro and Claudio Schifanella explores Twitter as one of the fastest and dynamic information services in the world. The authors present an approach for extracting, in real-time, the emerging topics expressed by the community along the interests of a specific user. The social community is modeled as a directed graph of the active authors based on their social relationships, calculating their authority by relying on the well-known PageRank algorithm. The stream of information in the entire network is monitored by studying the life cycle of each term according to an aging model that also leverages the reputation of each author. The set of most emerging keywords are selected by dynamically ranking the terms depending on their life status, defined through a burstiness value. Finally, each topic is created by constructing and analyzing a keyword graph, which links the extracted emerging terms with all the co-occurring keywords. In order to personalize the list of retrieved emerging topics, the temporal time frames, in which the user has been active, and the generated content are also analyzed. This time-aware information is finally used to highlight the topics that best match the interests of the user.

Chapter 4 "Mining Popular Routes from Social Media" by Ling-Yin Wei, Yu Zheng and Wen-Chih Peng deals with mining spatial trajectories, which are time series augmented with location coordinates, or linked to objects of known locations. The amount of such data is rapidly growing due to advances in location detection technologies using mobile devices and emerging location-based Web services. Many of such trajectories have irregular and low frequency which causes uncertainty about an object's location in time between available data points. To work with such data the authors present a Route Inference framework based on Collective Knowledge (RICK), which derives the popular routes from uncertain trajectories by aggregating them and building a routable graph using collaborative learning. Then the top-k routes going through the locations within the specified time span are constructed. The framework was developed and tested using two real-life datasets—the users' check-in dataset in Manhattan, NY from the local search and recommendation service Foursquare, and the taxi trajectories in Beijing. The results showed that the framework is both effective and efficient.

Chapter 5 "Social Interaction over Location-Aware Multimedia Systems" by Yi Yu, Roger Zimmermann and Suhua Tang gives an overview of research and development in processing location-aware data including techniques for extracting location information from multimedia data, geo-tag data processing from social networks, and applications to numerous location-based services. The authors introduce the basic concepts and techniques of location-aware data processing, and applying them to identifying individual user interests and geographic-social behaviors. In particular, they present the concept of geo-fencing and related techniques, which form a basis for user-centric mobile location-based services. The chapter describes the authors' experience in processing geographic-aware social media and social interaction data from Flickr, Foursquare, and Twitter, including leveraging tweets with geospatial information for mining music listening patterns, mapping geo-categories to moods, and multimedia content diffusion.

Chapter 6 "In-house Multimedia Data Mining" by Christel Amato, Marc Yvon, and Wilfredo Ferré present research conducted by the European IBM Human Centric Solutions Center. It describes technical solutions made for the in-house multimedia project, where the goal was creating a framework for monitoring activities of elderly people at home using several streams of data coming from sensors, such as water leakage detector, light on/off detector, CO and CO_2 level, smoke detector, temperature and humidity values. The solution has a hierarchical structure with Zigbee communication system for collecting and aggregating data locally, standard telecom equipment for transmitting data to the IBM server via 3G networks, and the IBM cloud system for processing data and draw insights. A pilot experiment, which was conducted during eight months in the city of Bolzano in Italy, showed the reliability of the proposed technical solution. The collected data provided the base for developing a range of applications that derive insights about wellness of elderly people.

Chapter 7 "Content-based Privacy for Consumer-Produced Multimedia" by Gerald Friedland, Adam Janin, Howard Lei, Jaeyoung Choi, and Robin Sommer is devoted to a crucial topic of privacy threat, which has become the focus of current interest due to the exponential growth of multimedia materials on the Web and improvements in multimedia content analysis techniques such as face recognition, speaker verification, location estimation, etc. The unethical use of multimedia data collected on the Web scale could make the privacy threat enormous and pervasive. The multimedia community therefore has an obligation to understand these risks, mitigate the effects, and educate the public on the issues. The authors outline existing and future multimedia content analysis and linking techniques that could support unethical use, and describe possible attack vectors. Then they describe some preliminary experiments providing evidence that multimedia analytics can circumvent one aspect of privacy by linking accounts. Finally, mitigation and educational techniques are outlined.

The third part consists of two chapters that present research in biometric data processing. Chapter 8 "Large-scale Biometric Multimedia Processing" by Stefan van der Stockt, Aaron Baughman, and Michael Perlitz describes research and development studies on the analysis of large-scale biometrics datasets. The authors provide an overview of the current state of the art in the field, including search space reduction, feature selection, and parallel processing of biometric data. They present their results in solving the fingerprint identification task using the bootstrapped C-means clustering to reduce the search space and using support vector machine recognizer for identification. The proposed approach shows high precision and reduced processing time. The task of reducing the number of features is very important for improving performance of large-scale biometric systems. The authors propose an innovative algorithm based on evolutionary computing to perform efficient facial feature selection for identification purposes. The authors describe designs of large-scale biometric systems that take the advantages of multi-core, distributed, cloud and mobile computing technologies.

Chapter 9 "Detection of Demographics and Identity in Spontaneous Speech and Writing" by Aaron Lawson, Luciana Ferrer, Wen Wang, and John Murray investigates how identity and demographic categories are manifested in spoken and written language, and highlights approaches to capture this information for real world analysis, including talker and writer identification, and authentication. The authors address language use in the virtual world of on-line games and text entries on mobile devices in the form of chats, emails and nicknames, and demonstrates socio-linguistic features and text factors that correlate with demographics, such as age, gender, personality and interaction style. As for the spoken language analysis, the authors overview the major problems of speaker identification, including differences inherent to the talker and external environment. Recent findings in terms of features (acoustic and prosodic), as well as modeling techniques that have provided breakthroughs in recent evaluations, such as low-dimensional iVector representations of an utterance and probabilistic linear discriminant analysis (PLDA) for score generation, are examined. The authors present an on-going work that combines research from both written and spoken authentication and characterization approaches to provide continuous authentication of users on their mobile devices using spoken and written inputs on the device. This continuous authentication will make use of the shared space of language, which covers speech and writing, and the sociolinguistic relationships that emerge from the intersection of language use and personality, background, gender, age, ethnicity, and interaction style.

The fourth part is devoted to multimedia data modeling, search and evaluation. It includes some traditional topics in multimedia data research, such as content-based image search, video retrieval and concept detection, as well as new topics, such as identifying features that drive attention in video, and detecting illegal changes in images. It includes six chapters. Chapter 10 "Evaluating Web Image Context Extraction" by Sadet Alcic and Stefan Conrad deals with evaluation of image retrieval from the Web. For image retrieval, both visual features extracted from images and textual information extracted from the image context such as image captions or text surrounding the image can be used. The problem of finding the relevant image context is not trivial. Several methods that automatically determine and extract the Web image context from Web documents have been applied in various applications over the years. However, in these applications context extraction is only a preprocessing step and therefore the quality of the extraction task has not been evaluated on its own. The authors propose an evaluation framework that objectively measures and compares the quality of Web Image Context Extraction (WICE) algorithms. The main parts of the framework are a large ground truth dataset consisting of diverse Web documents from real Web servers and objective quality measures tailored to the special characteristics of the image context extraction task. Common extraction methods from the literature are implemented and integrated into the Framework, and the evaluation results are summarized and discussed.

Chapter 11 "Content Based Image Search for Clothing Recommendations in E-Commerce" by Haoran Wang, Zhengzhong Zhou, Changcheng Xiao and Liqing Zhang is devoted to content based image search. The authors present three models for searching similar clothing. The first model is based on sketch-based image search and uses contour features. To expedite search, the features are sorted according their importance and a three-level hierarchical search process is implemented. At each level, more features are used and fewer top rated images are selected. The second model uses the spatial bag-of-feature approach, which takes into account the spatial distribution of features in the image. The third model is a query adaptive shape topic model, which combines shape features with high-level concepts that are represented by natural language words assigned to clothing images. Experimental results based on a dataset of 100,000 clothing images with more than 30 categories of clothing showed that the third model which uses both low-level visual and high-level semantic features outperforms the models based only on visual features.

Chapter 12 "Video Retrieval based on Uncertain Concept Detection using Dempster-Shafer Theory" by Kimiaki Shirahama, Kenji Kumabuchi, Marcin Grzegorzek and Kuniaki Uehara presents research in high-level concept detection and concept-based video retrieval. The authors give an introduction and overview of the current state of the art in the fields of content-based and concept-based retrieval. In opposite to the content-based retrieval, which uses low-level visual features to fetch relevant video shots, the concept-based retrieval approach first uses a number of concept detectors each gives probability of presence of a particular high-level concept in a shot and then merges the results and ranks shots according to their relevance. Merging the results of concept detectors is a very critical step that defines the accuracy of retrieval. The authors proposes a novel approach that uses the Dempster-Shafer Theory for merging the concept detectors' scores, taking into account their uncertainties. Experiments using TRECVID 2009 data and 24 queries show that, for three queries, the approach outperforms the best results and for four other queries the results are in the top five results.

Chapter 13 "Multimodal Fusion: Combining visual and textual cues for concept detection in video" by Damianos Galanopoulos, Milan Dojchinovski, Tomas Kliegr, Krishna Chandramouli, and Vasileios Mezaris describes research in concept-based video retrieval. The authors explore different approaches to improve concept detection accuracy by merging the results from recognizers that are based on visual features with results from text-based recognizers that use automated speech recognition transcripts. The chapter gives an overview of visual-based and text-based concept detection algorithms and justifies using the Explicit Semantic Analysis approach for text-based concept recognition. For merging recognition results from recognizers of different modalities, the authors suggest linear combination of results, meta-classification using additional recognizers that use the original results as inputs, and second level fusions where the results of original recognizers are combined with results of meta-classification. Experiments using 34 concepts and the TRECVID 2012 Semantic indexing task dataset show that using meta-classification with SVM could improve the accuracy of recognition by 13 % and using second level fusion by 36 %.

Chapter 14 "Mining Videos for Features that Drive Attention" by Farhan Baluch and Laurent Itti describes research on identifying visual features that capture people's attention in video. The authors conducted experiments with eight subjects who were watching videos and their attention allocations were measured by an eye tracker. The aggregated localization data served as the ground truth for measuring accuracy of attention localization using models based on different visual features and their weighted combinations. The authors explored 18 features from low-level color, intensity, motion and texture features to more advanced shape features such as T- and X-shaped edge junctions and human-like shapes. After estimating the performance of each feature individually, the authors selected the following top features: motion, color, orientation, intensity and flicker. A simple linear combination of the features results in a model that performs reasonably well. In particular, a genetic algorithm was proposed to estimate the weights for combining these features and was shown to improve the model performance.

Chapter 15 "Exposing Image Tampering with the Same Quantization Matrix" by Qingzhong Liu, Andrew H. Sung, Zhongxue Chen and Lei Chen is devoted to multimedia forensics, which has emerged recently as a new discipline. The authors focus on image forgery detection using a shifted-recompression-based approach to detect the image tampering with the same quantization matrix. Several classifiers are designed and experiments are performed to evaluate the effectiveness of the approach. Results indicate that the approach is indeed highly effective in detecting image tampering and relevant manipulations by using the same quantization matrix.

The fifth part is a collection of four chapters that present algorithms for multimedia data processing and presentation. Chapter 16 "Fast Binary Embedding for High-Dimensional Data" by Felix X. Yu, Yunchao Gong, and Sanjiv Kumar presents novel algorithms for dimensionality reduction using binary embedding of high-dimensional data. Traditional binary coding methods often involve very high computation and storage cost. The authors propose two solutions: the Bilinear Binary Embedding, which converts high- dimensional data to compact similarity-preserving binary codes using compact bilinear projections, and the Circulant Binary Embedding, which generates binary codes by projecting the data with a circulant matrix using Fast Fourier Transformation to speed up the computation. Both methods dramatically reduce the time and space complexity comparing with the best state-of-the-art techniques. The authors present the two approaches in a unified framework, covering randomized binary embedding, learning-based binary embedding, and learning with dimension reductions. To demonstrate the advantages of the proposed methods, experiments were conducted using three real-world high-dimensional datasets used by the current state-of-the-art method for generating binary codes. The proposed methods showed both significant reduction in processing time and an increase in accuracy.

Chapter 17 "Fast Approximate k-Means via Cluster Closures" by Jingdong Wang, Jing Wang, Qifa Ke, Gang Zeng, and Shipeng Li presents a novel approximate k-means clustering algorithm that outperforms in terms of both accuracy and running time the state-of-the-art approximate k-means algorithms such as hierarchical k-means, approximate k-means and Canopy clustering. The approach was motivated by the observation that during iterative reassigning data points to clusters, the points that are changing their cluster assignments frequently are located on or near cluster boundaries. The algorithm efficiently identifies those active points by pre-assembling the data into groups of neighboring points using multiple random spatial partition trees, and uses the neighborhood information to construct a closure for each cluster, in such a way only a small number of cluster candidates need to be considered when assigning a data point to its nearest cluster. The authors provide the complexity analysis of the algorithm and describe its applications on image data clustering and image retrieval.

Chapter 18 "Fast Neighborhood Graph Search using Cartesian Concatenation" by Jingdong Wang, Jing Wang, Gang Zeng, Rui Gan, Shipeng Li, and Baining Guo is devoted to improving efficiency of approximate nearest neighbor search for large scale and high-dimensional multimedia data. The authors describe an approach that greatly augments neighborhood graph search by proposing a new data structure. First, Cartesian concatenation is applied to produce a large set of vectors, called bridge vectors, from several small sets of sub-vectors. Each bridge vector is connected with a few reference vectors near to it, forming a bridge graph. The neighborhood graph is augmented with the bridge graph. The proposed approach finds nearest neighbors by simultaneously traversing the neighborhood graph and the bridge graph using best-first strategy. The success of this approach stems from two factors: the exact nearest neighbor search over a large number of bridge vectors can be done quickly, and the reference vectors connected to a bridge (reference) vector near the query are also likely to be near the query. Experimental results on searching over several large-scale datasets show that the proposed approach outperforms state-of-the-art approximate nearest neighbor search algorithms in terms of efficiency and accuracy.

Chapter 19 "Listen to the Sound of Data" by Mark Last and Anna Usyskin describes an approach to data perception using the auditory channel, which could complement data visualization techniques or substitute them in cases when visual representation is impossible to use. After introducing the sonification techniques and overviewing the current state of the art in the field, the authors present a sonification algorithm for univariate or multivariate (up to ten dimensions) time series. The input time series are converted into a Western tonal music in MIDI format. The approach is tested by conducting two usability studies. During the studies, subjects listened to sonified versions of time series and were asked questions about how the data values are changing over time, or which dataset of two alternatives is similar to third one. The results of both studies showed that subjects were able to successfully perform a variety of common data exploration tasks using the proposed sonification approach.

Acknowledgments Special thanks to David McQueeney and Michele Merler for guidance and content review.

References

1. Kurzweil R (2005) The singularity is near: when humans transcend biology. Penguin Group, New York
2. Baker JM, Deng L, James G, Khudanpur S, Lee C-H, Morgan N, O'Shaughnessy D (2009) Research developments and directions in speech recognition and understanding, part 1. IEEE Signal Process Mag 26(3):75–80
3. Hinton G, Deng L, Mohamed A-R, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath T, Dahl G, Kingsbury B (2012) Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process Mag 29(6):82–97
4. Madriga AC (2012) How Google builds its maps—and what it means for the future of everything. The Atlantic, 6 September 2012
5. Hafner K, Lyon M (1996) Where wizards stay up late: the origins of the internet. Simon and Schuster Paperbacks, New York
6. Global Technology Outlook 2013 (2013) IBM research
7. The iPhone is not a smartphone (2007) Engadget.com, 9 January 2007. Retrieved 24 July 2014
8. T-mobile G1 event round-up (Press release) (2008) Talk media Inc. US, 22 October 2008. Retrieved 24 July 2014
9. Kessler S (2011) Facebook photos by the numbers. Mashable, Retrieved 23 July 2014
10. Yong JL, Ghosh J, Grauman K (2012) Discovering important people and objects for Egocentric video summarization. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Providence, June 2012
11. Zheng L, Grauman K (2013) Story-driven summarization for Egocentric video. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Portland, June 2013
12. Moore GE (1998) Cramming more components onto integrated circuits. Electronics 38(8):114–117
13. Dewdney AK (1998) On the Spaghetti computer and other analog gadgets for problem solving. Sci Am 250(6):19–26

14. OECD declaration on open access to publicly funded data (2014) http://www.oecd.org/science/sci-tech/sciencetechnologyandinnovationforthe21stcenturymeetingoftheoecdcommitteeforscientificandtechnologicalpolicyatministeriallevel29-30january2004-finalcommunique.htm, 21 July 2014
15. Fountain JE (2001) Building the virtual state: information technology and institutional change, Brookings Institution Press, August 2001
16. CV datasets on the web (2014) www.cvpaper.com/datasets.html, 21 July 2014
17. The babbage engine (2014) http://www.computerhistory.org/babbage/engines/, 22 July 2014
18. The next generation: semiconductors (2014) http://www.computerhistory.org/revolution/digital-logic/12/272, 22 July 2014
19. Kephart JO, David M (2003) Chess the vision of autonomic computing. Computer 36(1):41–50
20. de Castro, Leandro R, Jonathan T (2002) Artificial immune systems: a new computational intelligence paradigm. Springer, New York, September 2002
21. Holland JH (1992) Adaptation in natural and artificial systems [Book]. - [s.l.], 5th edn. MIT Press
22. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning, reading. Addison-Wesley, Boston
23. Collins RJ (1992) Studies in artificial evolution. Dissertation, Doctor of Philosophy in Computer Science, University of California, Los Angeles
24. Gordon D (1999) Ants at work: how an insect society is organized. Free Press, New York
25. Sato M, Fukaya M, Iwasaki T (2002) Serpentine locomotion with robotic snakes. IEEE Control Syst 22(1):64–81
26. Yasong L, Ausama A, Sameoto D, Menon C (2012) Abigaille ii: toward the development of a spider-inspired climbing robot. Robitica 30(1):79–89
27. Merolla P, Arthur J, Akopyan F, Imam N, Manohar R, Modha DS (2011) A digital Neurosynaptic core using embedded crossbar memory with 45 pJ per spike in 45 nm. In: IEEE custom integrated circuits conference (CICC), San Jose
28. Christian M-S, Schmeck H, Ungerer T (eds) (2011) Organic computing—a paradigm shift for complex systems. Springer
29. Paun G, Rozenberg G, Salomaa A (1998) DNA computing: new computing paradigms. Springer, Berlin
30. Church GM, Gao Y, Kosuri S (2012) Next-generation digital information storage in DNA. Science 337(6102):1628. doi:10.1126/science.1226355
31. Gudemann M, Nafz F, Ortmeier F, Seebach H, Reif W (2008) A specification and construction paradigm for organic computing systems. In: IEEE self-adaptive and self-organizing systems (SASO)
32. Fey D, Komann M, Shurtz F, Loos A (2007) An organic computing architecture for visual microprocessors based on marching pixels. In: IEEE international symposium on circuits and systems (ICAS)
33. Seebach H, Ortmeier F, Reif W (2007) Design and construction of organic computing systems. In: IEEE congress on evolutionary computation
34. Wurtz RP (2007) Organic computing for video analysis. In: Hybrid Intelligent Systems
35. Wen X, Gu G, Li Q, Gao Y, Zhang X (2012) Comparison of open-source cloud management platforms: OpenStack and OpenNebula. In: 9th international conference on Fuzzy systems and knowledge discovery (FSKD). doi:10.1109/FSKD.2012.6234218
36. DRAFT NIST big data interoperability framework: volume 1, definitions. NIST special publication, Information Technology Laboratory, Gaithersburg, 23 April 2014. http://jtc1bigdatasg.nist.gov/_uploadfiles/N0028_NBD-PWG_Vol1-Definitions_V1Draft_Pre-release.pdf
37. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113
38. Carney D, Cetintemel U, Cherniack M, Convey C, Lee S, Seidman G, Stonebraker M, Tatbul N, Zdonik S (2002) Monitoring streams: a new class of data management applications. VLDB'02
39. Chandrasekaran S, Cooper O, Deshpande A, Franklin MJ, Hellerstein JM, Hong W, Krishnamurthy S, Madden S, Raman V, Reiss F, Shah M (2003) TelegraphCQ: continuous dataflow processing for an uncertain world. CIDR

40. Balazinska M, Balakrishnan H, Madden SR, Stonebraker M (2008) Fault-tolerance in the Borealis distributed stream processing system. ACM Trans Database Syst, pp 13–24
41. Arasu A, Babcock B, Babu S, Datar M, Ito K, Nishizawa I, Rosenstein J, Widom J (2003) STREAM: the Stanford stream data management system. SIGMOD
42. Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I (2002) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. NSDI 2012, April 2012
43. Ferrucci D (2012) Introduction to 'This is Watson'. IBM J Res Dev 56(3/4), May/July 2012
44. Metropolis N, Howlett J, Rota G-C (eds) (1980) History of computing in the twentieth century. Academic Press, Orlando Florida
45. IBM System/360 (2014) http://www.computerhistory.org/revolution/mainframe-computers/7/161, 22 July 2014
46. Intel's Microprocessor (2014) http://www.computerhistory.org/revolution/digital-logic/12/285, 22 July 2014
47. Murray C, Hoane Jr AJ, Hsu F (2002) Deep blue. Artif Intell 134:57–83
48. Shah H (2011) Turing's misunderstood imitation game and IBM's Watson success. In: AISB 2011 Convention, University of York
49. Hayes P, Ford K (1995) Turing test considered harmful. In: Proceeding of 14th international joint conference artificial intelligence, vol 1, pp 972–977
50. Baughman AK, Chuang W, Dixon KR, Benz Z, Basilico J (2014) DeepQA Jeopardy! gamification: a machine-learning perspective. IEEE Trans Comput Intell AI Games 6(1):55–66

Part II
Mobile and Social Multimedia Data Exploration

Chapter 2
Sentiment Analysis Using Social Multimedia

Jianbo Yuan, Quanzeng You and Jiebo Luo

Abstract Sentiment analysis is one of the most active research areas in natural language processing, web/social network mining, and text/multimedia data mining. The growing importance of sentiment analysis coincides with the popularity of social network platforms, such as Facebook, Twitter, and Flickr, which provide a rich repository of people's opinion and sentiment about a vast spectrum of topics. Moreover, the fact that we are exposed to a tremendous amount of data in different forms including text, images, and videos makes sentiment analysis a very challenging task due to its nature of multimodality. In this chapter, in order to provide a big picture of sentiment analysis, we will discuss some of the latest works on topics of sentiment analysis based on visual content and textual content.

2.1 Introduction

Sentiment analysis and opinion mining are fields of study that analyze people's opinions, evaluations, attitudes, and emotions generally from written language [1]. To the best of our knowledge, the term sentiment analysis first appeared in [2]. Because of the explosive communication and information exchanges using social media, researchers are now given the opportunity to access a tremendous amount of texts and images that express people's opinions and sentiments. Therefore, research in sentiment analysis not only has an important impact on Natural Language Processing, but may also have a profound impact on management sciences, political science, economics, and social sciences as they are all affected by opinions.

J. Yuan · Q. You · J. Luo (✉)
Department of Computer Science, University of Rochester,
Computer Studies Bldg, Rochester, NY 14620, USA
e-mail: jluo@cs.rochester.edu

J. Yuan
e-mail: jyuan10@ece.rochester.edu

Q. You
e-mail: qyou@cs.rochester.edu

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_2

Sentiment analysis applications have spread to almost every possible domain, from consumer products, services, health care, and financial services to social events and political elections. Many big corporations have also built their own in-house capabilities, e.g., Microsoft, Google, Hewlett-Packard, SAP, and SAS. Such practical applications and industrial interests have provided strong motivations for research in sentiment analysis.

Most recently, social networks such as Twitter and microblogs such as Weibo have become major platforms of information exchange and communication between users, between which the common information carrier is tweets. Social networks such as Twitter and microblogs such as Weibo provide billions of pieces of both textual and visual information, making it possible and imperative to detect sentiment indicated by both textual and visual data, respectively. Multimedia content, such as images, are more likely to express and convey people's subtle feelings compared with text information alone [3]. However, sentiment analysis based on a visual perspective is still in its infancy.

With respect to sentiment analysis, much work has been done on textual information [4–6], as well as online sentiment dictionary [7, 8]. Semantics and concept learning [9–12] based on visual features is another way of sentiment analysis without employing textual information. However, semantics and concept learning approaches are hampered by the limitations of computer vision. The analysis of aesthetics [13, 14], interestingness [15] and affect or emotions [16–19] of images are most related to sentiment analysis based on visual content. Moreover, sentiment analysis based on human activities' content, for example, images of facial expressions, has played a significate role in the fields of psycology and human–computer interaction. Many works have been explored on facial emotion classification [20–22]. Hernandez et al. [23] created a computer vision-based system that automatically encouraged, recognized, and counted smiles on a college campus. Hoque et al. [24] took a step forward by building a novel system that provides ubiquitous access to social skills training based on recognitions of facial expressions, speech, and prosody and responds with verbal and nonverbal behaviors.

There are many existing works on sentiment analysis of social media. In particular, Twitter sentiment analysis is one of the most popular research topics. Most existing methods differ in terms of features and emphasize on different aspects of the problem. Guerra et al. [25] proposed a method to measure the bias of social media users toward a topic. Transfer learning is employed to learn the textual features, in such a way that they can build a more accurate classification model using the user biases as a new feature. However, the identification of users' bias on a particular topic itself may be challenging. In [26], the authors employed label propagation to handle noisy labels and use the network for the propagation of the labels. Their results indicate an improvement of accuracy over existing approaches. In [27], the authors used Twitter as a platform to analyze the language characteristics of mothers during postpartum. Their results indicate that using social media can help discover and understand the health and wellness of women following childbirth. Meanwhile, in [28], a method on streaming data sentiment analysis is proposed. The core of the solution is a training augmentation procedure. It will automatically incorporate new relevant messages into the training data. In [29], the authors used the social relations extracted from tweets, and applied graph Laplacian to form a sparse formulation. An optimization algorithm is proposed to solve this problem. All of these methods only use textual features for sentiment analysis. Even though noisy labels and network structures are also considered, these approaches did not combine with image features for sentiment analysis, which is another main content feature of tweets.

Meanwhile, other works related to the mining of different aspects of social networks have also been proposed. Kosinski et al. [30] analyzed the likes in Facebook, and discovered that people in social media are more likely to share some common interests with their friends and some particular community. Based on their model, they are able to predict the behavior of the users according to their online social activities. Rao et al. [31] used Bayesian models for latent attribute detection based on topic models. Goel et al. [32] used social media to study the browsing behavior of online users. Wong et al. [33] used online social network data to quantify political leaning using the information extracted from tweets and retweets. Choudhury et al. [34] analyzed the sentiment or mood in social media. They used valence and activation to represent moods. Their work provided validation of conceptualization of human mood.

For social media networks, the network structure itself can also be employed for the analysis of sentiment propagation of different nodes across the network. In [35], the authors used the hyperlinks in the network to analyze the sentiment flow. Their results indicate that a node is significantly influenced by its immediate neighbors. The structure of information propagation graph also illustrates the impact of different sentiment flow patterns. Similarly, users connected in social networks are more likely to have similar opinions. To analyze sentiment, [36] employed network relationship to analyze the sentiment of a group of users over a particular topic. In [29], both the user-content and the user–user relations were exploited for sentiment analysis. More specifically, they proposed a semi-supervised learning framework by using the network relations and formalized the problem in an optimization framework. An empirical study of the proposed framework over two existing Twitter datasets illustrated the improved performance of the algorithm. You and Luo [3] analyzed the sentiment changes of Twitter users using both textual and visual features.

In the remainder of this chapter we will present some latest works on topics of sentiment analysis based on both textual data and visual data. In Sect. 2.2, we introduce Sentribute, a novel image sentiment analysis framework based on middle level attributes and eigenface expression detection. In Sect. 2.3, we present a new study aimed at analyzing the sentiment changes of Twitter users due to multimedia data including both visual and textual information. We conclude and look forward to future work in sentiment analysis in Sect. 2.4.

2.2 Sentiment Analysis on Visual Contents

As stated, so far analysis of textual information has been well developed in areas including opinion mining [4, 5], human decision making [5], brand monitoring [37], stock market prediction [38], political voting forecasts [4, 39], and intelligence gathering [40]. Figure 2.1 shows an example of image tweets. In contrast, analysis of visual information covers areas such as image information retrieval [41, 42], aesthetics grading [14] and the progress is relatively behind. On the other hand, a recent study shows that images constitute about 36 % of all the shared links on Twitter,[1] which makes visual data mining an interesting and active area to explore. As an old saying has it, an image is worth a thousand words. Much like the textual content-based mining approach, extensive studies have been done regarding aesthetics and emotions in images [13, 15, 43].

Visual content analysis has always been important yet challenging. Thanks to the popularity of social networks, images become a convenient carrier for information diffusion among online users. Aiming to conduct visual content-based sentiment analysis, current approaches include employing low-level features [16, 44, 45], via facial expression detection [46] user intent [47], and understanding images using attribute learning [48, 49]. Sentiment analysis approaches based on low-level features have the limitation of low interpretability, which in turn makes it undesirable for high-level use. Metadata of images is another source of information for high-level feature learning [50]. However, not all images contain such kind of data and researchers are trying to incorporate techniques such as attribute learning and scene understanding before going to final sentiment classification. As for understanding

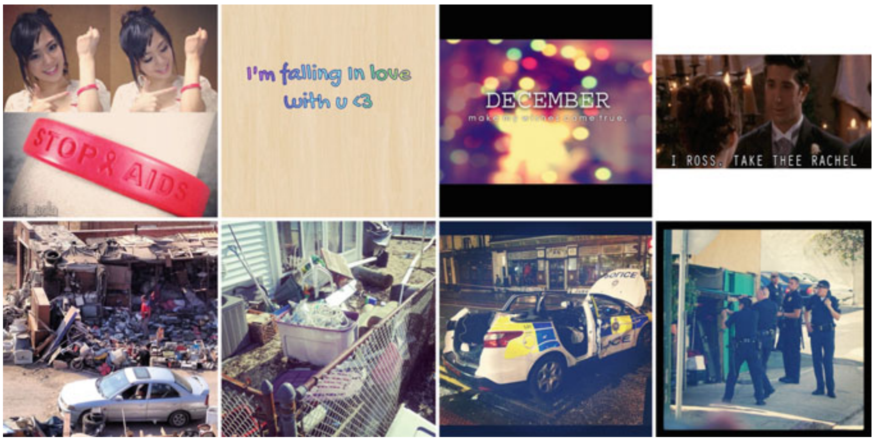


Fig. 2.1 Selected images crawled from Twitter showing (*first row*) positive sentiment and (*second row*) negative sentiments

[1] http://socialtimes.com/is-the-status-update-dead-36-of-tweets-are-photos-infographic/.

the visual concepts of an image, [48] established a large-scale Visual Sentiment Ontology (VSO) consisting of more than 3,000 adjective noun pairs and used as detectors for image sentiment. Sentribute [49], on the other hand, built 102 middle level attributes and used them as features for sentiment classification.

To understand the diffusion patterns and different aspects of the social images, we need to interpret the images first. Similar to textual content, images also carry different levels of sentiment to their viewers. However, different from text, where sentiment analysis can use easily accessible semantic and context information, how to extract and interpret the sentiment of an image remains quite challenging. In this section, we introduce an image sentiment prediction framework, which leverages the mid-level attributes of an image to predict its sentiment. This makes the sentiment classification results more interpretable than directly using the low-level features of an image. To obtain better performance on images containing faces, we employ eigenface-based facial expression detection as an additional mid-level attribute. An empirical study of the proposed framework shows improved performance in terms of prediction accuracy. More importantly, by inspecting the prediction results, we are able to discover interesting relationships between mid-level attribute and image sentiment.

Compared to the state-of-the-art algorithms, the main contribution of Sentribute to this area is two-fold: first, the proposed Sentribute, an image-sentiment analysis algorithm based on 102 mid-level attributes, of which results are easier to interpret and ready-to-use for high-level understanding. Second, we introduce eigenface to facial sentiment recognition as a solution for sentiment analysis on images containing people. This is simple but powerful, especially in cases of extreme facial expressions, and contributed an 18 % gain in accuracy over decision making only based on mid-level attributes, and 30 % over the state-of-art methods based on low-level features.

2.2.1 Framework Overview

Figure 2.2 presents the proposed Sentribute framework. The idea for this algorithm is as follows: first, we extract scene descriptor low-level features from the SUN Database [47] and use these four features to train the classifiers by Liblinear [16] for generating 102 predefined mid-level attributes, and then use these attributes to predict sentiments. Meanwhile, facial sentiments are predicted using eigenfaces. This method generates really good results, especially in cases of predicting strong positive and negative sentiments, which makes it possible to combine these two predictions and generates a better result for predicting image sentiments with faces. To illustrate how facial sentiment helps refine our prediction based on only mid-level attributes, we present an example in Sect. 2.4, of how to correct the false positive/negative prediction based on facial sentiment recognition.

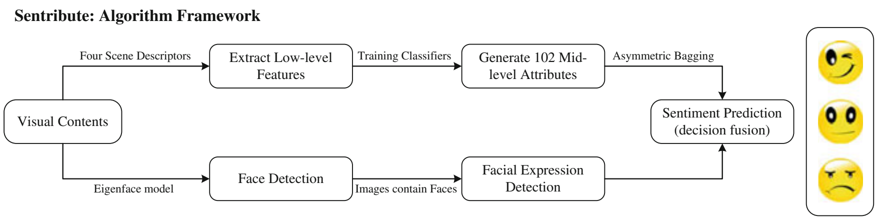

Sentribute: Algorithm Framework
Four Scene Descriptors
Extract Low-level Features
Training Classifiers
Generate 102 Mid-level Attributes
Asymmetric Bagging
Visual Contents
Sentiment Prediction (decision fusion)
Eigenface model
Face Detection
Images contain Faces
Facial Expression Detection

Fig. 2.2 Selected images crawled from Twitter showing **a** positive sentiment and **b** negative sentiments

2.2.2 Sentribute

In this section we outline the design and construction of the proposed Sentribute, a novel image sentiment classification method based on mid-level attributes, together with a decision refine mechanism for images containing people. For image sentiment analysis, we conclude the procedure starting from dataset introduction, low-level feature selection, building mid-level attribute classifier,, and image sentiment classification. As for facial sentiment recognition, we introduce eigenface to fulfill our intention.

Dataset: Our proposed algorithm mainly contains three steps: first is to generate mid-level attributes labels. For this part, we train our classifier using SUN Database,[2] the first large-scale scene attribute database, initially designed for high-level scene understanding and fine-grained scene recognition [51]. This database includes more than 800 categories and 14,340 images, as well as discriminative attributes labeled by crowd-sourced human studies. Attributes labels are presented in the form of zero to three votes, of which 0 vote means this image is the least correlated with this attribute, and three votes means the most correlated as shown in Fig. 2.3. Due to this voting mechanism, we have an option of selecting which set of images to be labeled as positive: images with more than one vote, introduced as soft decision (SD), or images with more than two votes, introduced as hard decision (HD). Mid-level attribute classifiers learned based on soft decisions are less likely to be overfitting and less accurate than the classifiers learned based on hard decisions.

The second step of our algorithm is to train sentiment predicting classifiers with images crawled from Twitter together with their textual data covering more than 800 images. Twitter is currently one of the most popular microblog platforms. Sentiment ground truth is obtained from visual sentiment ontology[3] with permission of the authors. The dataset includes 1,340 positive, 223 negative and 552 neutral image tweets. For testing, we randomly select 810 images, containing positive (660 image tweets) and negative (150 image tweets). Figure 2.1 shows images chosen from our dataset as well as their sentiment labels.

2 http://groups.csail.mit.edu/vision/SUN/.

3 http://www.ee.columbia.edu/ln/dvmm/vso/download/sentibank.html/.

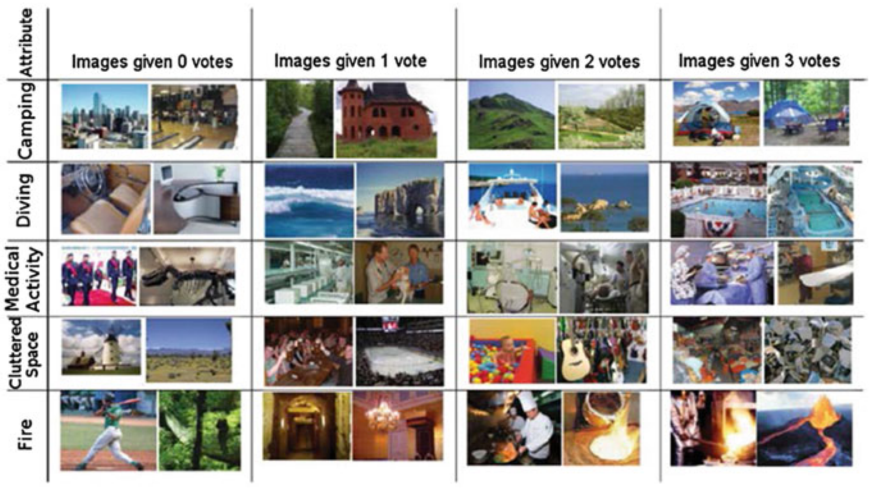


Fig. 2.3 The images in the table above are grouped by the number of positive labels (votes) received from AMT workers. From *left* to *right* the visual presence of each attribute increases [51]

The final step is facial emotion detection for decision fusion mechanism. We chose to use the Karolinska Directed Emotional Faces dataset [52] mainly because the faces are all well aligned with each other and have consistent lighting, which makes generating good eigenface much easier. The dataset contains 70 men and women over 2 days expressing seven emotions (scared, anger, disgust, happy, neutral, sad, and surprised) in five different poses (front, left prole, right prole, left angle, right angle).

Feature Selection: In this part, we aim to select low-level features for generating mid-level attributes, and we choose four general scene descriptors: GIST descriptor [18], HOG 2x2, self-similarity (SSIM), and geometric context color histogram (GEO-COLOR-HIST) features [53]. These four features were chosen because they are each individually powerful and because they can describe distinct visual phenomena in a scene perspective other than using specific object classifier. These scene descriptor features suffer neither from the inconsistent performance compared to commonly used object detectors for high-level semantics analysis of an image, nor from the difficulty of result interpretation generated based on low-level features.

Generating Mid-level Attribute: Given the selected low-level features, we are then able to train our mid-level attribute classifiers based on SUN Database. We have 14,340 images as training data, and the low-level features of each image add up to more than 170,000 dimensions. For classifier options, Liblinear[4] outperforms against LibSVM[5] in terms of training time and maintains similar performance in accuracy in cases where the number of samples are huge and the number of feature dimensions

[4] http://www.csie.ntu.edu.tw/~cjlin/liblinear/.

[5] http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

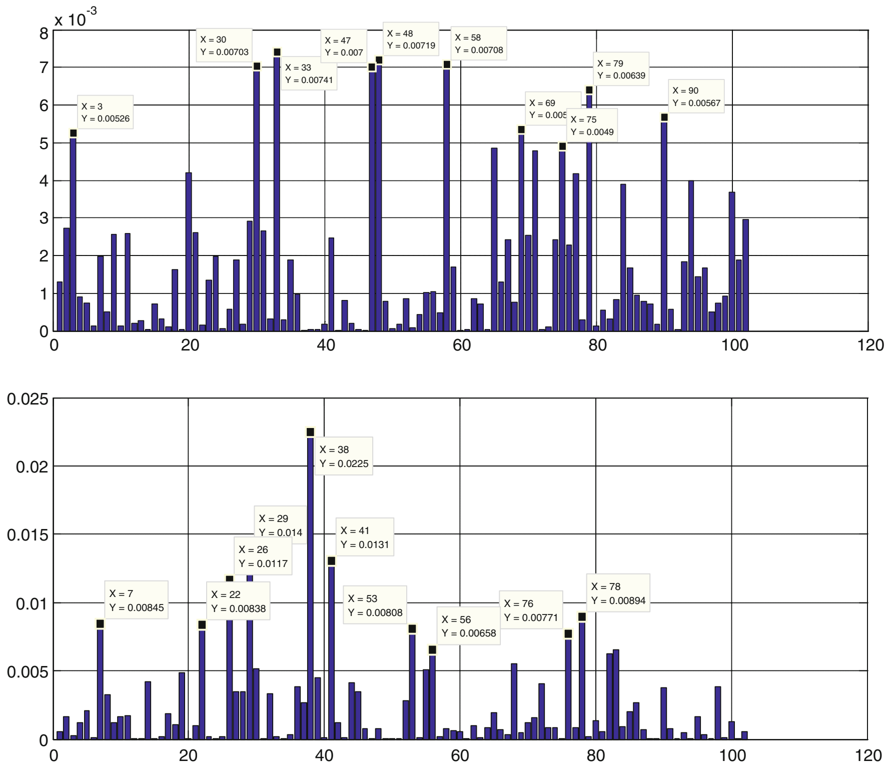


Fig. 2.4 Computing mutual information for each label (*first row* is based on SD and *second row* is based on HD), where *X* label indicates the number of each feature and *Y* label stands for the MI value

is huge. Therefore, we choose Liblinear toolbox to implement SVM algorithm to achieve time saving.

The selection of mid-level attribute also plays an important part in image sentiment analysis. We choose 102 predefined mid-level attributes based on the following criteria: (1) have descent detection accuracy, (2) potentially correlated to one sentiment label, and (3) easy to interpret. We then select four types of mid-level attributes accordingly: (a) Material: such as metal, vegetation; (b) Function: playing, cooking; (c) Surface property: rusty, glossy; and (d) Spatial Envelope [18]: natural, man-made, enclosed.

We conduct mutual information (MI) analysis to discover mid-level attributes that are most correlated with sentiments. Mutual information is a measure of variables' mutual dependence (Fig. 2.4).

For each mid-level attribute, we computed the MI value with respect to both positive and negative sentiment categories (Fig. 2.4). Table 2.1 illustrates the 10 most distinguishable mid-level attributes for predicting both positive and negative labels

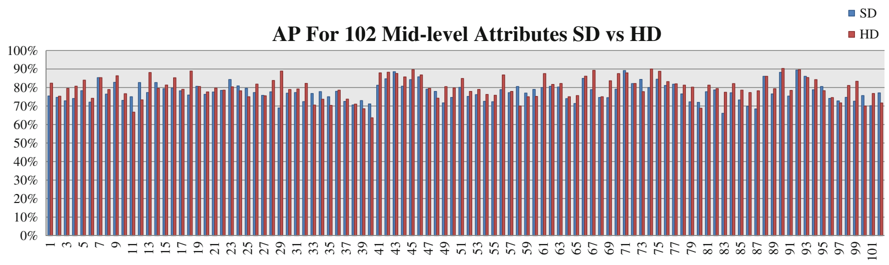


Fig. 2.5 AP of the 102 attributes based on SD and HD

in a descending order based on both SD and HD. Figure 2.5 demonstrates Average Precision (AP) for the 102 attributes we selected, for both SD and HD. It is not surprising to see that attributes of material (flowers, trees, ice, still water), function (hiking, gaming, competing) and spatial envelop (natural light, congregating, aged/worn) all play an important role based on the result of mutual information analysis.

Image Sentiment Classification: In our dataset we have 660 positive samples and 150 negative samples. It is likely to obtain a biased classifier based on these samples alone. Therefore, we introduce asymmetric bagging [54] to deal with biased dataset. Figure 2.6 presents the idea of asymmetric bagging: instead of building one classifier, we now build several classifiers, and train them with the same negative samples together with different sampled positive samples of the same amount. Then we can combine their results and build an overall unbiased classifier.

Facial Sentiment Recognition: Our proposed algorithm, Sentribute, contains a final step of decision fusion mechanism by incorporating eigenface-based emotion detection approach. Images containing faces contribute to a great partition of the whole images so that, 382 images from our dataset have faces. Therefore, facial emotion detection is not only useful but important for the overall performance of our algorithm.

In order to recognize emotions from faces we use classes of eigenfaces corresponding to different emotions. Eigenface was one of the earliest successful implementations of facial detection [55]; we modify the algorithm to be suitable for detecting classes of emotions. Although this method is widely appreciated already, we are the first to modify the algorithm to be suitable for detecting classes of emotions, and this method is simple yet surprisingly powerful for detecting facial emotions for front and consistent lightened faces. Note that we are not trying to propose an algorithm that outperforms the state-of-the-art facial emotion detection algorithms. This is beyond the scope of this section.

According to Ekman [56], there are six principal emotions that human's experience: fear, anger, disgust, happiness, sadness, and surprise. Due to the accuracy of the model and the framework of integrating the results with Sentribute, we reduce the set of emotions to positive, neutral, and negative emotions. This is done by classifying the image as one of the seven emotions and then mapping the happy and surprised emotions to positive sentiment, neutral sentiment to itself, and all other emotions to

Table 2.1 Attributes with the top 10 mutual information

TOP 10	Soft decision	Hard decision
1	Congregating	Railing
2	Flowers	Hiking
3	Aged/worn	Gaming
4	Vinyl/linoleum	Competing
5	Still water	Trees
6	Natural light	Metal
7	Glossy	Tiles
8	Open area	Direct sun/sunny
9	Glass	Aged/worn
10	Ice	Constructing

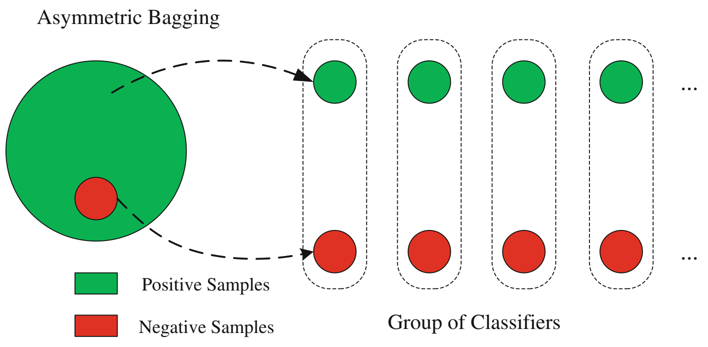


Fig. 2.6 Asymmetric bagging

negative sentiment. At a high level, we are computing the eigenfaces for each class of emotion; we then compare the features of these eigenfaces with the features of the target image projected onto the emotion class space.

The algorithm requires a set of faces to train the classifier (more specifically to find the features of the images). We chose to use the Karolinska Directed Emotional Faces (KFEF) dataset [52] for many reasons, specifically the faces are all well aligned with each other and have consistent lighting, which makes generating good eigenfaces much easier. The dataset contains 70 men and women over 2 days expressing seven emotions (fear, anger, disgust, happy, neutral, sad, and surprised) in five different poses (front, left profile, right profile, left angle, right angle). We use a subset of the KDEF database for our training set, only using the seven frontal emotions from one photographing session.

Training the dataset and extracting the eigenfaces from the images of each emotion class was accomplished by using principal component extraction. We preprocess the training data by running it through fdlibmex,[6] a fast facial detection algorithm to obtain the position and size of the face. We then extract the face from the general image and scale it to a 64×64 grayscale array; it is then vectored into a 4,096 length vector. We concatenate the individual faces from each class into an $M \times N$ array X, where M is the length of each individual image and N is the number of images in the class. We then are able to find the eigenfaces by using Principal Component Extraction. Principal component extraction converts correlated variables, in our case a set of images, into an uncorrelated variables via an orthogonal transform. We implement principal component analysis by first computing the covariance matrix

$$C = (x - \mu)(x - \mu)^T, \tag{2.1}$$

where μ is the vector of empirical mean of matrix X over each row. The eigenvectors of C (donated by E^c where c is the emotion class) are then calculated and arranged by decreasing eigenvalues. Only the 20 largest eigenvectors are chosen for each class of facial emotions. The principal eigenfaces are simply the eigenvectors of the system that have the largest eigenvalues. We compute the features F^c of class c as shown below.

$$E^c = PCA(X^c) \tag{2.2}$$

$$F^c = E^c(X^c - \mu^c) \tag{2.3}$$

In order to classify the target image preprocessing is necessary to preprocess the image as we preprocess the training dataset, which we will denote y. The classification of a test face is performed by comparing the distance of the features of the target face (projected onto the emotion subspace) to the features of the eigenfaces of the subspace. We then choose the class that minimizes this function as the predicted class, specifically

$$\arg\min_c \sum_i \parallel E_i^c(y - \mu) - F_i^c \parallel, \tag{2.4}$$

where i is each individual feature column vector in the array [55].

Given the distance value we are able to set a threshold value in order to filter out results that are weakly classified. Figure 2.7 shows examples of classified facial emotions.

2.2.3 Experiments

Image Sentiment Classification: As mentioned before, state-of-the-art sentiment analysis approach can be mainly concluded as: (1) textual information-based

[6] http://www.mathworks.com/matlabcentral/fileexchange/20976.

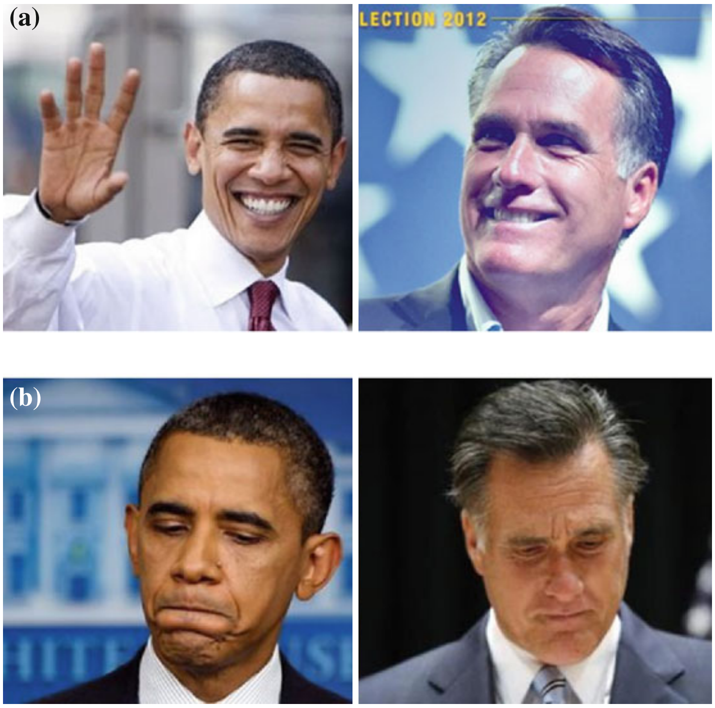


Fig. 2.7 Examples of eigenface-based emotion detection. **a** Classification: Positive. **b** Classification: Negative

sentiment analysis, as well as online sentiment dictionary [7, 8] and (2) sentiment analysis based on low-level features. Therefore, in this section, we set three baselines: (1) low-level feature-based approach, (2) textual content-based approach [8], and (3) online sentiment dictionary SentiStrength [7].

1. *Image Sentiment Classification Performance*:

First we demonstrate results of our proposed algorithm, image sentiment classification based on 102 mid-level attributes (SD vs. HD). Both Linear SVM and Logistic Regression algorithms are employed for comparison.

As demonstrated in Table 2.2, performance of precision for both Linear SVM and Logistic Regression outperforms that of recall. Due to the benefits of using asymmetric bagging, we are now able to raise the classification accuracy of negative samples. Smaller number of false positive samples and relatively larger number of detected true positive samples contribute to this unbalanced value of precision and recall performance.

The next thing we are interested in is the comparison against baseline algorithms.

2. *Low-level Feature-Based and Textual Content-Based Baselines*:

For low-level feature-based algorithm, Ji et al. employed the following visual features: a dimensional color histogram extracted from the RGB color space, a 512-dimensional GIST descriptor [18], a 53-dimensional local binary pattern

Table 2.2 Image sentiment classification performance

		Precision (%)	Recall (%)	Accuracy (%)
Linear SVM	SD	82.6	56.8	55.2
	HD	86.7	59.1	61.4
Logistic regr	SD	84.3	54.7	54.8
	HD	88.1	58.8	61.2

Table 2.3 Accuracy of sentiment classification

(a) Comparison between low-level-based algorithm and mid-level-based algorithm

	SVM (low)	Logistic regr (low)	SVM (mid)
AC	50 %	53 %	61.4 %

(b) Comparison between mid-level visual content-based algorithm and textual content-based algorithm

	Contextual polarity	Sentistrength	SVM (mid)
AC	61.7 %	61 %	61.4 %

(LBP), a bag-of-words quantized descriptor using a 1,000 word dictionary with a two-layer spatial pyramid, and a 2,659-dimensional Classemes descriptor. Both linear SVM and logistic regression algorithms are used for classification. For textual content-based algorithm, we choose contextual polarity, a phrase-level sentiment analysis system [6], as well as SentiStrength API.[7] Table 2.3 shows the results of accuracy based on low-level features, mid-level attributes, and textual contents.

Decision Fusion: The final step of Sentribute is decision fusion. By applying eigenface-based emotion detection, we are able to improve the performance of our decision based on mid-level attributes only. We only take into account images with complete face with reasonable lighting condition. Therefore among all the images with faces, we first employ a face detection process and generate a set of 153 images as the testing dataset for facial emotion detection and decision fusion. For each face we detected, we assigned them a label indicating sentiments: 1 for positive, 0 for neutral, and -1 for negative sentiments. We thus computed a sentiment score for each image as a whole. For instance, if we detect three faces from an image, two of them are detected as positive and one of them is detected as neutral, then the overall facial sentiment score of this image is 2. These sentiment scores can be used for decision fusion with the decision made based on mid-level attributes only, i.e., we add up the facial sentiment score and the outputs of the classifiers based on mid-level attributes only returned by our classifiers to implement a decision fusion mechanism. Table 2.4 shows the improvements in accuracy after decision fusion.

[7] http://sentistrength.wlv.ac.uk/.

Table 2.4 Accuracy of Sentribute algorithm

	Accuracy (%)
Mid-level-based prediction	64.71
Facial emotion detection	73.86
Sentribute (after synthesis)	82.35

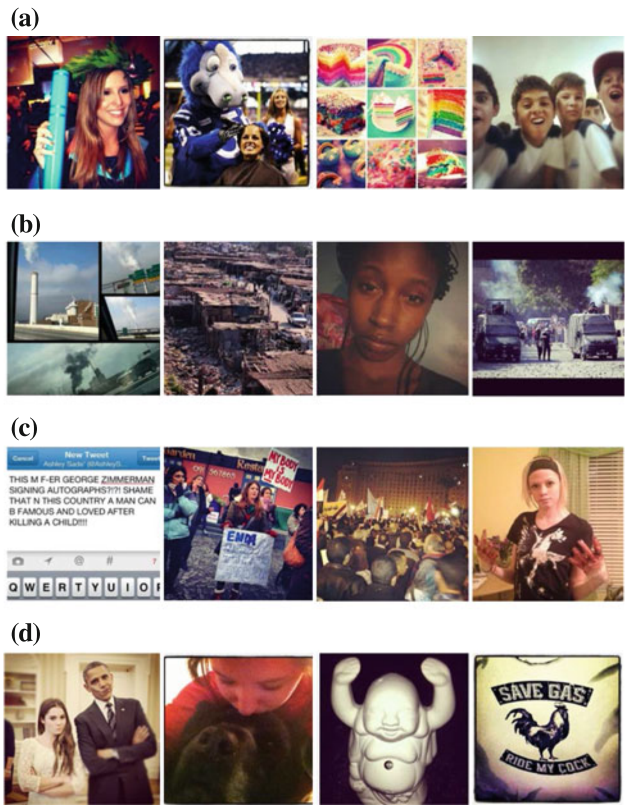
Figure 2.8 presents examples of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) samples generated by Sentribute. False classified samples show that it is hard to distinguish images only containing texts from both positive and negative labels, and images of big event/celebration (football game or a concert) from those of protest demonstration. They both share similar general scene descriptors, similar lighting condition, and similar color tone. Another interesting false detected sample is the first image shown in false negative samples. Figures make frown expression on their faces, however the sentiment behind this expression is positive since they were meant to be funny. This sample is initially classified as positive based on mid-level attributes only, and then refined as negative because two strong negative facial expressions are detected by our eigenface expression detector. These kinds of images show a better decision fusion metric would be one of our potential improvements.

2.2.4 Conclusion

In this section, we present Sentribute, a novel image sentiment analysis algorithm based on mid-level attributes. Asymmetric bagging approach is employed to deal with unbalanced training data. To enhance the classification performance, eigenface-based emotion detection algorithm is applied, to deal with images containing faces and achieve a significant gain in accuracy over results based on mid-level attributes alone. The proposed algorithm explicitly explores visual content for sentiment analysis by employing mid-level attributes and without using textual content.

2.3 Sentiment Analysis in Multimedia Tweets

Online social networks have attracted the attention of people from both the academia and real-world. In particular, the rich multimedia information accumulated in recent years provides an easy and convenient way for more active communication between people. This offers an opportunity to research people's behaviors and activities based on those multimedia content that can be considered as social imagematics. One emerging area is driven by the fact that these massive multimedia data contain people's daily sentiments and opinions. However, existing sentiment analysis typically only pays attention to the textual information regardless of the visual content,

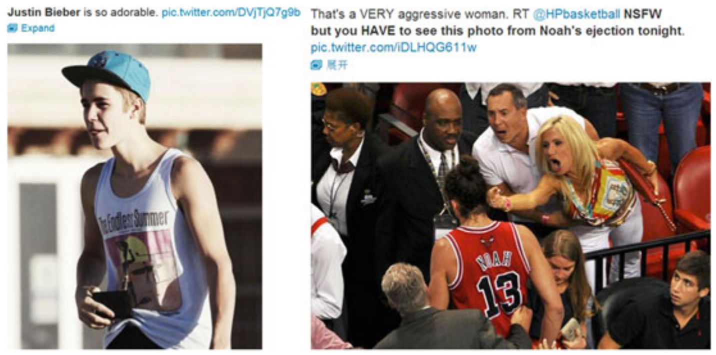


THIS M F-ER GEORGE ZIMMERMAN SIGNING AUTOGRAPHS?!?! SHAME THAT N THIS COUNTRY A MAN CAN B FAMOUS AND LOVED AFTER KILLING A CHILD!!!!

Fig. 2.8 Examples of sentiment detection results by Sentribute. **a** True positive samples. **b** True negative samples. **c** False positive samples. **d** False negative samples

which may be more informative in expressing people's sentiments and opinions. In this section, we attempt to analyze the online sentiment changes of social media users using both the textual and visual content. In particular, we analyze the sentiment changes of Twitter users using both textual and visual features. An empirical study of real Twitter datasets indicates that the sentiments expressed in textual content and visual content are correlated. The preliminary results in this section give insight into the important role of visual content in online social media.

Twitter is one of the most influential social networks across the world. Research work of different topics related to Twitter has been published in different conference venues. The large amount of daily generated user content attracted many researchers around the world to analyze potential interesting patterns in social media, including prediction of political election, sentiment analysis, information diffusion, topic trend, etc. However, it should be noted that at the beginning, Twitter as a social platform only allows a maximum of 140 characters to compose users' messages. However, things changed in 2011, when Twitter allowed online users to post images in their


Justin Bieber is so adorable. pic.twitter.com/DVjTjQ7g9b

That's a VERY aggressive woman. RT @HPbasketball NSFW but you HAVE to see this photo from Noah's ejection tonight. pic.twitter.com/iDLHQG611w

Fig. 2.9 Example of an image tweet, where the *left* image shows a picture of justin Bieber and the *right* image shows the ejection of Noah during the NBA playoffs

tweets. We denote the tweets that contain images as image tweets. The impacts of image tweets are tremendous. This part will focus on one particular impact of image tweets, namely the impact on sentiment analysis.

Multimedia content, like images, are more likely to express and convey people's subtle feelings compared with text information [3]. With the popularity of smartphones and convenient social media APPs, more and more people are likely to post image tweets to attract attention from other users in Twitter. Figure 2.9 shows an example of an image tweet, where the big picture conveys more information about the Tweet.

One of the most interesting aspects of Twitter is that people's sentiments in Twitter seem to be related to real social life. For instance, in [57], the authors found that the sentiment changes of Twitter users are closely related to the overall economy situations in the U.S. and the stock market. However, most research on sentiment changes are related to the overall text tweets. Little attention has been paid to the analysis of image tweets. The work described in this part is an attempt toward the analysis of sentiment conveyed in the multimedia content in tweets. We intend to investigate social multimedia analysis, which we refer to as social imagematics. We conduct an empirical study on the sentiments expressed in people's tweets, especially the impact of sentiments in image tweets.

2.3.1 Approaches

As discussed in Sect. 2.1, there are many existing works on sentiment analysis using textual features. In this section, we employ existing algorithms to analyze the sentiment of the textual tweets. For the sentiment analysis of visual features, we build classifiers using low-level and mid-level respectively.

Textual Sentiment Analysis: There are many related works on sentiment analysis of Twitter [26, 29, 33, 36]. Meanwhile, there are also many online services that provide easy access API to evaluate the sentiment of online tweets. Many of these tools [8] come directly from the academic research. Since we are more concerned with image tweets and the sentiment of images, we directly use existing online service for the sentiment analysis of collected tweets.

In particular, we use the sentiment140[9] [58]. Sentiment140 is a semi-supervised machine learning approach. It exploits emotions as noisy labels for training data. Moreover, it provides convenient API for the sentiment analysis of different tweets. Typically, one can send the data to the server using HTTP request. The server then returns the sentiment for each line contained in that file. The returned value in this file contains three different values (0, 2, and 4). Here 0 represents the negative sentiment, 4 represents the positive sentiment, and 2 means neutral. In this way, we are able to classify the tweets into different sentiment categories.

Sentiment Changes with the Number of Images: Users in Twitter generally preferred different types of tweets. Some of the users like to post many image tweets, while many other users love to post traditional text tweets. To analyze the sentiments of users with different preferences over image tweets, we conduct an experiment on the relation between the proportion of image tweets and the proportion of positive tweets. We use the textual sentiment analysis in Sect. 2.3.1 to analyze the sentiments of different users. Then, the number of positive tweets over the sum of positive and negative tweets is used to represent the proportion of positive sentiment.

We randomly picked about 300 users and downloaded their tweets using the user timeline API. Figure 2.10 shows that users who like to post many image tweets are more likely to have positive sentiments. On the other hand, for users with fewer proportion of image tweets, the proportion of positive sentiments among these users varies significantly.

Visual Sentiment Classification: Image sentiment analysis is quite challenging. As discussed in [59], the authors used the textual sentiment analysis as the rough labels of the corresponding images. Then, RGB Hist and SIFT features are employed to train a classifier and classify the test images. Their results indicate that the positive and negative sentiments seem to share different interesting image patterns.

In our implementation, we use the image sentiment corpora from visual sentiment ontology[10] with kind permission from the authors. Then according to the dataset, we trained two levels of classifiers. The first classifier only uses the low-level features, which include HOG [60], GIST [18], SSIM [61], and GEO-COLOR-HIST [62]. Different features have different advantages over different tasks [53]. HOG is good for object and human recognition. GIST is another feature designed for scene recognition. On the other hand, SSIM provides measure of invariant scene layout. Meanwhile, geometric color histogram offers a robust histogram feature, which is invariant of scene layout.

[8] http://matei.org/ithink/2012/02/08/a-list-of-twitter-sentiment-analysis-tools/.

[9] http://www.sentiment140.com/.

[10] http://visual-sentiment-ontology.appspot.com/.

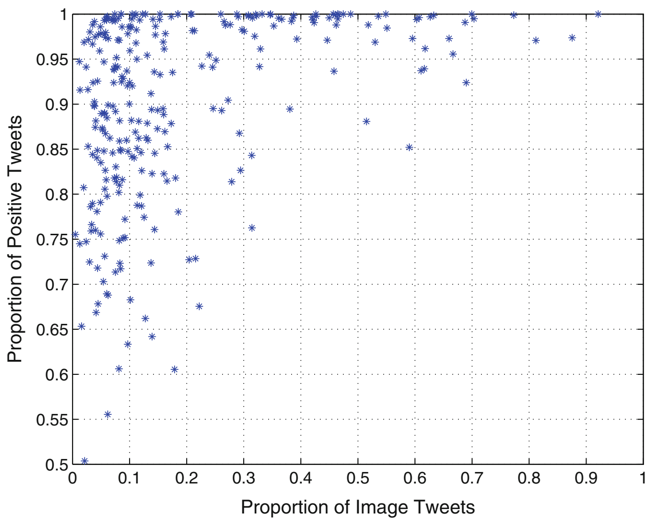


Fig. 2.10 Relationship of proportion of image tweets and the proportion of positive tweets

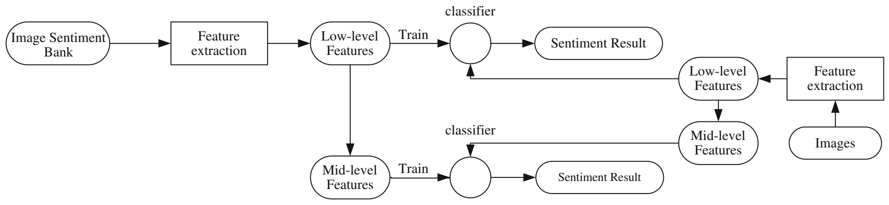


Fig. 2.11 Framework of image sentiment classification using low-level and middle level features respectively

The low-level features can be easily extracted from the given images. Figure 2.11 shows the framework employed for image sentiment classification. The main component in this framework is the low-level and middle level image features. Accordingly, there are two classifiers. In our implementation, we choose liblinear[11] as the classifier for both levels due to its scalability in large-scale learning. The first classifier is based on the low-level features discussed above. Based on these low-level features, we also train and learn some middle level features. Middle level features are more interpretable than low-level features. In our implementation, we use the middle level features described in Table 2.5. For each middle level feature, we need to train a classifier, which can determine whether or not the given image contains the corresponding middle level description. By combining all the middle level features, we are able to construct a middle level features description for the given image set. Then, a second-level classifier based on the extracted middle level features is constructed and employed to classify the test images into different sentiment categories.

[11] http://www.csie.ntu.edu.tw/~cjlin/liblinear/.

Table 2.5 Summary of the middle level features used in this study

Dirt/soil	Matte	Man-made	Rugged scene
Natural light	Dirty	Open area	Cluttered space
Direct sun/sunny	Rusty	Semi-enclosed area	Scary
Electric/indoor lighting	Arm	Enclosed area	Soothing
Aged/orn	Cold	Far-away horizon	Stressful
Glossy	Natural	No horizon	

For all the images contained in image tweets, we then download these images according to the URL contained in the metadata of each image tweet. Then low-level and middle level features are extracted using the same procedure for the training images. In this way, we are able to classify the sentiment of image tweets according to the visual features of the images contained in image tweets.

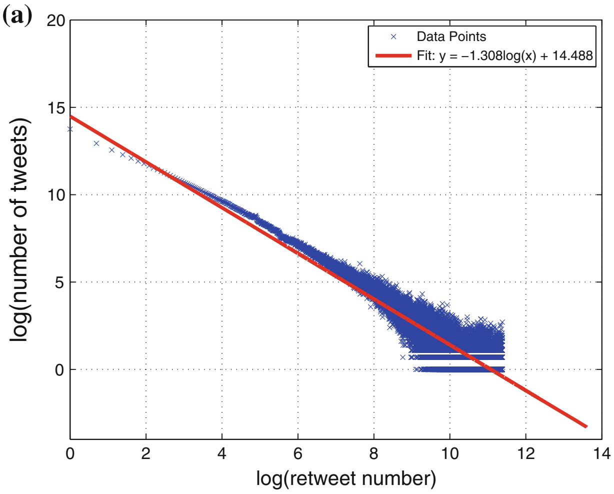
2.3.2 Experiments

We collect tweets using online Twitter API.[12] Twitter provides different categories of API. We mainly use the Twitter streaming API and Twitter timeline API. In order to choose some relatively active users, we use the streaming API to download over 19 million tweets. Active users simply refer to users who tweet, reply, and retweet more than others over a certain period of time. We chose empirical thresholds (over 100 original tweets in 1 month) to determine the relatively active users. To store such a large amount of data, we use couchdb,[13] a document database, to store the download tweets. Then, by analyzing the downloaded 19 million tweets, we are able to identify the activity levels of different online users. First we identify over 8,000 users, and we use the timeline API to download the tweets of these users. We collected over 20 million tweets for all the 8,000 users. Next, the tweets of these 8,000 users are further analyzed. Among these 8,000 Twitter users, we further pick out about 300 users who are relatively active in posting both text and image tweets based on the threshold we mentioned because we want to analyze the correlation between sentiments behind text tweets and image tweets. Given these users and the URL contained in their image tweets, we collect all the users' posted images. We got over 90,000 thousand images for these active Twitter users.

In the downloaded 25 million tweets, we analyze the proportion of image tweets. Over the 25 million tweets, about 6 million tweets are image tweets ($5{,}988{,}058/25{,}580{,}000 = 0.23$). About every 1 in 4 tweets contains images in Twitter. Figure 2.12 shows the distribution of number of retweets. Similar to many other user activities, the distribution is a power law distribution with long tail. Figure 2.12a, b shows that the number of image retweets share a similar distribution,

[12] https://dev.twitter.com/.

[13] http://couchdb.apache.org/.

(a)


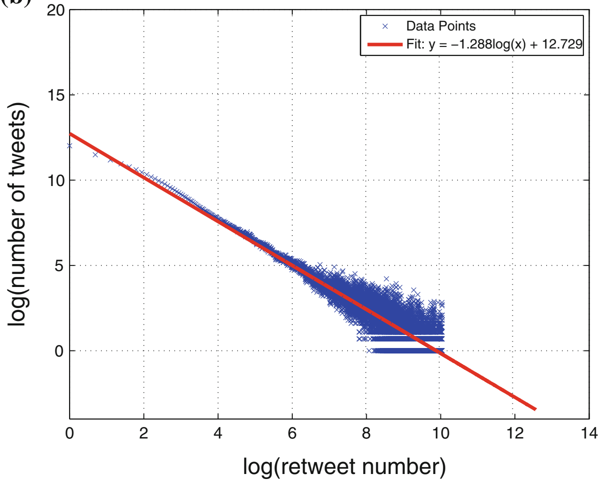
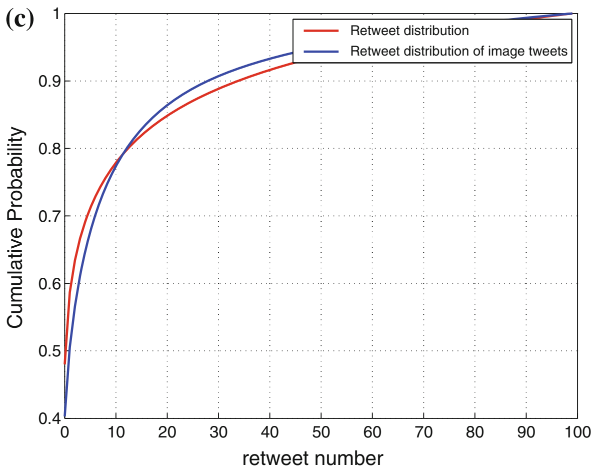
(b)


Fig. 2.12 Statistics of retweets number for all tweets and image tweets only. **a** Distribution of number of retweets. **b** Distribution of number of image retweets. **c** Probability distribution of *top* retweets number between 1 and 100. **d** Probability distribution of *top* retweets number between 1 and 1,000

with a slight difference in the slope of the fitted line of the log-log plot of the distribution. If we further look at the cumulative probability distribution of retweets number for all tweets and image tweets only, we can conclude from Fig. 2.12c, d that compared to image tweets, the proportion of tweets that received small number of retweets takes a larger proportion than image tweets. This evidence also verifies the fact that image tweets are more likely to attract online users' attention and are more easily diffused in the social network.



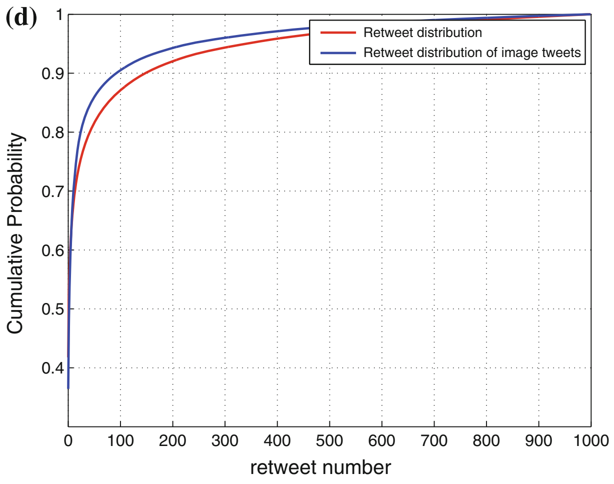


Fig. 2.12 (continued)

Correlation of Sentiment Between Image Tweets and Text Tweets: To illustrate the correlation between text and image tweets, we randomly select 10 users from the 300 users. We employ the methods discussed in Sect. 2.3.1. The sentiment analysis results using text and image features are shown in Figs. 2.13 and 2.14. In both figures, the red line represents the sentiment changes of each user according to the sentiment analysis of using text tweets, while the blue line represents the sentiment changes of each user according to the sentiment analysis of image tweets. The blue lines in the left column give the sentiment analysis using low-level image features, while the blue lines in the right column give the sentiment analysis using middle level image features. In Fig. 2.13, we average the long-term sentiment for each user in terms of

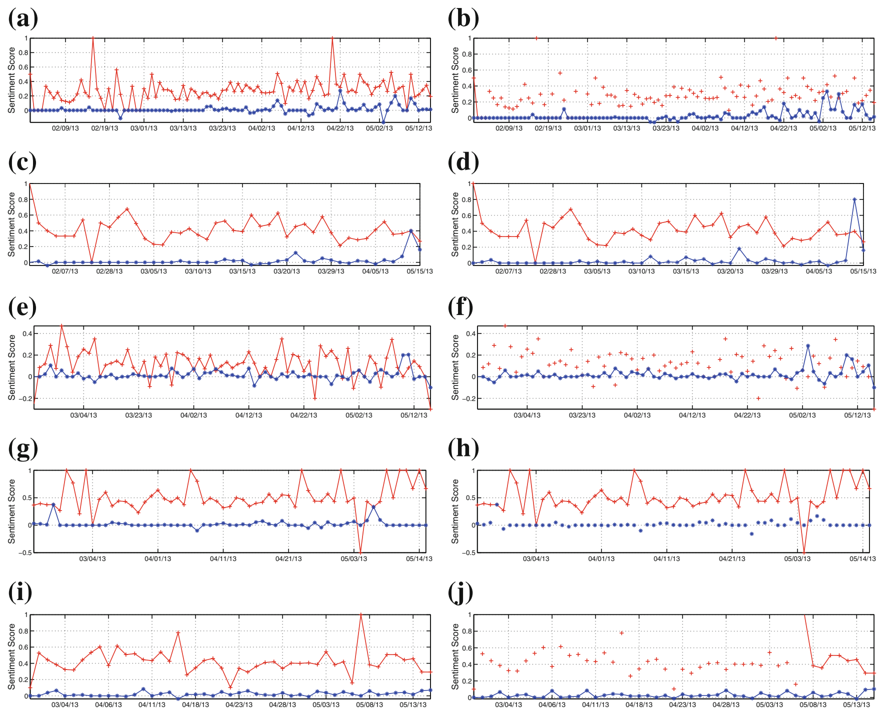


Fig. 2.13 Long-term sentiment changes of tweets and image tweets using low-level and mid-level features. The *red line* represents the sentiment of each user using the textual features and the *blue line* represents the sentiment of each user using the visual features from the image tweets. **a** 110914277. **b** 110914277. **c** 1135866961. **d** 1135866961. **e** 183352499. **f** 183352499. **g** 320657019. **h** 320657019. **i** 341587111. **j** 341587111. **k** 606333611. **l** 606333611. **m** 745235832. **n** 745235832. **o** 910880371. **p** 910880371. **q** 924674300. **r** 924674300. **s** 98005782. **t** 98005782

days, which means that each point represents the average sentiment score for a user. Similarly, in Fig. 2.14, the sentiment is averaged in terms of 1 h.

Table 2.6 shows the correlation coefficients between sentiment of the selected users using text features and image features. Although there is noise in the prediction of user's sentiment, the results indicate that there is still positive correlation between the sentiment expressed in text tweets and image tweets. In particular, for user 606333611, the sentiments are highly correlated. The reasons for this may include two aspects. First, we see this user is a relatively more active user. This can be reflected by the date in the x-axis of the figure. Since Twitter only allows us to download up to 3,200 of a user's most recent statuses, therefore, this user posted many tweets in a relatively short period. Second, there is no negative sentiment predicted by the text tweets. At the same time, for some users, they only have positive sentiment (there is no negative and neutral sentiment), thus the correlation is unavailable.

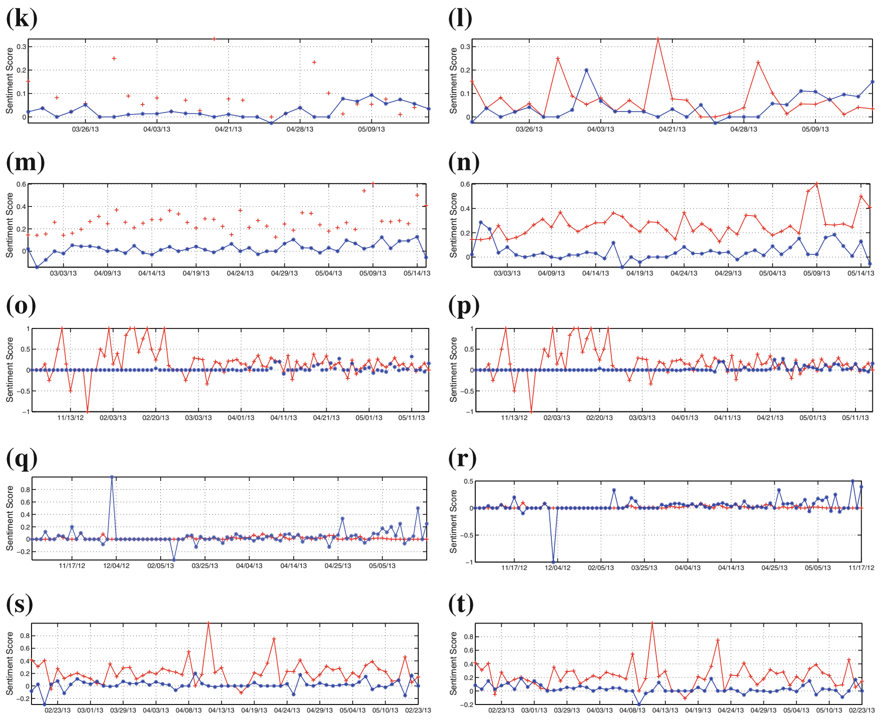


Fig. 2.13 (continued)

However, overall we see that sentiment classification using middle level features seems to be more correlated with the sentiment of using text tweets.

Correlation of Sentiment in a Shorter Period: The above results are averaged in terms of a day. This may not reflect people's sentiment fluctuation in a particular day. In this section, we average the short-term sentiment of a user in terms of an hour. The results are shown in Fig. 2.14. The results indicate that different users have different sentiment change patterns. Some users are more likely to have emotional fluctuation in terms of both text and image tweets. For some users, their sentiment changes are reflected by text tweets. Meanwhile, some users are more likely to post images to express their sentiment changes. There is a correlation between the sentiment changes for the randomly selected 10 users. Table 2.7 shows the correlation coefficients for the 40 most recent periods. Different from the results in terms of days, in this case some of the correlation coefficients are negative. However, for most users, the correlation coefficients are mostly positive. The results of using low-level visual features and middle level visual features are not consistent all the time. The results on one hand indicate the difficulty in image sentiment analysis. On the other hand, they also illustrate the different patterns of online users in expressing their sentiment.

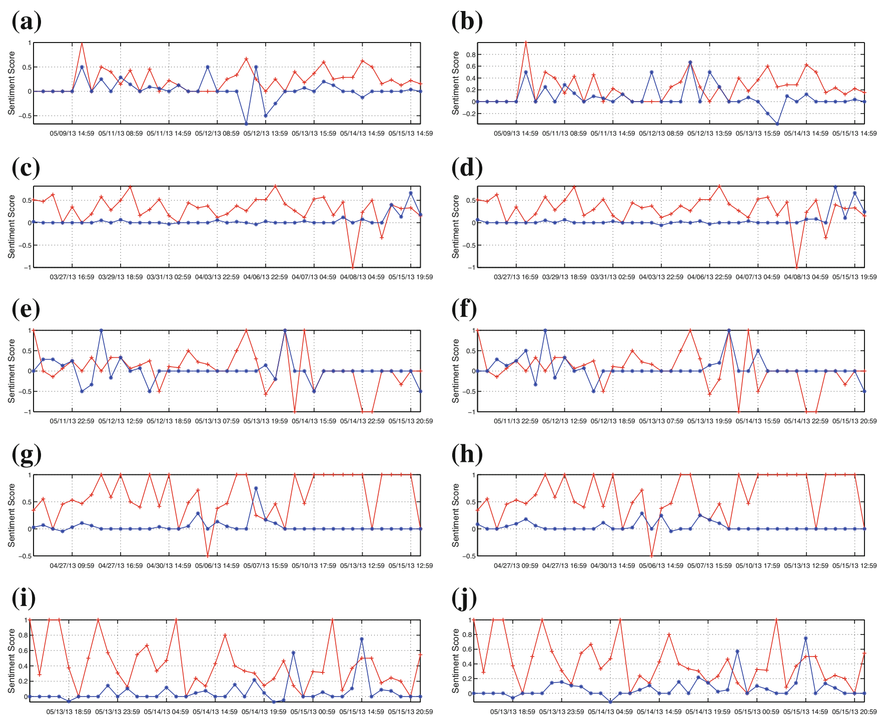


Fig. 2.14 Short-term sentiment of the recent 40 periods. We choose 1 h in which the users posted tweets as one short period. **a** 110914277. **b** 110914277. **c** 1135866961. **d** 1135866961. **e** 183352499. **f** 183352499. **g** 320657019. **h** 320657019. **i** 341587111. **j** 341587111. **k** 606333611. **l** 606333611. **m** 745235832. **n** 745235832. **o** 910880371. **p** 910880371. **q** 924674300. **r** 924674300. **s** 98005782. **t** 98005782

2.3.3 Conclusion

The results in this section are based on preliminary work. Some users are more likely to express their sentiments using image tweets, while others are still more likely to express their sentiment using text tweets. This reveals the challenges in predicting the sentiment of online social network users. The results in this section are encouraging for using the multimedia information for sentiment analysis.

Nevertheless, sentiment analysis is quite challenging for social multimedia. The short text nature of tweets imposes more challenges on this task. The results in this study indicate that both the textual and visual features are informative in determining one's sentiment. We discover the correlation between the sentiment expressed by text tweets and image tweets. At the same time, different users also reveal different behavior patterns in online social networks. Although the results do indicate some correlation between image tweets and textual tweets, to get more robust and more interpretable results, we need more features and more robust data to discover the

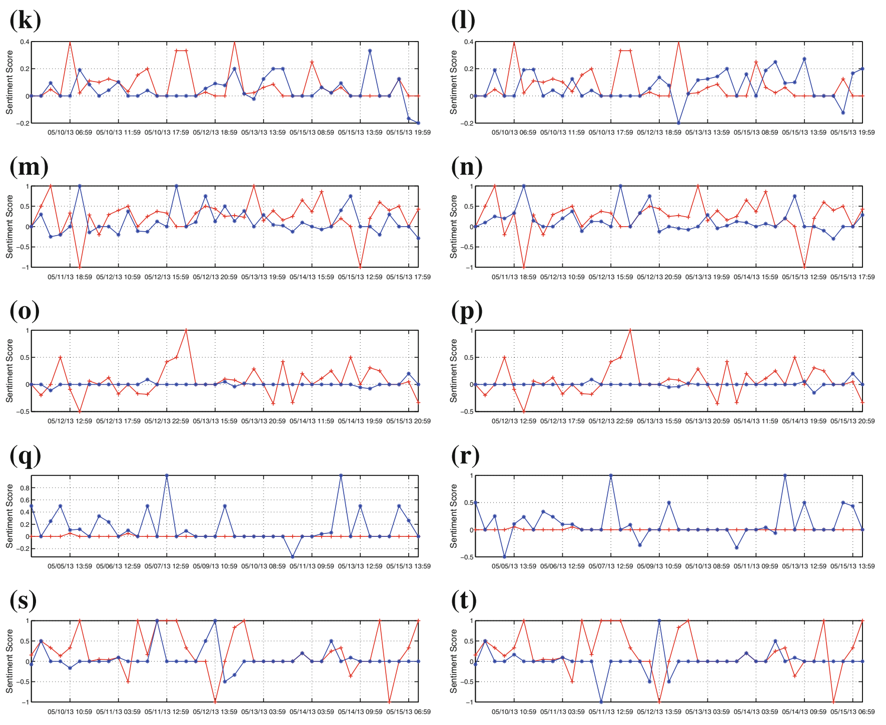


Fig. 2.14 (continued)

Table 2.6 Correlation coefficients of textual sentiment and visual sentiment (NA means not available)

User id	Low-level features	Mid-level features
0110914277	0.132197	0.137298
1135866961	0.059131	0.108657
0183352499	0.038009	0.095219
0320657019	0.105444	0.084368
0341587111	NA	NA
0606333611	0.618337	0.322811
0745235832	NA	NA
0910880371	0.199853	0.023016
0924674300	0.297284	0.317496
0098005782	0.015088	0.166366

influence of multimedia content in the social network. The sentiment analyses of images are still not mature. This, on the other hand, indicates that we have a great opportunity for discovery in this area.

Table 2.7 Correlation coefficients of textual sentiment and visual sentiment for recent 40 periods

User id	Low-level features	Mid-level features
0110914277	0.176150	0.132065
1135866961	0.172788	0.172788
0183352499	0.075004	−0.197358
0320657019	0.226449	0.212064
0341587111	0.150699	0.221518
0606333611	0.398337	0.065079
0745235832	0.089547	0.006048
0910880371	−0.071518	−0.244712
0924674300	0.245525	0.252585
0098005782	−0.127538	−0.027864

2.4 Discussion and Future Work

In this chapter, we have discussed some of the current works in the field of sentiment analysis and presented our new research results on image and multimedia sentiment analysis. We are living in an increasingly open society and individuals are now more and more willing to share feeling with others and listen to others' opinions at the same time. Due to the enormous growth in social network platforms, sentiment analysis is receiving more attention. Although we now have more data sources at greater scales than ever before, sentiment analysis based on visual and multimodality perspective is still in its infancy. In the computer vision field, the development of attribute learning and deep neural network structures have shown some promising results, which can lead to sentiment analysis approaches such as *Sentribute*. Additionally, from a multimodality perspective, topics on deep multimodal structures are drawing more attention these days. For example, Srivastava showed in [63] that multimodal learning with deep boltzmann machines can improve the classification performance from the joint features extracted from both text and images. These techniques are expected to bring a new chapter to sentiment analysis and opinion mining.

Acknowledgments We would like to thank Prof. Shih-Fu Chang and his group for providing the test dataset and helpful discussions. This work is supported in part by the University of Rochester, and a generous gift from Adobe Systems.

References

1. Liu B (2012) Sentiment analysis and opinion mining. Synth Lect Hum Lang Technol 5(1):1–167
2. Nasukawa T Yi J (2003) Sentiment analysis: capturing favorability using natural language processing. In: Proceedings of the 2nd international conference on knowledge capture. ACM, pp 70–77

3. You Q, Luo J (2013) Towards social imagematics: sentiment analysis in social multimedia. In: Proceedings of the thirteenth international workshop on multimedia data mining, MDMKDD'13. ACM, New York, pp 3:1–3:8
4. O'onnor B, Balasubramanyan R, Routledge BR, Smith NA (2010) From tweets to polls: linking text sentiment to public opinion time series. In: Proceedings of the international AAAI conference on weblogs and social media, pp 122–129
5. Pang B, Lee L (2008) Opinion mining and sentiment analysis. Found Trends Inf Retr 2(1–2):1–135
6. Wilson T, Wiebe J, Hoffmann P (2005) Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, pp 347–354
7. Esuli A, Sebastiani F (2006) Sentiwordnet: a publicly available lexical resource for opinion mining. In: Proceedings of LREC, vol 6, pp 417–422
8. Thelwall M, Buckley K, Paltoglou G, Cai D, Kappas A (2010) Sentiment strength detection in short informal text. J Am Soc Inf Sci Technol 61(12):2544–2558
9. Farhadi A, Endres I, Hoiem D, Forsyth D (2009) Describing objects by their attributes. In: IEEE conference on computer vision and pattern recognition (CVPR 2009), pp 1778–1785, IEEE
10. Naphade MR, Lin C-Y, Smith JR, Tseng B, Basu S (2002) Learning to annotate video databases. In: SPIE conference on storage and retrieval on media databases
11. Ordonez V, Kulkarni G, Berg TL (2011) Im2text: describing images using 1 million captioned photographs. In: Neural information processing systems (NIPS)
12. Snoek CG, Worring M (2008) Concept-based video retrieval. Found Trends Inf Retr 2(4):215–322
13. Datta R, Joshi D, Li J, Wang JZ (2006) Studying aesthetics in photographic images using a computational approach. In: Computer vision-ECCV 2006. Springer, pp 288–301
14. Marchesotti L, Perronnin F, Larlus D, Csurka G (2011) Assessing the aesthetic quality of photographs using generic image descriptors. In: 2011 IEEE international conference on computer vision (ICCV), pp 1784–1791
15. Isola P, Xiao J, Torralba A, Oliva A (2011) What makes an image memorable? In: 2011 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, pp 145–152
16. Jia J, Wu S, Wang X, Hu P, Cai L, Tang J (2012) Can we understand van Gogh's mood?: learning to infer affects from images in social networks. In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 857–860
17. Machajdik J Hanbury A (2010) Affective image classification using features inspired by psychology and art theory. In: Proceedings of the international conference on multimedia. ACM, pp 83–92
18. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. Int j comput vis 42(3):145–175
19. Yanulevskaya V, Uijlings J, Bruni E, Sartori A, Zamboni E, Bacci F, Melcher D, Sebe N (2012) In the eye of the beholder: employing statistical analysis and eye tracking for analyzing abstract paintings. In: Proceedings of the 20th ACM international conference on multimedia, MM'12. ACM, New York, pp 349–358
20. Bartlett MS, Littlewort G, Frank M, Lainscsek C, Fasel I, Movellan J (2005) Recognizing facial expression: machine learning and application to spontaneous behavior. In: IEEE computer society conference on Computer vision and pattern recognition, CVPR 2005, vol 2. IEEE, pp 568–573
21. Fasel B, Luettin J (2003) Automatic facial expression analysis: a survey. Pattern Recognit 36(1):259–275
22. Wan S, Aggarwal J (2014) Spontaneous facial expression recognition: a robust metric learning approach. Pattern Recognit. 47(5):1859–1868
23. Hernandez J, Hoque ME, Drevo W, Picard RW (2012) Mood meter: counting smiles in the wild. In: Proceedings of the 2012 ACM conference on ubiquitous computing. ACM, pp 301–310

24. Hoque ME, Courgeon M, Martin J-C, Mutlu B, Picard RW (2013) Mach: my automated conversation coach. In: Proceedings of the 2013 ACM international joint conference on pervasive and ubiquitous computing. ACM, pp 697–706
25. Guerra PHC, Veloso A, Meira W Jr, Almeida V (2011) From bias to opinion: a transfer-learning approach to real-time sentiment analysis. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 150–158
26. Speriosu M, Sudan N, Upadhyay S, Baldridge J (2011) Twitter polarity classification with label propagation over lexical links and the follower graph. In: Proceedings of the first workshop on unsupervised learning in NLP. Association for Computational Linguistics, pp 53–63
27. De Choudhury M, Counts S, Horvitz E (2013) Major life changes and behavioral markers in social media: case of childbirth. In: Proceedings of the 2013 conference on computer supported cooperative work. ACM, pp 1431–1442
28. Silva IS, Gomide J, Veloso A, Meira W Jr, Ferreira R (2011) Effective sentiment stream analysis with self-augmenting training and demand-driven projection. In: Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval. ACM, pp 475–484
29. Hu X, Tang L, Tang J, Liu H (2013) Exploiting social relations for sentiment analysis in microblogging. In: Proceedings of the sixth ACM international conference on web search and data mining. ACM, pp 537–546
30. Kosinski M, Stillwell D, Graepel T (2013) Private traits and attributes are predictable from digital records of human behavior. In: Proceedings of the national academy of sciences
31. Rao D, Paul M, Fink C, Yarowsky D, Oates T, Coppersmith G (2011) Hierarchical Bayesian models for latent attribute detection in social media. In: Proceedings of the ICWSM, pp 598–601
32. Goel S, Hofman JM, Sirer MI (2012) Who does what on the web: a large-scale study of browsing behavior. In: Proceedings of the 6th international AAAI conference on weblogs and social media
33. Wong FMF, Tan CW, Sen S, Chiang M (2013) Quantifying political leaning from tweets and retweets
34. De Choudhury M, Counts S, Gamon M (2012) Not all moods are created equal! Exploring human emotional states in social media. In: Sixth international AAAI conference on weblogs and social media
35. Miller M, Sathi C, Wiesenthal D, Leskovec J, Potts C (2011) Sentiment flow through hyperlink networks. In: Proceedings of the AAAI CWSM
36. Tan C, Lee L, Tang J, Jiang L, Zhou M, Li P (2011) User-level sentiment analysis incorporating social networks. arXiv preprint arXiv:1109.6018
37. Jansen BJ, Zhang M, Sobel K, Chowdury A (2009) Twitter power: tweets as electronic word of mouth. J Am soc inf sci technol 60(11):2169–2188
38. Bollen J, Mao H, Zeng X (2011) Twitter mood predicts the stock market. J Comput Sci 2(1):1–8
39. Tumasjan A, Sprenger TO, Sandner PG, Welpe IM (2010) Predicting elections with twitter: what 140 characters reveal about political sentiment. In: Proceedings of the fourth international AAAI conference on weblogs and social media, pp 178–185
40. Yang CC, Ng TD (2007) Terrorism and crime related weblog social network: link, content analysis and information visualization. In: Intelligence and security informatics, New Brunswick, pp 55–58
41. Datta R, Joshi D, Li J, Wang JZ (2008) Image retrieval: ideas, influences, and trends of the new age. ACM Comput Surv (CSUR) 40(2):5
42. Zhang S, Yang M, Cour T, Yu K, Metaxas DN (2012) Query specific fusion for image retrieval. In: Computer vision-ECCV 2012. Springer, pp 660–673
43. Wang W, He Q (2008) A survey on emotional semantic image retrieval. In: 15th IEEE international conference on image processing (ICIP 2008), pp 117–120
44. Lang PJ, Bradley MM, Cuthbert BN (1999) International affective picture system (Iaps): Technical manual and affective ratings

45. Li B, Feng S, Xiong W, Hu W (2012) Scaring or pleasing: exploit emotional impact of an image. In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 1365–1366
46. Vonikakis V, Winkler S (2012) Emotion-based sequence of family photos. In: Proceedings of the 20th ACM international conference on multimedia, MM'12. ACM, New York, pp 1371–1372
47. Hanjalic A, Kofler C, Larson M (2012) Intent and its discontents: the user at the wheel of the online video search engine. In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 1239–1248
48. Borth D, Chen T, Ji R, Chang S-F (2013) Sentibank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content. In: Proceedings of the 21st ACM international conference on multimedia. ACM, pp 459–460
49. Yuan J, Mcdonough S, You Q, Luo J (2013) Sentribute: image sentiment analysis from a mid-level perspective. In: Proceedings of the second international workshop on issues of sentiment discovery and opinion mining. ACM, p 10
50. Cambria E, Hussain A (2012) Sentic album: content-, concept-, and context-based online personal photo management system. Cogn Comput 4(4):477–496
51. Patterson G Hays J (2012) Sun attribute database: discovering, annotating, and recognizing scene attributes. In: 2012 IEEE conference on computer vision and pattern recognition (CVPR), pp 2751–2758
52. Lundqvist D, Flykt A, Öhman A (1998) The karolinska directed emotional faces-KDEF. CD-ROM from department of clinical neuroscience, psychology section, Karolinska Institutet, Stockholm, Sweden. Technical report, ISBN 91-630-7164-9
53. Xiao J, Hays J, Ehinger KA, Oliva A, Torralba A (2010) Sun database: large-scale scene recognition from abbey to zoo. In: 2010 IEEE conference on computer vision and pattern recognition (CVPR), pp 3485–3492
54. Tao D, Tang X, Li X, Wu X (2006) Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. Pattern Anal Mach Intell IEEE Trans 28(7):1088–1099
55. Turk MA Pentland AP (1991) Face recognition using eigenfaces. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, CVPR'91. IEEE, pp 586–591
56. Ekman P (1992) An argument for basic emotions. Cogn Emot 6(3–4):169–200
57. Gilbert E, Karahalios K (2010) Widespread worry and the stock market. In: Proceedings of the international conference on weblogs and social media, vol 2, pp 229–247
58. Go A, Bhayani R, Huang L (2009) Twitter sentiment classification using distant supervision. Technical report, Stanford, pp 1–12
59. Siersdorfer S, Minack E, Deng F, Hare J (2010) Analyzing and predicting sentiment of images on the social web. In: Proceedings of the international conference on multimedia. ACM, pp 715–718
60. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE computer society conference on computer vision and pattern recognition (CVPR 2005), vol 1. IEEE, pp 886–893
61. Shechtman E Irani M (2007) Matching local self-similarities across images and videos. In: IEEE conference on computer vision and pattern recognition (CVPR'07), pp 1–8
62. Lalonde J-F, Hoiem D, Efros AA, Rother C, Winn J, Criminisi A (2007) Photo clip art. In: ACM transactions on graphics (TOG), vol 26. ACM, p 3
63. Srivastava N, Salakhutdinov R (2012) Multimodal learning with deep Boltzmann machines. In: NIPS, pp 2231–2239

Chapter 3
Twitter as a Personalizable Information Service

Mario Cataldi, Luigi Di Caro and Claudio Schifanella

Abstract Twitter is a free social networking microblogging service that allows registered members to broadcast, in real-time, short posts called tweets. Twitter members can broadcast tweets and follow other users' tweets by using multiple devices, making this information system one of the fastest in the world. In this chapter, we leverage this characteristic to introduce a novel topic-detection method aimed at informing, in real-time, a specific user about the most emerging arguments expressed by the network around his/her domain interests. With this goal, we aim at formalizing the information spread over the network by studying the topology of the network and by modeling the implicit and explicit connections among the users. Then, we propose an innovative term aging model, based on a biological metaphor, to retrieve the freshest arguments of discussion, represented through a minimal set of terms, expressed by the community within the foci of interest of a specific user. We finally test the proposed model through various experiments and user studies.

3.1 Introduction

Microblogging today has become a very popular communication system among users. In fact, due to the short format of messages and the accessibility of microblogging platforms, users tend to shift from traditional communication tools (such as blogs, websites, and mailing lists) to microblogging services. Billions of messages

M. Cataldi (✉)
LIASD, Department of Computer Science,
Université Paris 8, Paris, France
e-mail: m.cataldi@iut.univ-paris8.fr

L. Di Caro · C. Schifanella
Department of Computer Science, University of Turin,
Corso Svizzera 185, Turin, Italy
e-mail: dicaro@di.unito.it

C. Schifanella
e-mail: schi@di.unito.it

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_3

appear daily in these services such as Twitter,[1] Tumblr,[2] Facebook,[3] etc. The authors of those messages share content about their private life, exchange opinions on a variety of topics, and discuss a wide range of information news.

Among all the existing platforms, after its launch on July 2006, Twitter became the most popular *microblogging system*; on February 2012, its number of users was estimated to be about 500 million worldwide with around a half million of new accounts per day, which makes Twitter one of the fastest-growing websites in the world. Moreover, in contrast to other popular social networks such as Facebook or Google+, most of its users are adults; in fact, according to a demographic report,[4] 88 % of the users are older than 18, which makes the service likely oriented to information than other social aspects.[5] In fact, as information producers, people post tweets (text messages up to 140 characters) for a variety of purposes, like including daily conversations, share of information/URLs, news reports, etc. This produces a continuous real-time information stream about every argument.

Even if Twitter cannot represent an alternative to the authoritative information media, considering the number of its users and the impressive response time of their contributions, it represents a sort of real-time news sensor that can also predate the best newspapers in informing the Web community about the emerging topics and trends. In fact, the most important information media always needs a certain amount of time to react to a news event; i.e., professional journalists require time, collaborators, and/or technology support to provide a professional report. However, within Twitter, a user can easily report, in 140 characters, what is happening in front of the user's eyes, without any concern about the readers or the writing style. This characteristic makes Twitter probably one of the fastest, low-level, information service in the world.

In this chapter, this informative role of Twitter is recognized, presenting an extension of the approach proposed in [2] to extract, in real-time, the most emerging topics expressed by the community along the interests of a specific user. Considering an active user, the interests are analyzed by extracting and formalizing the content of her generated tweets. Then, the social community is modeled as a directed graph of the active authors based on their social relationships, calculating their authority by relying on the well-known PageRank algorithm [3]. The stream of information expressed by the entire network is monitored by studying the life cycle of each term according to an aging model that also leverages the reputation of each author. The set of most emerging keywords is selected by dynamically ranking the terms depending on their life status defined through a burstiness value. Finally, each topic (expressed as minimal set of terms, as in [4, 5]) is created by constructing and analyzing a keyword graph which links the extracted emerging terms with all their co-occurrent

[1] http://www.twitter.com.

[2] http://www.tumblr.com.

[3] http://www.facebook.com.

[4] http://palatnikfactor.com/2010/01/29/twitter-demographic-report-who-is-really-on-twitter/.

[5] A deeper analysis of the Twitter network is also provided in [1].

keywords. At this point, in order to personalize the list of retrieved emerging topics, the temporal time frames in which the user has been active are analyzed, as well as the generated content to estimate the user's interests according to this temporal information. This time-aware information is finally used to highlight the topics that best match the interests of the user.

The chapter is organized as follows: Sect. 3.2 presents an overview of the current state of the art on content aggregation, recommendation, trend analysis, social monitoring, and content personalization by reporting a short summary of the existing approaches. The proposed personalized topic detection is presented with a formalization of the assumptions and providing real case scenarios (Sect. 3.3). Section 3.4 shows a set of experiments and user studies that demonstrate the validity of the approach. Section 3.5 concludes with future issues.

3.2 Related Work

In the last decade, the enormous amount of content generated by Web users created new challenges and new research questions within the data mining community. In this section, we present an overview of those works that share part of our techniques, ideas, and motivations. In particular, we initially report related work on aggregation, recommendation, and propagation of information from large-scale social networks. We then survey the related work on automatic detection of events within user-generated environments and, finally, we analyze the current state of the art on personalization and user context analysis.

3.2.1 Aggregation, Propagation, and Recommendation Through Social Networks

A first issue when dealing with large and heterogeneous data sources is the aggregation of the content through filtering and merging techniques. Considering Twitter as a source of text data, many Web services like *TweetTabs*[6] and *Where What When*[7] aggregate messages and links through user-friendly interfaces. In general, clustering techniques help in finding groups of similar content that can be further filtered using labeling techniques [6].

While these services simply aggregate messages and/or links, one of the most explored tasks in mining of text entries streams from social media is the recommendation of topics, URLs, friends, and so forth. So far, two main high-level approaches have been studied: collaborative filtering and content-based techniques. While the

[6] http://tweettabs.com.

[7] http://where-what-when.husk.org.

first aims at selecting and proposing content by looking at what similar users have already selected (as in [7]), the second analyzes the semantics of the content without considering its origin (as in [8]). More recently, hybrid approaches have been also proposed in [9, 10]. Recommendation systems can differ on what they recommend: URLs (as in [11]), users that share same interests (as in [12]) and tags in folksonomy-based systems (like the approach proposed in [13]).

Another issue when dealing with large, time-sensitive, and user-generated text content is the analysis of how such information spreads through the blogosphere or a social network. Generally speaking, research on flows of information in networks initially started from the analogy with the spread of a disease in a social environment. This model is based on the following disease life cycle: someone is first susceptible to the disease and then, if exposed to the disease by an infectious contact, he becomes infected (and infectious) with some probability. In general, there exist two main approaches to model propagation, namely threshold models [14] and cascade models [15]. While the first treats the problem as a chain-reaction of influence where each node in the network obeys to a monotone activation function, in the second, nodes close to each other have some chance to influence each other as well.

In [16], the authors studied the dynamics of Web environments at topic- and user-level, inducing propagation networks from a sequence of posts in the blogosphere. In [17] the authors studied how to learn influence probabilities from a log of past propagations in the *Flickr*[8] community. In [18] the authors developed the Linear Influence model, where the influence functions of individual nodes govern the overall rate of diffusion through the network. Reference [19] presents a comparison between three measures of influence: indegree, retweets, and mentions in Twitter, investigating the dynamics of users' influence across topics and time. In [20] the authors proposed a model to capture properties of information diffusion like speed, scale, and range.

Focusing on Twitter-based approaches, *Trendistic*[9] and *Twopular*[10] represent two examples from which it is possible to analyze the trends of some keywords along a timeline specified by the user. In general, several studies examined topics and their changes across time in dynamic text corpora. The general approach orders and clusters the documents according to the timestamps, analyzing the relative distributions (see also [21]). Reference [22] represents social text streams as multi-graphs, where each node represents a social actor and each edge represents the information flow between two actors. In [23, 24], the authors present a system that uses curve analysis of frequencies for automatic segmentation of topics.

In [25] the tolerance rough set model is used to enrich the set of feature words into an approximated latent semantic space, from which they extract hot topics by a complete-link clustering. Reference [26] analyzes tweets in order to predict whether the user is looking for news or not, and determine keywords that can be added to her Web search query. Reference [27] explores the longevity of trending topics on

[8] http://www.flickr.com.

[9] http://trendistic.com.

[10] http://twopular.com.

Twitter, and analyzes the role of users in the emergence of trends. The role of the users, their collaboration, and the influence among them has been also extensively studied in social networks [28], from qualitative studies on cooperation behaviors [29–31] to more quantitative approaches [32, 33]. The latter includes collaboration network-based studies (because of a social network can be easily seen as a social network where people form teams to produce some results), which are generally aimed at understanding the structural determinants and patterns of collaboration [34–37]. Such networks have been deeply analyzed by looking at properties like network topology, size, and evolution [38, 39].

In [40] the authors were the first to present an aging theory based on a biological metaphor. Using this approach, the work presented in [41] is able to rank topics from online news streams through the concept of *burstiness*. The burstiness of a term, already introduced in [42], is computed with a χ-statistic on its temporal contingency table. Although this work shares our goal, it is based on the concepts of user attention and media focus, whereas our proposed approach is independent of them and is assumed to be less complex and more general.

3.2.2 Event Identification in Social Networks

Identifying events in real-time on Twitter is a challenging problem, due to the heterogeneity and immense scale of the data. In fact, as already reported in the Introduction, users post messages with a variety of purposes. For this, while many contents are not specifically related to any particular real-world event, informative event messages nevertheless abound.

Regarding the classification of single tweets, [43] defines a typology of five generic classes of tweets (news, events, opinions, deals, and private messages) in order to improve information filtering and recognize events. The research works proposed in [44–46] focused on identifying events in social media in general, and on Twitter. Recent works on this social network started to process data as a stream with the goal of identifying events of a particular type (e.g., news events , earthquakes, etc.). In [47]) the authors identify the first Twitter message associated with an event in order to analyze the starting point of the related information flow.

The real-time social content can also be seen as a sensor that captures what is happening in the world: similarly to the recommendation task, this can be exploited for a zero-delay information broadcasting system that detects emerging concepts. Generally, all the techniques rely on some measure of importance of the keywords. [48] presents the $TF * PDF$ algorithm which extends the well-known *TF—IDF* to avoid the collapse of important terms when they appear in many text documents. Indeed, the *IDF* component decreases the frequency value for a keyword when it is frequently used. Considering different newswire sources or channels, the weight of a term from a single channel is linearly proportional to the term's frequency within it, while it is exponentially proportional to the ratio of documents that contain the term in the channel itself. In [49] a supervised learning system for the extraction of events from unstructured textual information that relies on geotagged Twitter posts is presented. Reference [50] considers Twitter as a social sensor for detecting large-scale events like earthquakes, typhoons, and traffic jams. The authors analyze the context of such keyword in order to discriminate them as positive or negative (the sentence "Someone is shaking hands with my boss" should be captured as negative even though it contains the term "shake").

In [51], the authors present a system called OLDA (Online Topic Model) which permits to automatically capture the thematic patterns and identifies emerging topics of text streams and their changes over time. However, in contrast with our approach, the system detects topics of discussion without any knowledge about user properties and/or preferences.

3.2.3 Content Personalization

Since our system includes a module for the personalization of the emerging topics to be retrieved from Twitter, we also cover here the relevant works in this field. Most of the literature refers to this task as "personalization of search results," or "user-driven personalization" (as in the works proposed in [52–55]). As stated in [56], the motivation behind the interest around this area of research is based on the reasonable assumption that different users generally expect different information even with the same query. There obviously exist several approaches to facing such a task. For instance, depending on the domain, one may be interested in *re-ranking* the results based on their relevance [53, 55], rather than *diversify* them (as in [57–60]). Yet, the actual personalization of contents (whether they are search results or emerging topics as in our case) could be made by leveraging different kinds of data: users' contents [61, 62], ontologies [63–65], and users's social network and activity [66, 67]. Given this brief overview, our system can be classified as a re-ranking approach that makes use of the users' contents, i.e., the tweets posted by them. In addition to this simple scheme, we also wanted to take into account the temporal aspect associated to the tweets in order to weight more what has been recently posted, and the other way around.

3.3 Searching Emerging Topics Along the Users' Interests

This section illustrates the method for analyzing, in real-time, the dynamic stream of information expressed by the Twitter community and retrieve the most emerging topics within the user's interests. First, the set of tweets, generated within a specific time interval, is represented as a set of keyword vectors, then a term aging model is explained to monitor the usage of each keyword over time. Moreover, the social reputation of the Twitter users is leveraged to balance the importance of the information expressed by the community. Finally, the user context is taken into account, provided by the generated set of tweets, to highlight the most emerging topics within her interests. In the following sections, these steps are explained in detail.

3.3.1 Real-Time Vectorization of Tweets

As in most information retrieval (IR) systems, the first step in every task is the extraction of the relevant keywords (also called terms in the chapter) from the stream of tweets. Thus, given a time range r set by the system (depending on the preferred topic-detection frequency, see also Sect. 3.3.2.2), the tth considered interval I^t is defined as

$$I^t = < i_t, i_t + r > \tag{3.1}$$

where i_t is the starting instant of the tth considered time interval (and $i_0 = 0$ represents the first considered instant). Thus, the corpus TW^t is extracted, with $n = \left|TW^t\right|$ text tweets extracted during the time interval I^t, associating to each tweet tw_j a representative *tweet vector*, $\mathbf{tw_j}$, that formalizes the information retrieved from it.

Each component of the vector $\mathbf{tw_j}$ represents a weighted term extracted from the related tweet tw_j. In contrast with common systems, no preliminary phase of stop-word elimination is performed; in fact, the system considers all the languages in which Twitter's users update their status. The idea is to leverage the Twitter user's network, using their worldwide extension, in order to be able to retrieve in real-time relevant news. In fact, the stream of information directly rises in the geographical origin of the event and expands its influence proportionally to its global importance; for example, the first news reports about the revolutionary wave of demonstrations and protests occurring in the Arab world on 2011, also known as "Arab Spring,"[11] had been initially generated in Tunisia and Egypt and then, due to the global political and social importance of these events, they have been also commented by users of different countries and continents. Thus, in order to be able to quickly catch a relevant news from this worldwide information network, there is no need to discriminate the information based on the language or the country in which it has been generated. Obviously, this approach has the significant disadvantage of maintaining all the keywords, including stop-words, typos, and irrelevant terms; however, it is possible to recognize this noise by adapting text analysis methods that consider inverse frequency-like techniques. This information refinement step is applied in Sect. 3.3.2.

Considering this idea, all the keywords are preserved together with those keywords that appear less frequently but that could be highly relevant for a specific topic. Thus, the system calculates the weight $w_{x,j}$ of the xth vocabulary term in jth tweet by using the augmented normalized term frequency [68]:

[11] These protests have also been knows for the massive use of Twitter post because of the protesters' reliance on Twitter and other social-networking Internet sites to communicate with each other.

$$w_{x,j} = 0.5 + 0.5 \cdot \frac{tf_{x,j}}{tf_j^{max}} \tag{3.2}$$

where $tf_{x,j}$ is the term frequency value of the xth vocabulary term in jth tweet and tf_j^{max} returns the highest term frequency value of the jth tweet.

Thus, for each tweet tw_j, a tweet vector

$$\mathbf{tw_j} = \{w_{1,j}, w_{2,j}, ..., w_{v,j}\} \tag{3.3}$$

is defined, where K^t is the vocabulary (set of keywords) of the corpus (and thus of the tweet tw_j) in the time interval I^t and $v = |K^t|$ is its size. Notice that, at this step, we do not make use of any word filtering technique. In fact, considering our time-sensitive model, we believe that stop-words and/irrelevant keywords will always have constant use along time and will be easily detectable (and discardable) by performing a simple temporal analysis. In contrast, a keyword filtering method would always depend on a specific stop-word list, which could be dependent on a specific domain of interest. We will explain this idea in detail in the next sections.

At the end of this step, the knowledge expressed by each collected tweet in the considered time interval has been formalized as a weighted tweet vector.

3.3.2 Analyzing the Stream of Information by Taking into Account Temporal Conditions

This section illustrates the analysis of the extracted tweets stream in order to study the semantic relationships that exist among the keywords reported by the community in a given time interval.

Generally speaking, a term can be viewed as a semantic unit which can potentially link to a news event. The goal of capturing such correlation relies on an accurate modeling of both the chronological sequences of tweets and the authors. Following this intuition, the system uses a content aging theory to automatically identify coherent discussions through a life cycle-based content model.

Many conventional clustering and classification strategies cannot be applied to this problem due to the fact that they tend to ignore the temporal relationships among documents (tweets in our case) related to a news event. Relying on this temporal feature, the system uses a metaphor where each term is seen as a living organism. The life cycle of a keyword can be considered as analogous to the one of a living being: with abundant nourishment (i.e., related tweets), its life cycle is prolonged; however, a keyword or a live form dies when nourishment becomes insufficient.

Relying on this analogy, it is possible to evaluate the usage of a keyword by its *burstiness*, which indicates the *vitality* status of the keyword and can qualify the keyword's usage. In fact, a high burstiness value implies that the term is becoming important in the considered community, while a low burstiness value implies that it is currently becoming out of favor. Considering this biological metaphor, the contribution in terms of nutrition of each nourishment changes depending on its chemical composition; for example, each food brings a different calorie contribution depending on its ingredients. Therefore, the system uses the concept of authority to define the *quality* of the nutrition that each tweet gives to every contained keyword. This way, different tweets containing the same keyword generate different amount of nutrition depending on the representativeness of the author in the considered community.

Thus, considering a keyword $k \in K^t$ and the set of tweets $TW_k^t \in TW^t$ containing the term k at time interval I^t, the amount of nutrition is defined as

$$\text{nutr}_k^t = \sum_{tw_j \in TW_k^t} w_{k,j} * rep(user(tw_j)) \tag{3.4}$$

where $w_{k,j}$ represents the weight of the term k in the tweet vector $\mathbf{tw_j}$ (thus, $tw_j[k]$), the function $user(tw_j)$ returns the author u of the tweet tw_j and $rep(u)$ returns the reputation value associated to u by using some reputation evaluation system. In this chapter the user reputation is computed by leveraging the connectivity of the author graph based on the *follower* relationship (see Sect. 3.3.2.1).

Thus, considering a keyword k used by the community in the time interval I^t, this nutrition formula evaluates the usage of this term by considering its frequency in the tweets that mention it as well as the reputation of each single user that reported k. Please notice that the authoritativeness of the users who first talk about an emerging topic could be not necessary high. However, considering the dynamicity of the Twitter network, the information spreads very quickly and reaches in a limited time window (if the information is significant, in some sense, for the active users), a very large number of authors with different authority values. Thus, the authority of the users is not the only essential parameter. If a news is massively commented by only not authoritative users, the system will yet be able to detect the emerging keywords and retrieve the related topics. In other words, a keyword is labeled as emergent based on the fastness of its spread over the network. In fact, as reported in [27], "factors such as user activity and number of followers do not contribute strongly to trend creation and its propagation." The authority of the users only helps to further highlight this fact.

3.3.2.1 Reputation of the Users

While the contents themselves constitute the entire semantics from where to extract emerging facts, a fundamental issue in the treatment of such knowledge is the importance of the source. Figuring out a level of importance of a specific source (i.e., a Twitter user) represents a key point toward a well-advised filtering and weighting of the contents.

A Twitter user can follow the text stream of other users by making explicit the social relationship of *follower*. On the other hand, a user who is being followed by another user does not necessarily have to reciprocate the relationship by following her back, which makes the graph of the network directed. This social model enables us to define an author-based graph $G(U, F)$ where U is the set of users and F is the set of directed edges (i.e., the follower relation); thus, given two users u_i and u_j, the edge $\langle u_i, u_j \rangle$ exists only if u_i is a follower of u_j.

Thus, the reputation of each user is measured by analyzing the connectivity in G; In particular, since users tend to follow people they think are interesting (for example because they share the same topics of interest), we can assume that a user with a high number of followers (incoming edges) represents an influential information source into this social community. For example, most of the people easily agree that Barack Obama (with more than 10 million followers) represents a strongly authoritative twitter user, simply because each of his words can be instantly read by thousands of other users, and can influence their normal text stream activity. Moreover, the concept of "reputation" can be also extended by taking into account the fact that the importance of a user is also related to the degree of importance of its followers; considering for example Barack Obama again, each of the users followed by him assumes more importance based on the influence of this authoritative relationship. For all these considerations, this scenario can be easily compared to the problem of topological-based computation of web pages authority in large hyper textual systems. In particular, as in alternative works [69–71], we can refer to the well-known PageRank algorithm [3] for this task. PageRank calculates the authority of each page by analyzing the topological graph of the considered Web entities. Following this strategy, the reputation of a user depends on the number and the reputation of its followers. Hence, given a user $u_i \in U$, its *reputation* is computed as follow:

$$rep(u_i) = d \times \sum_{u_j \in follower(u_i)} \frac{rep(u_j)}{|following(u_j)|} + (1 - d) \tag{3.5}$$

where $d \in (0, 1)$ is a dumping factor,[12] $follower(u_i)$ is a function that returns the set of users following u_i and $following(u_j)$ returns the set of users that u_j follows.

In Fig. 3.1, an example of user reputation computation on the input graph, obtained by performing a graph sampling process [72] in which the "Barack Obama" vertex represents the starting point, is depicted. User reputation values are visually represented by the circle sizes. In this case, "Barack Obama" is the most influential user, since it has the highest number of followers (more than 10 millions). Moreover, its reputation is propagated to the "davidaxelrod" user—the twitter account of David Axelrod, an American political consultant—because of the follower relation by "Barack Obama;" this scenario confers to "davidaxelrod" a high reputation value, even if it has a significantly lower number of followers ($\sim$60 k followers).

[12] The dumping factor d, introduced by the authors in [3], represents the probability that a "random surfer" of the graph G moves from a user to another; it is usually set to 0.85.

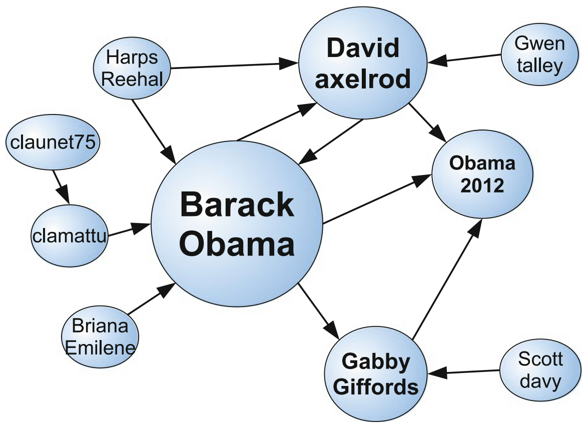


Fig. 3.1 The reputation value computation: a sample of the "Barack Obama" community. The size of the nodes represents their importance in the considered community

3.3.2.2 Computing Term Burstiness Values

Once the nutrition of a term is calculated, the aim is to map it into a value of *burstiness*. The burstiness value of a term indicates its actual contribution (i.e., how much it is emergent) in the corpus of tweets. Our idea is that the temporal information associated to the tweets can be used as discriminant function in that sense. The term *burstiness* used in this chapter closely resembles to the concept of *burstiness* introduced in many related works as [40, 73, 74]. In detail, the burstiness of a term is defined as the ratio of the frequency of the number of occurrences of a keyword on the current time period with respect to its occurrences on previous time slots. Please notice that the term burstiness has been used also w.r.t. many different factors (occurrences in different documents, sentences, and/or position in a text); in this chapter we refer to this concept only w.r.t. temporal conditions.

In our work, having for each keyword k its amount of nutrition nutr_k^t in a time interval I^t, it is possible to rank the hottest terms only considering their related nutrition value. A term can be defined as *hot* if its usage is extensive within the considered time interval.

However, as explained in the Introduction, in this step we are interested in detecting the *emerging* terms during the considered time interval I^t. For this, we need to introduce a temporal evaluation of each keyword usage to analyze this property. For this, a keyword is defined as *emergent* if it results to be *hot* in the considered time interval but not in the previous ones. In other words, we analyze the keyword life cycles by comparing their nutrition values obtained on the considered time frame with the usage of the same terms in the past time intervals. Namely, the current nourishment is analyzed in comparison to the ones built in the previous time intervals.

Let consider the examples shown in Figs. 3.2 and 3.3; considering the time frame from August 2011 to February 2012, the term "car" is, obviously, significantly more used by the twitter community than the term "concordia." Thus, according to our definition, the term "car" can be considered *hotter* than "Concordia." On the other

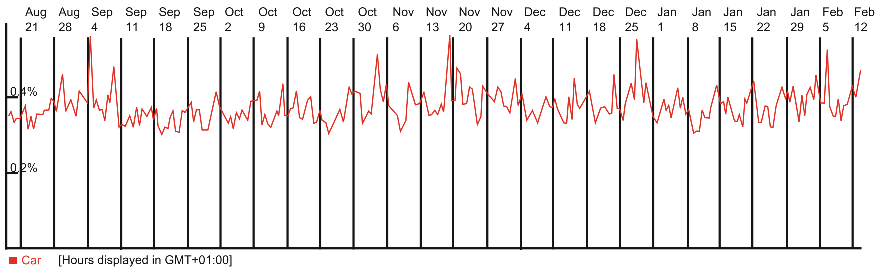


Fig. 3.2 Statistical usage of the term "car" (provided by Trendistic) in Twitter from August 2011 to February 2012

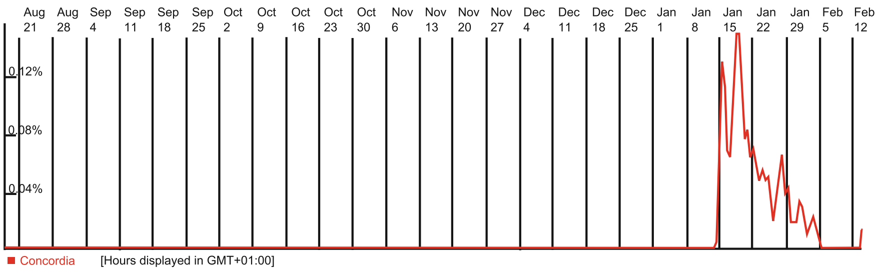


Fig. 3.3 Statistical usage of the term "Concordia" (provided by Trendistic) in Twitter from August 2011 to February 2012; the peak represents the catastrophic sinking of the Italian boat "Costa Concordia" occurred in Italy on January 13, 2012

hand, considering the specific time instant defined by the day January 13, 2012, the term "concordia" can be easily seen as *emerging* keyword, while there is no significant change in the usage of term "car" (because of the extensive use of the term which can be correlated to the sinking of the cruise ship "Costa Concordia" on January 13, 2012).

Obviously, it is not necessary to consider the complete usage history of each keyword: in fact, if for example the keyword "Iraq" has been extensively used in past intervals, it does not mean that it cannot become again important in the community for another event in future time frames. However, if its nutrition value stays constant during closer time intervals (for example two intervals in the same day), it means that the community is probably still referring to the same news event. In this case, according to our definition, even if the keyword can be considered as *hot*, it cannot be referred as *emerging* due to this temporal discrimination.

It is important to note that this temporal parameter influences the emerging keywords retrieved by the system. Let us consider for example the keywords reported by only one user through her tweets: if we only consider a short keyword's history (for example by taking into account only such intervals included in 24 h), the system will only detect such keywords that emerge in a daily perspective (referring to her daily activities, related for example to her job or hobbies). Otherwise, if we consider a longer history (i.e., all the intervals included in a calendar year), the resulting emerging keywords will represent globally relevant activities that modify the general daily trends (for example unexpected facts or events).

With this goal we introduce a parameter s, where $0 < s < t$, that limits the number of previous time slots considered by the system to study the keywords life cycles and defines the *history worthiness* of the resulting emerging keywords.

Now given a keyword k, it is possible to calculate its burstiness value at the time interval I^t as

$$\text{burst}_k^t = \sum_{x=t-s}^{t-1} \left(\left((\text{nutr}_k^t)^2 - (\text{nutr}_k^x)^2 \right) \cdot \frac{1}{\log(t-x+1)} \right) \tag{3.6}$$

where nutr_k^x represents the nutrition obtained by the keyword k during the interval time I^x.

This formula permits to quantify the usage of a given keyword k with respect to its previous usages in a limited number of time intervals. In fact, considering two distinct time intervals I^x and I^t, with $x < t$, this formula quantifies the difference in terms of usage of a given keyword, by considering the difference of nutritions received in the time frames I^x and I^t, and taking also into account the temporal distance among the two considered intervals.

3.3.3 Selection of Emerging Terms

This section presents the user-driven and automatic technique to select a limited set of relevant terms that emerge in the considered time interval.

At the end of this step, given a set of tweets TW^t generated on the time interval I_t, the system calculates a limited set of keywords EK^t that are emerging on the considered time interval.

3.3.3.1 User-Driven Keywords Selection

The first approach for the selection of emerging terms relies on a user-specified threshold parameter. Our initial assumption is that, given two keywords with very high burstiness values, they can be considered as emerging or not depending on the user evaluation. Indeed, if a user wants to be informed only about the most emerging events (i.e., when worldwide distribution of users reported it on a big scale, as for disasters and/or worldwide tragedies), the user probably prefers to avoid such contents that are only relatively emerging (for example, topics referring to a less globally important news event).

In order to do that, we introduce a *critical drop* value that allows the user to decide when a term is *emergent*. In particular, the critical drop is defined as

$$drop^t = \delta \cdot \frac{\sum_{k \in K^t} (\text{burst}_k^t)}{|K^t|} \tag{3.7}$$

where $\delta \geq 1$. It permits to set the critical drop by also taking into account the average burstiness value. We define the set of emerging keyword EK^t as

$$\forall k \in K^t, k \in EK^t \iff \text{burst}_k^t > drop^t \tag{3.8}$$

It is possible to note that the cardinality of EK^t is directly proportional to the value of δ.

3.3.3.2 Automatic Keywords Selection

An intrinsic limitation of the ranking approach introduced in Sect. 3.3.3.1 is that it could be very hard for a user/application to set a proper δ value. In fact, it can be a hard task to numerically quantify a threshold from an abstract perception as the desired cardinality of emerging keywords. Moreover, depending on the temporal context, it could be necessary to set the threshold value differently. Thus, we leverage, as explained in [75], a fully automatic ranking model that dynamically sets the *critical drop*. In particular, following this strategy, for each keyword expressed by the community, the most emergent are those whose burstiness is above an adaptively computed critical point (Fig. 3.4). Intuitively, to preserve only the keywords with very high burstiness, we consider only those terms whose burstiness is higher than the average burstiness of the most discussed terms. In order to cope with this, we provide a completely automatic model that works as follows:

1. the systems first ranks the keywords in descending order of burstiness value previously calculated in Sect. 3.3.2.2.
2. it computes the *maximum drop* between consecutive entries and identifies the corresponding drop point.
3. it computes the *average drop* (between consecutive entities) for all those keywords that are ranked before the identified maximum drop point.
4. the first drop which is higher than the computed average drop is called the *critical drop*.

At the end of this four-step process, the keywords ranked before the critical drop are defined as emerging keywords on time interval I^t and are collected in set called EK^t.

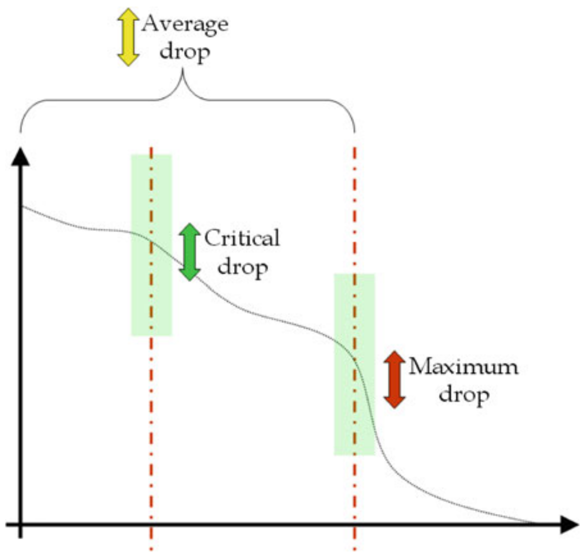


Fig. 3.4 The critical drop cut-off: the maximum drop is the highest variation in the ordered list of weights (*red mark*). The average drop (between consecutive entities) is the average difference between those items that are ranked before the identified maximum drop point (*yellow mark*). The first drop which is higher than the computed average drop is called the critical drop (*green mark*)

3.3.4 Leveraging Users' Context for Personalization Purposes

While the formula presented in Sect. 3.3.2.2 permits to monitor in real-time the stream of information expressed by the Twitter community, it does not take into account any user preference and/or context. For this, we introduce a normalization that uses the content generated by the user (i.e., the tweets/re-tweets generated by her) to personalize the list of emerging topics.

We first analyze the temporal time frames in which the user has been active within the Twitter environment. For this, we calculate the entire user's *life activity* as the time range included between the instant of her first tweet (or retweet) and the current instant.

From this, we segment the user's life activity into time intervals by taking into account her frequency of posting. The rationale is that any user has different habits and abilities with respect to the social network environment and expresses herself at different rhythms. Thus, we aim to take into account these different behaviors by analyzing the personal attitude with respect to the posting activity.

For this, we divided the user's life activity into numbered intervals, by segmenting the user lifetime based on the number of tweets generated by the user. Thus, with the function $interval(tw_j, u_i)$, we are able to determine in which interval, relatively to the considered user, each tweet has been generated. The lower the interval value, the more recent the tweet with respect to her posting frequency. Note that the value of the $interval$ function ranges from 1 to k, where k is the farthest time interval while 1 indicates the most recent one. This permits to take into account *when* the user expresses an interest to a topic (by reporting a correlated tweet) and numerically quantify her interest into the topic according to this temporal information.

At this point, given the set of tweets TW_{u_i} generated by the user u_i, we define the context of the user u_i as her vocabulary K_{u_i} containing the set of terms typed by the user in any of her tweets (or re-tweets). From this, we calculate the weight of each term z in K_{u_i}, within the context of u_i, by considering the relative time interval in which it has been generated. In detail, it is calculated as

$$p_z^{u_i} = \sum_{tw_j \in TW_{u_i}} \frac{w_{z,j}}{log(interval(tw_j, u_i) + 1)}. \tag{3.9}$$

where $w_{z,j}$ is the weight of the zth vocabulary term in jth tweet by using the augmented normalized term frequency [68] (calculated as explained in Sect. 3.3.1).

At this step, we can contextualize the importance of each retrieved emerging term (as explained in Sect. 3.3.3), by taking into account the importance of the term within the context of the considered user.

The problem of matching user contexts with emerging terms lies in the shortness of the tweets that likely makes the contexts vocabulary abridged as well. For this reason, the evaluation of the similarity between these two entities needs to be carefully chosen. In fact, the naive solution of calculating exact matches between contexts terms and emerging terms avoids to consider both morphological variations and similarities between different words with shared meanings. In that sense, we made use of the system proposed in [76] to estimate words similarities based on statistically significant co-occurrences within large corpora, like Wikipedia. An advantage of using such a resource relies on its multi-language nature (as in our system).

In detail, taking a pair of terms as input, [76] proposed to retrieve the Wikipedia articles they refer to, and compute a semantic similarity between the two terms based on the paths, in the Wikipedia categorization graph, that connect these articles. In this way, we calculate the burstiness of an emerging keyword k relatively to the user u_i as:

$$\text{burst}_{k,u_i}^t = \text{burst}_k^t * \left(1 + \frac{\sum_{z \in K_{u_i}} (\text{sim}(k, z) \cdot p_z^{u_i})}{|K_{u_i}|}\right) \tag{3.10}$$

where $|K_{u_i}|$ is the cardinality of the user context vocabulary, while sim is the function that returns a [0, 1]-similarity value between the emerging keyword and a context keyword, according to the method presented in [76]. Note that, if the semantic relatedness between the two terms results unknown (for example, on of them is not included in the considered Wikipedia data set), it is estimated as 0. On the other hand, if $k = z$, the similarity returned by the system is equal to 1.[13] An example, related to the semantic similarity between the term "keyboard" and "computer," is shown in Fig. 3.5.

[13] Notice that, considering that the semantic similarity of each pair of term is based on the external knowledge base of Wikipedia, this information is precomputed offline and it is periodically updated in order to reflect the novel information introduced in Wikipedia.

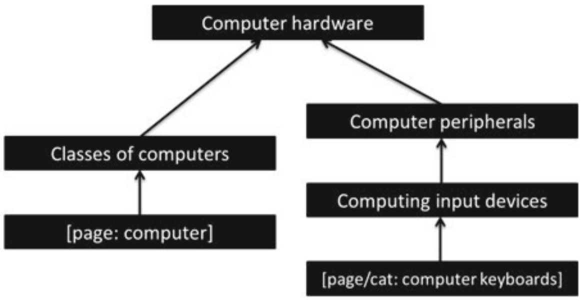


Fig. 3.5 The shortest path in Wikipedia, between the terms "keyboard" and "computer," which provides the base for calculating the semantic similarity between the terms (0.74 in this case)

Following this strategy, if an emerging term results outside the interests of the user (because no semantic paths emerged, within the Wikipedia data set, between the terms within the user context and the emerging one), we do not lower the overall burstiness of the considered emerging keyword. In fact, we only proportionally increase the burstiness of the emerging terms when they resulted semantically close to some of the interests of the user. We believe that this strategy permits to preserve the most discussed arguments and inform the users about the most important news/topics expressed by the community, even if not interested into the domain. Following this high-level assumption, when global events happen and they are extensively commented in Twitter, all the users can remain informed even if they are not specifically interested in the domain. This assumption will be further commented within the experimental analysis (Sect. 3.4.2).

3.3.5 From Emerging Terms to Emerging Topics

Considering the given corpus of tweet TW^t (representing the corpus of extracted tweets with $n = \left|TW^t\right|$ text tweets extracted within the time interval I^t), in this step we study the semantic relationships that exist among the keywords in K^t in order to retrieve the topics related to each emerging term.

In our system we define a *topic* as a minimal set of a terms semantically related to an emerging keyword. Thus, in order to retrieve the emerging topics, we consider the entire set of tweets generated by the users within the time frame I^t, and we analyze the semantical relationships that exist among the keywords by examining the co-occurrences information.

Let us consider, for example, the keyword "victory" in a given set of tweets: this term alone does not permit to express the related topic. In fact, considering as a time frame November 2008, the related topic can be easily defined by the association with other keywords (among the most used) as "elections," "Us," "Obama," and "McCain," while in other time frame, as for example February 2012, the term could be related to a sports event by other keywords as "superbowl"— due to the championship game of the National Football League (NFL)—"football" or "New York Giants"—the name of one of the teams that played the final game in 2012.

Thus, in order to express the topics related to the retrieved emerging keywords, we consider the time frame in which all the tweets have been generated and analyze the semantic relationships among the keywords based on the co-occurrences information.

We formalize this idea by associating to each keyword $k \in K^t$ a *correlation vector* $\mathbf{cv}_k^t$, formed by a set of weighted terms, which defines the relationships that exist among k and all the other keywords in the considered time interval. In other words, we compute the degree of correlation between a keyword k and another keyword z by using the set of documents containing both terms as positive evidence of the relationship between the two keywords, and the set of documents containing only one of them as positive evidence against the relationship. In detail, we treat each keyword k as a query and the set of the tweets containing the keyword TW_k^t as the *explanation* of this term in the time interval I^t.

Intuitively, this is analogous to treating (a) the keyword k as a query and (b) the set of tweets containing k as relevance feedback on the results of such query. Recognizing this, we identify the correlation weight $c_{k,z}^t$, between k and another keyword z at time I^t relying on a probabilistic feedback mechanism [77]:

$$c_{k,z}^t = log \frac{r_{k,z}/(R_k - r_{k,z})}{(n_z - r_{k,z})/(N - n_z - R_k + r_{k,z})} \times \left| \frac{r_{k,z}}{R_k} - \frac{n_z - r_{k,z}}{N - R_k} \right|, \quad (3.11)$$

where:

- $r_{k,z}$ is the number of tweets in TW_k^t containing the keywords k and z;
- n_z is the number of tweets in the corpus containing the keyword z (it is equal to $|TW_z^t|$);
- R_k is the number of tweets containing k (it is equal to $|TW_k^t|$); and
- N is the total number of tweets.

Note that the first term increases as the number of the tweets in which k and z co-occur increases, while the second term decreases when the number of tweets containing only the keyword z increases.

Thus, given a term k, we associate a so-called *correlation vector*

$$\mathbf{cv}_k^t = \langle c_{k,1}, c_{k,2}, \ldots, c_{k,v} \rangle, \quad (3.12)$$

which represents the relationships that exist between k and the other v keywords (with $v = |K^t|$) at the time interval I^t.

At this point we leverage information conveyed by the *correlation vectors* in order to identify the topics related to the emerging terms retrieved during the considered time interval. In order to do that, we construct a keyword-based topic graph in the form of a directed, node-labeled, edge-weighted graph, $TG^t(K^t, E, \rho)$, as follows:

- Let K^t be a set of vertices, where each vertex $k \in K^t$ represents a keyword extracted during the time interval I^t;

- For all $k \in K^t$ and $z \in K^t$ such that $\mathbf{cv}_k^t[z] \neq 0$, there exists an edge $\langle k, z\rangle \in E$ such that
$$\rho(\langle k, z\rangle) = \rho_{k,z} = \frac{\mathbf{cv}_k^t[z]}{\|\mathbf{cv}_k^t\|}$$

Therefore $\rho_{k,z}$ represents the relative weight of the keyword k in the corresponding vector $\mathbf{cv}_k^t$, i.e., the role of the keyword z in the context of the keyword k.

Finally, the complete graph $TG^t(K^t, E, \rho)$ is thinned by applying a locally adaptive edge thinning algorithm. For each $k \in K^t$, we consider the set of all outgoing edges and we apply an adaptive cut-off (as explained in Sect. 3.3.3.2) based on the corresponding weights. This process ensures that only those edges that represent the strongest relationships are maintained (note that, since the graph is directed and the thinning process is asymmetrical, it is possible that TG^t will contain the edge $\langle k, z\rangle$ but not vice versa).

3.3.6 Topic Detection, Labeling, and Ranking

Since in our system each topic is defined as a set of semantically related keywords, we leverage the topological structure of the topic graph TG^t to detect the emerging topics into the Twitter community. In order to do that, we consider the set of emerging keywords EK^t, computed as described in Sect. 3.3.3, and we search for the strongly connected components (SCC) rooted on them in TG^t.

According to the Kosaraju–Sharir algorithm ([78]), given a keyword k that represents a vertex within the topic graph TG^t, we find the set of vertices S reachable from k through a path, simply applying a depth-first search (DFS) visit (or any other similar algorithm). Then, we repeat the process on the same topic graph TG^t with reversed edges in order to find the set of vertices T that can reach k through a path. The strongly connected component EK_k^t is formed by all the vertices within the intersection between T and S. The complexity of this process is linear.

Thus, for each emerging keyword $z \in EK^t$, we define the related *emerging topic* as a subgraph $ET_z^t(K_z, E_z, \rho)$ representing a set of keywords semantically related to the term z within the time interval I^t. Considering the entire set of emerging keywords EK^t, in this step we compute the corresponding set of emerging topics as $ET^t = \{ET_1^t, ..., ET_n^t\}$ of strongly connected components. It is important to note that the number of the retrieved emerging topics can be lower than the number of the emerging keywords ($n \leq |EK^t|$); in fact two emerging keywords can belong to the same emerging topic.

At the end of this step, the set of keywords K_z^t belonging to the emerging topic ET_z^t is calculated by considering as starting point in TG^t the *emerging* keyword z, but also contains a set of common terms semantically related to z but not necessarily included in EK^t.

In Fig. 3.6 is depicted an example in which each topic is represented by a different color. As it can be seen, each strongly connected component contains both *emerging*

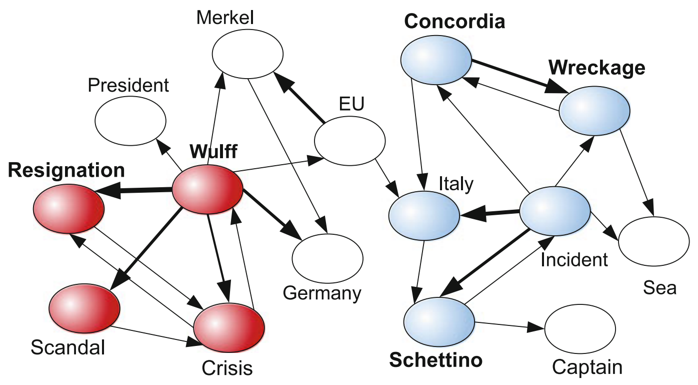


Fig. 3.6 A topic graph with two strongly connected components representing two different emerging topics: the resignation of President Wulff (in *red*) and the wreck of the Costa Concordia (in *blue*). Labels in bold represent emerging keywords while the thickness of an edge represents the semantical relationship between the considered keywords

terms (labeled in bold) like "Concordia," "Wulff," and other popular keywords, like "Merkel," "sea," and "EU," which are constantly used by twitter users and do not represent statistically emerging terms. In fact, terms like "Merkel" or "EU" represent very popular terms always reported by the users in Twitter as in any other information sources.

With this approach, we not only retrieve such terms that directly co-occur with the emerging terms, but we can also retrieve those that are indirectly correlated with the emerging ones (by co-occurring with keywords that they themselves co-occur with the emerging terms). In fact, the considered topic graph leverages the information contained in all the tweets, even those that do not report emerging terms; indeed a user can always report an emerging topic by simply using synonyms.

Thus, as a last step, we need to establish an order among the retrieved topics in order to guide the user in understanding which topic is *more* emergent in the considered time frame. In order to do that, we introduce a *ranking* value as

$$rank_{ET_z^t} = \frac{\sum_{k \in K_z^t} (\mathrm{burst}_k^t)}{|K_z^t|} \tag{3.13}$$

that leverages the average burstiness of the terms in K_z^t to define the importance of the topic led by the emerging keyword $z \in EK^t$. Finally, using this value we are able to rank the retrieved topics in descending order of importance.

At this point, the system has found a set of topics ET^t emerging within the time interval I^t. Nevertheless, it is now necessary to carefully select a minimal set of keywords (belonging to the considered topic) to represent each retrieved emerging topic to the user. In fact, we believe that too many keywords could represent an information overload for the user; on the other hand the cardinality of the retrieved emerging topics strictly depends on the topological structure of the topic graph TG^t.

In order to avoid this problem, given a topic $ET_z^t \in ET^t$ and the related keywords K_z^t, we apply an unsupervised keyword ranking mechanism (as described in Sect. 3.3.3.2) that permits to select the most representative keywords for each cluster. Note that, even using this adaptive cut-off, there is no guarantee to obtain a relatively small set of keywords representatives. Thus, we also introduce a numerical threshold χ, set by the user/application, that limits the representative keywords of each topic. This threshold can be set depending on visualization constraints of the device and/or the user preferences.[14]

3.4 Experiments: Case and User Studies

In this section, we evaluate the proposed method by analyzing real case scenarios and user studies. In particular, we conducted several experiments by monitoring the Twitter community during the period between January 10, 2012 and July 4, 2012 (i.e., we made use of the available dataset provided in [2]). In our experiments we have monitored a stream that consists of two random samples (taken in two different time intervals) among all public messages. This access level provides a statistically significant input for data mining and research applications[15]; in fact, we evaluated more than 15 million of tweets, generated by $\sim$1 million Twitter users, which included more than 600 k different keywords.[16]

The main aim of the experimental evaluation was twofold: from one side analyze, through examples and case studies, the impact and the proper setup of each parameter proposed within the presented technique. On the other hand, considering that our approach is also based on the idea to take into account the users' interests, we asked different users to provide a real feedback on the retrieved personalized topics. Moreover, as in [79], we compared the retrieve topics to the ones reported by Associated Press website,[17] evaluating the precision of the presented method.

In particular, for the first task, we analyzed different experimental setup strategies: we initially considered a real case study by evaluating the emerging topics retrieved by the system within two different time intervals. Then, we varied the number of considered time intervals (Sect. 3.3.2.2) in order to determine how this parameter affects the quality of the retrieved topics. We also compared the presented keywords selections methods (Sect. 3.3.3) evaluating their impact on the resulting emerging topics.

[14] We used $\chi = 5$ as default value.

[15] http://apiwiki.Twitter.com.

[16] The sampling rate of the used standard Twitter account is 1 % over an average of 200 million per day.

[17] http://www.ap.org.

For the second objective of this experimental evaluation, we asked human users to subjectively judge the topics and the personalization strategy through a questionnaire. Note that the evaluation of a personalized topic detection strategy results in a very complex task; in fact no manually annotated data set exists for this problem and the unique possible ground truth is provided by the users. Based on these considerations, different users, with different backgrounds, age, and interests, have been involved in a user study made up of a very diverse set of questions that aim at covering the problems faced by our approach. We also analyzed the advantages and disadvantages of the proposed approach highlighting possible future works and strategies for improvements.

3.4.1 Experiments Based on Real Case Scenarios

As previously reported, we initially evaluated the proposed approach by analyzing the retrieved emerging topics within the considered time period. For this experiment we considered the automatic selection method and we set the time range r as 30 min[18] (Sect. 3.3.1). As explained in the Introduction, considering the impressive response time of the users, our assumption is that, if continuously monitored within small time ranges, Twitter can also predate the most authoritative news sources in informing the community about emerging news events. Obviously, it is possible to set higher time range values: in this case, the resulting topics will be statistically more significant (a higher number of authors certify the importance of the argument) but the advantage in terms of time with respect to the traditional information source will vanish. The number s of time slots considered was 400 (with the selected time range, we considered more than a week). In fact this time slot size is sufficient to avoid such terms that are significant in a daily/weekly perspective (see also Sect. 3.4.1.1).

Using this experimental setup, the system implementing the framework introduced in this chapter discovered 147 emerging topics. At this point, in order to test the precision of the proposed method, we asked two external domain experts to manually compute the precision of the proposed method.[19] For this, as in [79], we considered as baseline the Really Simple Syndication (RSS) of the Associated Press website. The resulting average precision was 0.892, proving the effectiveness of the proposed system in detecting relevant topics of discussion (because they deserved to be commented in authoritative information services).

[18] In our experimental evaluation we set r equals to 30 in order to adapt our system to the high dynamicity of Twitter users. This is clearly in agreement with the experimental results shown in [27]. In fact, in this work, the authors revealed that there are few topics that last for longer times, while most topics decay in about 20–40 min.

[19] Note that we do not compute any recall value. In fact, recall is strictly dependent on the considered ground truth (CNN, AP, some online newspaper, etc.) and its news domain. For example, some important news can be reported by some authoritative news source and ignored by others. For this, counting how many news articles are detected is dependent on a specific ground truth and the resulting analyzing could not be considered significative for the evaluation task.

Table 3.1 The list of retrieved topics based on the five most emerging terms (in bold)

Date	Emerging Topics
13-01-2012	{**concordia**, cruise, disaster, schettino, sea}[a]
13-02-2012	{**adele**, grammy, 2012, perry, kate}[b]
06-02-2012	{**jubilee**, uk, england, diamond, elizabeth, II}[c]
01-02-2012	{**port**, killed, match, said}[d]
04-02-2012	{**protest**, kremlin, putin, government,}[e]

[a] http://www.bbc.co.uk/news/world-europe-16558910
[b] http://www.nytimes.com/2012/02/14/arts/music/at-the-54th-grammy-awards-everything-old-is-praised-again.html
[c] http://www.guardian.co.uk/uk/queen-diamond-jubilee
[d] http://www.bbc.co.uk/news/world-middle-east-16845841
[e] http://www.time.com/time/world/article/0,8599,2106183,00.html

In Table 3.1 we also show the retrieved topics based on the five most emerging terms (i.e., the terms with the highest burstiness values). The emerging terms are visualized in bold. We also link the professional news articles of the retrieved news topics. It can be seen that the emerging terms are always the most specific ones (i.e., "concordia," the name of the Italian cruise ship that partially sank on the night of January 13, 2012; in fact, they represent keywords that are generally very unusual in the community and only emerge in correspondence to unexpected events. Although, by analyzing the co-occurrences information in the tweets reporting such terms (as explained in Sect. 3.3.5), it is possible to link them to other popular keywords (as "sea," in the topic leaded by "concordia") that have very common usages in the community.

3.4.1.1 History Worthiness

As reported in Sect. 3.3.2.2, a term is defined as *emerging* by evaluating its usage in a limited number of previous time slots. However, depending on the selected number of considered time intervals, the retrieved topics can significantly differ. Thus, in order to study the impact of this parameter, we analyze two different numbers of considered slots, $s = 50$ and $s = 300$ (by considering again a time range r of 30 min and the automatic selection method). We recall that the parameter s, where $0 < s < t$, limits the number of previous time slots considered by the system to study the keywords life cycles and defines the history worthiness of the resulting emerging keywords.

In Table 3.2a and b we present the results obtained on January 20, 2012: the most emerging topics obtained by setting $s = 50$ represent common daily activities from a user perspective. Most of the terms, indeed, can be considered as emerging only if the system does not take into account comparable time intervals. Terms like "weekend" have very standard usages in the community due to the fact they represent periodic events. We guess that users systematically use such terms in correspondence to their natural occurrence. For example, in Fig. 3.7 we show the usage of the terms

Table 3.2 The emerging topics retrieved at January 20, 2012 by considering (a) a daily history worthiness ($s = 50$) and (b) a 7-day history worthiness. The labels in bold represent emerging terms

Date	Emerging Topics ($s = 50$)
(a)	
20-01-2012	{**tonight**, working, dead, work}
20-01-2012	{**weekend**, ready , start, Friday }
20-01-2012	{**weather**, nice , cold, morning, then}
20-01-2012	{**school**, day , later, sleep, morning}
Date	Emerging Topics ($s = 300$)
(b)	
20-01-2012	{**etta**, james, death, today, rip}
20-01-2012	{**army**, thx, day}

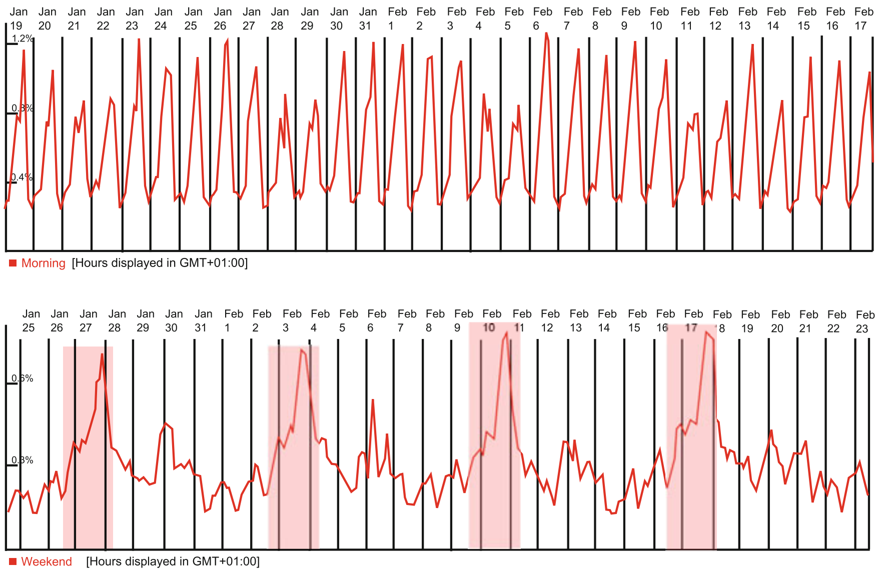


Fig. 3.7 Statistical usage in Twitter of the terms “morning” and “weekend” (provided by Trendistic) in consecutive days and/or weeks

“morning” and “weekend” in consecutive days; the system reported a peak if it only takes into account a 24 h history (or 7-day period); however, if it considers a relatively higher time frame, it recognizes a constant pattern in time. Thus, the life status of a keyword strictly depends on the considered number of time intervals (Sect. 3.3.2.2) and this value directly affects the temporal relevance of the retrieved topics.

Table 3.3 The five most emerging terms (and their related burstiness values) retrieved by the proposed temporal-based topic detection system within the considered time interval

Date	Emerging terms	Burstiness value (total avg = 135.67)
13-01-2012	**concordia**	193461.83
13-01-2012	**carre**	9677.42
13-01-2012	**sarah**	6764.79
13-01-2012	**dark**	3284.39
13-01-2012	**holmes**	1345.25

3.4.1.2 User-Selected Threshold Versus Automatic Ranking Mechanism

In Sect. 3.3.3 we reported two different selection methods (user-selected threshold and automatic) to identify emerging terms. In this experiment we evaluate the impact of each of them in the retrieved topics by analyzing the example proposed in Table 3.1: each of the considered five emerging terms (“concordia,” “adele,” “jubilee,” “port,” and “protest”) was identified as *emergent* in different time intervals.

In order to understand why they have been considered as emerging, we need to analyze the considered intervals and their associated burstiness values. Let us consider the time interval related to the term “concordia”: in Table 3.3 we show the five most emerging terms and their burstiness values retrieved by the system on January 13, 2012. We notice that, considering the automatic approach (Sect. 3.3.3.2), only the terms “concordia” has been considered as emergent. In fact, the system identified the difference in terms of burstiness values between “concordia” and the second most emerging terms (“carre”—who refers to the writer John Le Carré—) as the critical drop point and only considered as emerging such keywords that are ranked better than the critical drop.

However, considering the user-driven selection method (Sect. 3.3.3.1), the retrieved emerging terms depend on the δ value set by the user. In fact, considering for instance $\delta = 1{,}000$ (i.e., each term is identified as emergent only if its burstiness value is 1,000 times higher than the total average burstiness value), only the term “concordia” is selected as emergent. Instead, with $\delta = 10$, also the terms “carre,” “sarah,” and “dark” will be considered as emergents (and analyzed using the topic graph to retrieve the related topics).

3.4.2 *User Study: Evaluating Personalization and Topic Detection Strategies*

In order to analyze the personalization strategy of the proposed topic-detection technique, we conducted a user study by asking multiple Twitter users to reply to a questionnaire when evaluating the emerging topics, retrieved by the system, within their

Table 3.4 Two subjective questionnaires run on different time intervals and involving different users; they replied to four questions related to worldwide importance of the retrieved topics, understandability, closeness to their interest, and degree of novelty of the retrieved topics of discussion (variance is reported between parentheses)

Time interval	Closeness	Importance	Novelty	Understandability
February 2012	2.94 (0.54)	4.22 (0.56)	4.31 (0.45)	4.44 (0.41)
June 2012	3.21 (0.40)	3.98 (0.36)	4.61 (0.52)	4.12 (0.36)
Avg	3.07	4.1	4.46	4.28

interests. The users represent a various range of ages, backgrounds, jobs, interests, and education level and they have intermediate Web ability (they are not computer scientists or social networks analysts). Among all, 64 Twitter users replied to our call. Among them, 84 % of the users were men while only 16 % of the users were women. The average age of the users was 29 (with the minimum of 19 and the maximum of 54).

For this experiment, we asked to the users to analyze and report their opinions about the most emerging topics within two different time intervals; the first, from February 6 to 20, 2012 and the second from June 25 to July 4, 2012. In order to also test the reliability, we involved in the two tests different users with different foci of interest. The results are shown in Table 3.4.

The first question asked to evaluate, with a 5-point scale rating, how much the returned topics could be considered within their interests (expressed by their Twitter activity). The higher the rating, the more adherent the emerging topics to the user's interests. The result showed an average rate of 3.07, highlighting an intermediate level of pertinence of the retrieved topics to their topic interests. Note that, even if this result can appear lower than desired, it permits to report an important consideration. The personalization approach proposed in Sect. 3.3.4 does not filter the topics that most emerged when they are not within the user interests; in fact, it only highlights the discussions closer to the interests of the user but it does not filter the emerging ones that are not explicitly within them. In other words, this strategy permits to inform the users about the most important news/topics expressed by the community even if they are not specifically interested with the domain. With this approach, when global news events happen (and they are extensively discussed within the community) the user can remain informed even if not specifically interested in the domain. Let us consider for example the case of Haiti earthquake (happened in 2010) and/or the tsunami that hit Japan on 2011. Twitter collected a huge amount of data coming from people in directly-affected areas [80]. We can easily believe that very few people are specifically interested in monitoring these information domains per se, but every person could desire to be informed about these global events when they happen.

To better evaluate this assumption, we asked the users to report their opinion (with a 5-point scale rating) about the overall importance of the reported topics. For this reason, we asked the users the following question: "Can be considered the reported topics of general and world-wide interest?" The users reported their opinion for each of the presented topics. The average value was 4.1, demonstrating that the system was able to identify worldwide emerging discussions that can be considered interesting to any user, independently of the specific domain interests.

Moreover, the third question investigated the ability of the proposed mechanism to predate traditional information systems (newspapers, television, blogs, etc.). Specifically, we asked the user if they hadnalready read about the reported discussion, within alternative information channels, before being informed by our system ("Did you already heard about the topics reported by the system?"). The users expressed, again, their responses with a 5-point scale rating (the higher the rate, the more unknown the related news). The feedback of the users within the first time interval, with an average rate of 4.46, clearly showed the ability of the proposed topic detection and tracking mechanism to predate, in real-time, the traditional information systems in reporting fresh, emergent, information news.

Finally, each user also replied to the question "Were the reported topics easy to understand?" The users replied that the reported topics were, in average, very "easy to understand" (with an average subjective rate of 4.28), highlighting the fact that the proposed topic construction strategy (Sect. 3.3.5) is able to associate a minimal set of terms to each most emerging keyword that can effectively summarize the overall discussion.

Also notice that the results are consistent between the two considered time intervals, highlighting the ability of the proposed approach to automatically detect emerging topics in different conditions, with different users and with different domains of interests.

In conclusion, we can summarize these results as follows: as initially supposed, the proposed topic-detection mechanism is able to predate the classic information channels in informing the users about the most discussed news events, in a way that is both compact and understandable from a human point of view. In fact, the user study highlighted the fact that the topic construction mechanism provides sufficient details (i.e., number of keywords) to understand the core of the related event/discussion. Moreover, even when the reported topics were not within the users interests, they resulted to be important to be reported for their global importance.

3.5 Conclusions

Starting from our previous work aimed at detecting real-time emerging topics on Twitter [2], this chapter focused attention on how the interests of the users can be exploited to contextualize such mined knowledge. The approach is based on the assumption that everything that appears to be of high interest for all Twitter users worldwide, should be of interest to anyone. Thus, there is no intention to leverage user profiles as central bases for the contextualization of the extracted topics. In fact, the idea was that user contents and interests only need to be accentuated and added to world-level important events.

In this chapter, we first formalized the life cycle of a term occurring in the tweets through an aging theory that aimed at modeling the temporal usage of each keyword expressed by the community. The assumption was that important topics start from little information (i.e., single terms) that acquire unexpected importance within the user-generated data. Then, we took into account the social relationships in the Twitter network (i.e., follower and following relationships) to estimate an authority level of the authors, thus normalizing their propagation of information in the network. Finally, we constructed the topics as minimal sets of keywords co-occurring with high emerging ones. In addition, we considered the user context (given by the posted tweets) to highlight those topics that better match her specific interests.

Within our experimentations, we found that the approach was able to extract important events even before the most important information media, which usually needed a certain amount of time to react.

The related work presented in Sect. 3.2 showed that Twitter is the ideal scenario for the study of real-time information spreading phenomena. Moreover, we think that the approach could be used in the communication network analysis of both more similar microblogs, like Weibo,[20] and systems like Facebook or, more in general, in online forums.

References

1. Poblete B, Garcia R, Mendoza M, Jaimes A (2011) In: Proceedings of the 20th ACM international conference on Information and knowledge management, CIKM '11. ACM, New York, pp 1025–1030
2. Cataldi M, Di Caro L, Schifanella C (2010) In: Proceedings of the 10th international workshop on multimedia data mining, MDMKDD '10. ACM, New York, pp 4:1–4:10. http://doi.acm.org/10.1145/1814245.1814249.
3. Page L, Brin S, Motwani R, Winograd T (1998) In: Proceedings of the 7th international world wide web conference. ACM, New York, pp 161–172
4. Allan J (2002) Topic detection and tracking: event-based information organization. Kluwer International series on information retrieval. Kluwer Academic Publishers, Norwell. http://books.google.it/books?id=50hnLI_Jz3cC
5. Makkonen J, Ahonen-Myka H, Salmenkivi T (2004) Inf Retr 7(3–4), p 347. http://dx.doi.org/10.1023/B:INRT.0000011210.12953.86
6. Treeratpituk P, Callan T (2006) In: dg.o '06: Proceedings of the 2006 international conference on digital government research. ACM, New York, pp 167–176. http://doi.acm.org/10.1145/1146598.1146650
7. Goldberg D, Nichols D, Oki BM, Terry D (1992) Commun ACM 35(12):61
8. Hassan A, Radev DR, Cho J, Joshi A (2009) In: Proceedings of the 3rd international aaai conference on weblogs and social media (ICWSM). The AAAI Press, Menlo Park, pp 34–41
9. Melville P, Mooney RJ, Nagarajan R (2001) In: Proceedings of the 2001 SIGIR workshop on recommender systems. ACM, New York, pp 16–23
10. Balabanovic M, Shoham Y (1997) Commun ACM 40:66
11. Chen J, Nairn R, Nelson L, Bernstein M, Chi E (2010) In: Proceedings of the SIGCHI conference on human factors in computing systems, CHI '10. ACM, New York, pp 1185–1194. doi:10.1145/1753326.1753503

[20] http://www.weibo.com.

12. Chen J, Geyer E, Dugan C, Muller M, Guy I (2009) In: CHI '09: Proceedings of the 27th international conference on Human factors in computing systems. ACM, New York, pp 201–210. http://doi.acm.org/10.1145/1518701.1518735
13. Jäschke R, Marinho L, Hotho A, Schmidt-Thieme M, Stumme G (2007) In: PKDD 2007. Springer, Berlin, pp 506–514. http://dx.doi.org/10.1007/978-3-540-74976-9
14. Granovetter M (1978) Am J Sociol 83(6):1420. doi:10.1086/226707
15. Goldenberg J, Libai B, Muller E (2001) Mark Lett 12(3):211
16. Gruhl D, Guha R, Liben-Nowell D, Tomkins A (2004) In: Proceedings of the 13th international conference on world wide web, WWW '04. ACM, New York, pp 491–501. doi:10.1145/988672.988739
17. Goyal A, Bonchi F, Lakshmanan LV (2010) In: Proceedings of the third ACM international conference on web search and data mining, WSDM '10. ACM, New York, pp 241–250. doi:10.1145/1718487.1718518
18. Yang J, Leskovec J (2010) In: Data mining (ICDM), 2010 IEEE 10th international conference on IEEE computer society, Washington, pp 599–608
19. Cha M, Haddadi H, Benevenuto F, Gummadi KP (2010) In: Proceedings of the 4th international AAAI conference on weblogs and social media (ICWSM). The AAAI Press, Menlo Park, pp 10–17
20. Yang J, Counts S (2010) In: Proceedings of the 4th international AAAI conference on weblogs and social media (ICWSM). The AAAI Press, Menlo Park, pp 355–358
21. Griffiths TL, Steyvers M (2004) Proceedings of the National Academy of Sciences 101 (Suppl. 1):5228
22. Zhao Q, Mitra P, Chen B (2007) In: Proceedings of the 22nd national conference on artificial intelligence—volume 2, AAAI'07. AAAI Press, Menlo Park, pp 1501–1506. http://dl.acm.org/citation.cfm?id=1619797.1619886
23. Qi Y, Candan KS (2006) In: HYPERTEXT '06. ACM, New York, pp 1–10. http://doi.acm.org/10.1145/1149941.1149944
24. Favenza A, Cataldi M, Sapino ML, Messina A (2008) In: NLDB '08. Springer, Berlin, pp 226–232. http://dx.doi.org/10.1007/978-3-540-69858-6
25. Wu Y, Ding Y, Wang X, Xu J (2010) J Comput 5(4):549
26. Abrol S, Khan L (2010) In: GIR '10: Proceedings of the 6th workshop on geographic information retrieval. ACM, New York, pp 1–8
27. Asur S, Huberman BA, Szabó G, Wang C (2011) In: Fifth international AAAI conference on weblogs and social media. The AAAI Press, Menlo Park, California
28. Katz JS, Katz JS, Martin BR, Martin BR (1997) Res Policy 26:1
29. Crane D (1969) Am Soc Rev 3:335
30. Chubin DE (1976) Soc Q 17(4):448
31. Shapin S (1981) Med Hist 25(3):341
32. de Beaver D, Rosen R (1979) Scientometrics 1(2):133
33. Melin G, Persson O (1996) Scientometrics 36:363
34. Newman MEJ (2001) Phys Rev E 64(1). http://dx.doi.org/10.1103/PhysRevE.64.016131
35. Barabasi AL, Jeong H, Neda Z, Ravasz E, Schubert A, Vicsek T (2002) Physica A: Stat Mech Appl 311(3–4):590
36. Schifanella C, Caro LD, Cataldi M, Aufaure MA (2012) in KDD. ACM, New York, NY, USA
37. Di Caro L, Cataldi M, Schifanella C (2012) Scientometrics pp 1–25. doi:10.1007/s11192-012-0762-1. http://dx.doi.org/10.1007/s11192-012-0762-1
38. Moon S, You J, Kwak H, Kim D, Jeong H (2010) In: Communication systems and networks (COMSNETS), 2010 second international conference on IEEE Press, Piscataway, pp 1–10
39. Hou H, Kretschmer H, Liu Z (2008) Scientometrics 75(2):189
40. Chen CC, Chen YT, Sun YS, Chen MC (2003) in ECML. Springer, Berlin
41. Wang C, Zhang M, Ru L, Ma S (2008) In: CIKM '08. ACM, New York, pp 1033–1042. http://doi.acm.org/10.1145/1458082.1458219
42. He Q, Chang K, Lim EP (2007) Data mining, IEEE international conference on 0, 493. http://doi.ieeecomputersociety.org/10.1109/ICDM.2007.17

43. Sriram B, Fuhry D, Demir E, Ferhatosmanoglu H, Demirbas M (2010) In Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval, SIGIR '10. ACM, New York, pp 841–842. doi:10.1145/1835449.1835643. http://doi.acm.org/10.1145/1835449.1835643
44. Becker H, Naaman M, Gravano L (2011) In: Fifth international AAAI conference on weblogs and social media. The AAAI Press, Menlo Park
45. Becker H, Naaman M, Gravano L (2010) in WSDM. ACM, New York
46. Sankaranarayanan J, Samet H, Teitler B, Lieberman M, Sperling J (2009) In: Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems. ACM, New York, pp 42–51
47. Petrović S, Osborne M, Lavrenko V (2010) In: Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics, HLT '10. Association for Computational Linguistics, Stroudsburg, pp 181–189. http://dl.acm.org/citation.cfm?id=1857999.1858020
48. Bun KK, Ishizuka M, Ishizuka BM (2002) In: Proceedings of 3rd Int'l conference on web informtion systems engineering (WISE 2002). IEEE Computer Society, Washington, 2002), pp 73–82
49. Lampos V, Cristianini N (2012) ACM Trans Intell Syst Technol 3(4), 72:1. doi:10.1145/2337542.2337557http://doi.acm.org/10.1145/2337542.2337557
50. Takeshi Sakaki MO, Matsuo Y (2010) ACM. USA, New York
51. AlSumait L, Barbará D, Domeniconi C (2008) In: Proceedings of the 2008 eighth IEEE international conference on data mining, ICDM '08. IEEE computer society, Washington, pp 3–12. doi:10.1109/ICDM.2008.140. http://dx.doi.org/10.1109/ICDM.2008.140
52. Sugiyama K, Hatano K, Yoshikawa M (2004) In: Proceedings of the 13th international conference on world wide web, WWW '04. ACM, New York, pp 675–684.doi:10.1145/988672.988764
53. Teevan J, Dumais ST, Horvitz E (2005) In: Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '05. ACM, New York, pp 449–456. http://doi.acm.org/10.1145/1076034.1076111.http://doi.acm.org/10.1145/1076034.1076111
54. Ziegler CN, McNee SM, Konstan JA, Lausen G (2005) In: Proceedings of the 14th international conference on world wide web, WWW '05. ACM, New York, pp 22–32. http://doi.acm.org/10.1145/1060745.1060754.http://doi.acm.org/10.1145/1060745.1060754
55. Noll MG, Meinel C (2007) In: Proceedings of the 6th international the semantic web and 2nd Asian conference on Asian semantic web conference, ISWC'07/ASWC'07. Springer, Berlin, pp 367–380. http://dl.acm.org/citation.cfm?id=1785162.1785190
56. Teevan J, Dumais S, Horvitz E (2005) In: Proceedings of the workshop on new technologies for personalized information access (PIA), pp 84–92
57. Lin GL, Peng H, Ma QL, Wei J, Qin JW (2010) In: Machine learning and cybernetics (ICMLC), 2010 international conference on, vol. 5, IEEE Computer Society, Washington, pp 2116–2421. doi:10.1109/ICMLC.2010.5580733
58. Agrawal R, Gollapudi S, Halverson A, Ieong S (2009) In: Proceedings of the second ACM international conference on web search and data mining, WSDM'09. ACM, New York, pp 5–14. http://doi.acm.org/10.1145/1498759.1498766
59. Radlinski F, Dumais S (2006) In: Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '06. ACM, New York, pp 691–692. http://doi.acm.org/10.1145/1148170.1148320
60. Wedig S, Madani (2006) In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '06. ACM, New York, pp 742–747. http://doi.acm.org/10.1145/1150402.1150497
61. Cantador I, Bellogín A, Vallet D (2010) In: Proceedings of the fourth ACM conference on recommender systems, RecSys '10. ACM, New York, pp 237–240. http://doi.acm.org/10.1145/1864708.1864756

62. Xu S, Bao S, Fei B, Su Z, Yu Y (2008) In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '08. ACM, New York, pp 155–162. http://doi.acm.org/10.1145/1390334.1390363
63. Han X, ShenZ, Miao C, Luo X (2010) In: Proceedings of the 6th international conference on Active media technology, AMT '10. Springer, Berlin, pp 34–46. http://dl.acm.org/citation.cfm?id=1886192.1886201
64. Sieg A, Mobasher B, Burke R (2007) In: Proceedings of the sixteenth ACM conference on conference on information and knowledge management, CIKM '07. ACM, New York, pp 525–534. http://doi.acm.org/10.1145/1321440.1321515
65. Gauch S, Chaffee J, Pretschner A (2003) Web intelligence and agent systems 1, 219. http://dl.acm.org/citation.cfm?id=1016416.1016421
66. Wang Q, Jin H (2010) In: Proceedings of the 19th ACM international conference on Information and knowledge management CIKM '10. ACM, New York, pp 999–1008. http://doi.acm.org/10.1145/1871437.1871564
67. Carmel D, Zwerdling N, Guy I, Ofek-Koifman S, Har'el N, Ronen I, Uziel E, Yogev S, Chernov S (2009) In: Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09. ACM, New York, pp 1227–1236. doi:10.1145/1645953.1646109
68. Salton G, Buckley C (1988) In: Information processing and management. Cornell University, Ithaca
69. Weng J, Lim EP, Jiang J, He Q (2010) In: Proceedings of the third ACM international conference on web search and data mining, WSDM '10. ACM, New York, pp 261–270. http://doi.acm.org/10.1145/1718487.1718520
70. Bakshy E, Hofman JM, Mason WA, Watts DJ (2011) In: Proceedings of the fourth ACM international conference on Web search and data mining, WSDM '11. ACM, New York, pp 65–74. http://doi.acm.org/10.1145/1935826.1935845
71. Kwak H, Lee C, Park H, Moon S (2010) In: Proceedings of the 19th international conference on world wide web, WWW '10. ACM, New York, pp 591–600. doi:10.1145/1772690.1772751
72. Leskovec J, Faloutsos C (2006) In: KDD '06: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 631–636. http://doi.acm.org/10.1145/1150402.1150479
73. Glance NS, Hurst M, Tomokiyo T (2004) In: WWW 2004 Workshop on the weblogging ecosystem. ACM, New York. http://www.blogpulse.com/papers/www2004glance.pdf
74. Liang X, Chen W, Bu J (2010) In: Computer engineering and technology (ICCET), 2010 2nd international conference on, vol. 6, IEEE Computer Society, Washington, pp 249–253
75. Cataldi M, Schifanella C, Candan KS, Sapino ML, Di Caro L (2009) In Proceedings of the international conference on management of emergent digital ecosystems, MEDES '09. ACM, New York, pp 33:218–33:225. http://doi.acm.org/10.1145/1643823.1643864
76. Ponzetto SP, Strube M (2007) In: Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions, ACL '07. Association for Computational Linguistics, Stroudsburg, pp 49–52. http://dl.acm.org/citation.cfm?id=1557769.1557785
77. Ruthven I, Lalmas M (2003) Knowl Eng Rev 18(2), 95. doi:10.1017/S0269888903000638. http://dx.doi.org/10.1017/S0269888903000638
78. Aho AV, Hopcroft JE, Ullman J (1983) Data structures and algorithms, 1st edn. Addison-Wesley Longman Publishing Co., Inc, Boston
79. Lu R, Xu Z, Zhang Y, Yang Q (2012) in PAKDD (2). Springer, Berlin
80. Acar A, Muraki Y (2011) Int J Web Based Commun 7(3):392

Chapter 4
Mining Popular Routes from Social Media

Ling-Yin Wei, Yu Zheng and Wen-Chih Peng

Abstract The advances in location-acquisition technologies have led to a myriad of spatial trajectories. These trajectories are usually generated at a low or an irregular frequency due to applications' characteristics or energy saving, leaving the routes between two consecutive points of a single trajectory uncertain (called an uncertain trajectory). In this paper, we present a Route Inference framework based on Collective Knowledge (abbreviated as RICK) to construct the popular routes from uncertain trajectories. Explicitly, given a location sequence and a time span, the RICK is able to construct the top-*k* routes which sequentially pass through the locations within the specified time span, by aggregating such uncertain trajectories in a mutual reinforcement way (i.e., uncertain $+$ uncertain $\rightarrow$ certain). Our work can benefit trip planning, traffic management, and animal movement studies. The RICK comprises two components: *routable graph construction* and *route inference*. First, we explore the spatial and temporal characteristics of uncertain trajectories and construct a routable graph by collaborative learning among the uncertain trajectories. Second, in light of the routable graph, we propose a routing algorithm to construct the top-*k* routes according to a user-specified query. We have conducted extensive experiments on two real datasets, consisting of Foursquare check-in datasets and taxi trajectories. The results show that RICK is both effective and efficient.

L.-Y. Wei (✉)
Department of Computer Science, National Chiao Tung University,
1001 University Road, 300 Hsinchu, Taiwan
e-mail: lywei.cs95g@nctu.edu.tw

Y. Zheng
Microsoft Research Asia, Beijing, China
e-mail: yuzheng@microsoft.com

W.-C. Peng
National Chiao Tung University, Hsinchu, Taiwan
e-mail: wcpeng@cs.nctu.edu.tw

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_4

4.1 Introduction

The increasing availability of location-acquisition technologies (e.g., GPS) has led to a huge volume of spatial trajectories that represent the movement routes of humans, animals, hurricanes, and vehicles. Without loss of generality, a trajectory is a sequence of data points where each data point records location information and a time-stamp [18]. For example, the driving routes of vehicles and migratory routes of animals are usually recorded by GPS trajectories. Meanwhile, users could perform check-in services (e.g., Foursquare) to note their locations via a mobile phone and share their photos and activities. The time-ordered check-in records of a user are able to be expressed by trajectories. Moreover, on a photo sharing website (e.g., Flickr), people share geotagged photos whose time-stamps and geolocations can be represented as trajectories as well. However, these trajectories are usually generated at a low frequency due to energy saving and features of applications, resulting in the uncertainty of a moving object's mobility in a trajectory.

Figure 4.1 shows statistic information from Foursquare datasets in Manhattan. As shown in Fig. 4.1a, most check-in time intervals vary from 1 to 180 min. Moreover, we further investigate the distances among these check-in records. The medians of the distances between two check-in records are less than 2 km in Fig. 4.1b. The above two observations show that even in Manhattan, which has a lot of tourists, the uncertain routes apparently exist between two check-in records.

These trajectories with low sampling rate do not detail the routes, and raise uncertain routes between two consecutive sampled points in the trajectories. In this paper, we call such trajectories uncertain trajectories. Examples of uncertain trajectories are illustrated in Fig. 4.2. Figure 4.2a shows two check-in trajectories, tra$_1$:A $\rightarrow$ C $\rightarrow$ D and tra$_2$:D $\rightarrow$ B, in a rural space (i.e., road network information is not available). If a tourist would like to travel from q_1 to q_2, he/she may have no idea of how to travel without the aid of road networks or by referring to a trajectory (e.g., tra$_1$ or tra$_2$). Another example is, given one migratory trajectory of a bird, we do not know where the bird flew between two sampled points which are several miles away from each other. Due to the uncertainty of the trajectories with low sampling rate, how to derive detailed routes from uncertain trajectories is an important task.

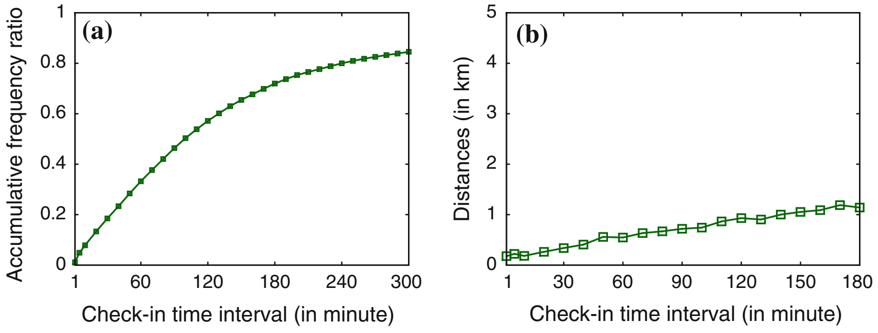


Fig. 4.1 Observations from Foursquare datasets

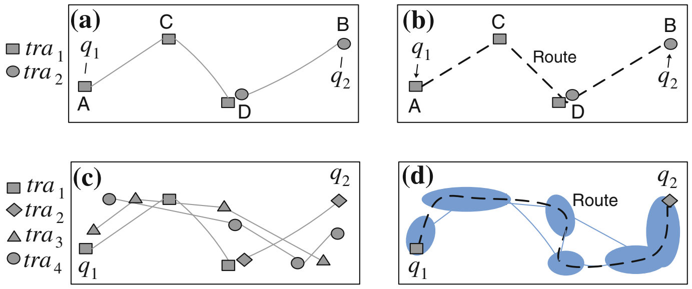


Fig. 4.2 Examples of uncertain trajectories. **a** Uncertain trajectories. **b** Simple concatenation. **c** Uncertain trajectories. **d** Mutual reinforcement

The prior work [17] proposed a framework to discover the routes from historical trajectories. Explicitly, given a set of historical trajectories, an underlying road network, and a location sequence, the work aims to suggest the top-k possible routes sequentially passing the queried locations. Note that by the aid of the given road network, the work explores possible routes derived from road networks. However, for some applications (e.g., animal migration routes or hurricane routes), road network information is not available. As for check-in datasets and geo-photo datasets, the service providers may not have road network information either. Without road network information, the work [17] cannot derive the top-k routes.

In this paper, we propose a Route Inference framework based on Collective Knowledge (abbreviated as RICK) to construct the popular routes from uncertain trajectories without road network information. Explicitly, given a location sequence and a time span, RICK constructs the top-k routes, sequentially passing the locations within the specified time span. The RICK is beneficial for many practical applications. Examples of applications are trip planning [1, 4, 6, 8, 14, 16], animal movement behavior studies [7] and traffic flow analysis [15]. For example, a user plans to take a tour that consists of three attractions (e.g., the Temple of Heaven, the Palace Museum and the Houhai Bar Street in Fig. 4.3a) in Beijing, while having no idea of how to travel around them. At this moment, the RICK can recommend the popular travel route,

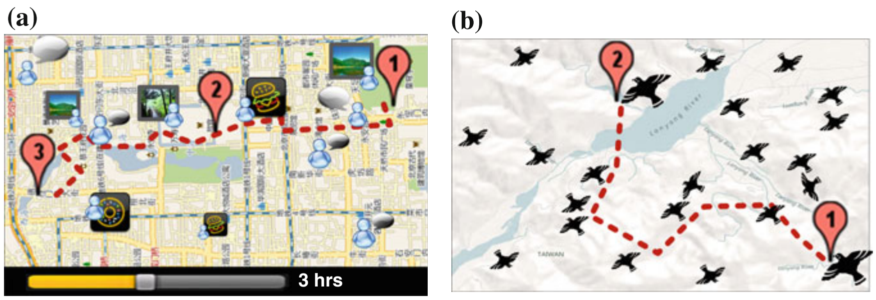


Fig. 4.3 Scenarios of applications

inferred from check-in records (or geo-tagged photos) generated by other tourists. Another example in Fig. 4.3b is to help biologists discover birds' moving behaviors from uncertain trajectories.

To infer routes from uncertain trajectories, in many cases, we need to construct a route based on segments from multiple uncertain trajectories as there is no historical trajectory passing all the queried locations. For instance, in Fig. 4.2a, tra_1 and tra_2 do not pass through both q_1 and q_2. If a tourist would like to travel from q_1 to q_2, existing trip planning [6, 12, 14, 16] could explore the sequential relations among these places and derive A → C → D → B by concatenation, as shown in Fig. 4.2b. However, the distances between two consecutive locations are still far away and a travel route still cannot be derived according to the users' trajectory. One could extract the trajectories that capture similar movements. However, due to the sparseness of data points in uncertain trajectories, two uncertain trajectories usually have totally different geospatial representations even though the two trajectories follow the same route or have the same subroutes (i.e., are correlated). As such, the similarity between two uncertain trajectories is hard to measure.

In this paper, given a set of uncertain trajectories (e.g., Fig. 4.2c), a routable graph (e.g., the blue part in Fig. 4.2d) is generated for indicating routing information in a free space by exploring spatio-temporally correlated uncertain trajectories. In light of the routable graph, we have designed a route score function and proposed a routing algorithm to construct the top-k routes (e.g., the dashed curve in Fig. 4.2d) satisfying the query. We have conducted extensive experiments on two real datasets and the results show the effectiveness and efficiency of the RICK.

The contributions of this paper are summarized as follows:

- Without the aid of road networks, we develop a route inference framework to infer routes from uncertain trajectories.
- We propose a routable graph with routing information, generated by exploring spatio-temporal correlations among uncertain trajectories.
- In light of the routable graph, we define a route score function and develop a routing algorithm to construct the top-k routes.
- We have conducted extensive experiments using real datasets of 15,000 driving trajectories and 6,600 check-in sequences. The results indicate the RICK is effective and efficient.

The remainder of the paper is organized as follows. Section 4.2 gives the preliminary of our work. Section 4.3 illustrates the routable graph construction. Section 4.4 details the route inference. Section 4.5 presents the experimental results. Section 4.6 reviews related work. Section 4.7 concludes the paper.

4.2 Preliminary

We present some terms and the problem addressed in this paper, and then overview the proposed framework.

Definition 4.1 (*Trajectory*) A *trajectory* tra_i is a time-ordered sequence of sampled points, i.e., $\text{tra}_i{:}p_1^i \rightarrow p_2^i \cdots \rightarrow p_n^i$. Each point is $p_k^i \in \text{tra}_i$ represented by $(p_k^i.l, p_k^i \cdot t)$ where $p_k^i.l$ is a geographic coordinate (a location for short) at a timestamp $p_k^i \cdot t$.

As discussed before, trajectories are usually generated at a low sampling rate, leading to the real route between two consecutive points of a trajectory being uncertain. If a time interval between two consecutive sampled points is large, the uncertainty of the route between the two points would increase. In this paper, we further claim that road networks are not always available for inferring routes between two locations. For example, for animal trace data and outdoor activities in urban areas, the movements are not along road networks. Thus, our problem is defined as follows:

Problem Given an uncertain trajectory dataset D and a user-specified query consisting of a time span Δt and a location sequence $q{:}q_1 \rightarrow q_2 \rightarrow \cdots \rightarrow q_m$, we infer the top-$k$ popular routes in a free space such that each route sequentially traverses the given locations, and the travel time of the route between any two consecutive query locations is within Δt.

Figure 4.4 overviews our RICK framework, which consists of two components: routable graph construction and route inference.

Routable graph construction: This offline component builds up a routable graph from an uncertain trajectory dataset. To generate a routable graph, there are two stages: region construction and edge inference. First, we partition the space into disjoint cells and then index the given uncertain trajectories in the gridded space. By exploring the spatio-temporal characteristics of the uncertain trajectories passing through these cells, we merge these individual cells to form some geographical regions. Here, each cell forms a vertex in the routable graph that we are going to build up. Second, we infer the edges between the cells with the uncertain trajectories. These edges can be categorized into two types: inside a region or between regions. The information

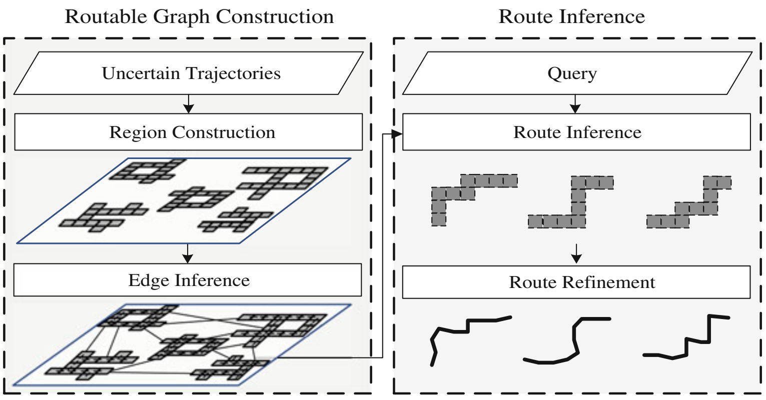


Fig. 4.4 Overview of RICK

inferred for an edge comprises a moving direction, a transition support, and a travel time, indicating the transition relationship between two cells.

Route inference: This component, consisting of route generation and route refinement, is responsible for on-line queries. In the route generation stage, given a query, we propose a routing algorithm to infer the top-k rough routes, each of which is represented by a sequence of cells, with the constructed graph. The routing algorithm first finds out the qualified subroutes between any two consecutive query locations, and then concatenates these subroutes into completed routes in a branch-and-bound manner. In addition, we define a score function based on historical movements for ranking these routes. In the route refinement stage, we further refine each rough route to derive a detailed route represented by a sequence of consecutive segments from historical data points of uncertain trajectories.

4.3 Routable Graph Construction

4.3.1 Region Construction

To construct a routable graph, we discover connected geographical areas by collaborative learning among historical uncertain trajectories. We first observe the spatial and temporal characteristics of the uncertain trajectories. For instance, Fig. 4.5a shows three trajectories, $\text{tra}_1 : p_1^1 \xrightarrow[10\text{ min}]{} p_2^1 \xrightarrow[20\text{ min}]{} p_3^1$, $\text{tra}_2 : p_1^2 \xrightarrow[13\text{ min}]{} p_2^2 \xrightarrow[10\text{ min}]{} p_3^2$, and $\text{tra}_3 : p_1^3 \xrightarrow[30\text{ min}]{} p_2^3$, where times are the travel times between two consecutive points. The locations of data points of tra_1 and tra_2 are different even if the two trajectories follow the same route (e.g., the black solid line). We observe that (1) $p_2^1 \cdot l$ and $p_2^2 \cdot l$ are at the same place; (2) tra_1 and tra_2 have similar travel times from their first points (e.g., p_1^1, p_1^2) to the place; (3) $p_1^1 \cdot l$ and $p_1^2 \cdot l$ are spatially close. The observations indicate that the route of tra_1 from p_1^1 to p_2^1 and the route of tra_2 from p_1^2 to p_2^2 may be the same. We say that the two subtrajectories $p_1^1 \to p_2^1$ and $p_1^2 \to p_2^2$ are *spatio-temporally correlated* (*st-correlated*). Moreover, if p_1^1 and p_1^2 are sampled on the same route, $p_1^1 \cdot l$ and $p_1^2 \cdot l$ could be connected. Specifically, this means that there exists at least

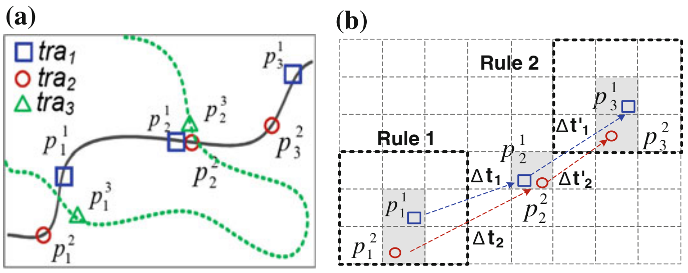


Fig. 4.5 Spatio-temporally correlated uncertain trajectories

one route between $p_1^1 \cdot l$ and $p_1^2.l$. On the other hand, in Fig. 4.5a, although $p_2^1 \cdot l$ and $p_2^3 \cdot l$ are at the same place and p_1^1 and $p_1^3 \cdot l$ are spatially close, tra$_3$ may be sampled from the other route (e.g., the green dotted line). The reason is that tra$_3$ has a longer travel time from p_1^3 to p_2^3.

Based on the aforementioned observations, we define some terms for constructing connected geographical areas. Table 4.1 summarizes the notations used in this paper. To clearly describe spatial relations among data points of uncertain trajectories, we adopt a gridded space. First, we divide a geographical range into disjoint cells by a given cell length l. The set of the cells is denoted as $\mathcal{G}$, and the index of a cell $\mathscr{G}$ is represented by (x, y), called GID. Moreover, each point of an uncertain trajectory can be mapped into a cell, and an uncertain trajectory can be transformed into a sequence of cells. As such, given an uncertain trajectory tra$_i$: $p_1^i \to p_2^i \to \cdots \to p_n^i$, the trajectory can be transformed into $p_1^i \cdot g \to p_2^i \cdot g \to \cdots \to p_n^i \cdot g$, where $p_i^k \cdot g$ represents the cell that $p_i^k.l$ locates in. Given two cells $g = (x, y)$ and $g' = (x', y')$, the cells g and g' are called *spatially close* if $|x - x'| \le 1$ and $|y - y'| \le 1$.

To explore connected geographical areas, we formally define st-correlated relations among uncertain trajectories. Given a cell g, if $g = p_k^i \cdot g$ for some $p_i^k \in$ tra$_i$, we say that tra$_i$ traverses cell g, denoted $g \in$ tra$_i$. If tra$_i$ traverses from cell g to cell g', we say $g \to g' \in$ tra$_i$. The set of the travel times of $g \to g'$ by tra$_i$ is denoted by $\mathscr{T}(\mathscr{G} \to \mathscr{G}', \text{tra}_i)$. If $g \to g' \in$ tra$_i$, $\mathscr{T}(\mathscr{G} \to \mathscr{G}', \text{tra}_i) \neq \emptyset$.

Definition 4.2 (*Spatio-temporally correlated relation*) Given two uncertain trajectories tra$_i$: $p_1^i \to \cdots \to p_n^i$ and tra$_j$: $p_1^i \to \cdots \to p_m^j$, and *a temporal constraint* θ, tra$_i$'s subtrajectory $p_k^i \to \cdots \to p_{k'}^i$ and tra$_j$'s subtrajectory $p_h^j \to \cdots \to p_{h'}^j$ are st-correlated if

(1) $\exists\ \Delta t_1 \in \mathscr{T}(p_k^i \cdot g \to p_{k'}^i \cdot g, \text{tra}_i)$, $\Delta t_2 \in \mathscr{T}(p_h^j \cdot g \to p_{h'}^j \cdot g, \text{tra}_j)$ s.t. $\frac{|\Delta t_1 - \Delta t_2|}{\max\{\Delta t_1, \Delta t_2\}} \le \theta$;

(2) One of the two rules is satisfied:
Rule 1: $p_k^i \cdot g$ and $p_h^j \cdot g$ are spatially close, and $p_{k'}^i \cdot g = p_{h'}^j \cdot g$.
Rule 2: $p_k^i \cdot g = p_h^j \cdot g$, and $p_{k'}^i \cdot g$ and $p_{h'}^j \cdot g$ are spatially close.

Note that if a trajectory tra$'$ is a subtrajectory of a trajectory tra, we denote it as tra$' \subseteq$ tra.

Table 4.1 Notations

Symbol	Description
p_i^j	The *j*th point in trajectory *i*
$p \cdot g$	The cell that *p* locates in
(x,y)	A cell id
$c(x,y)$	The number of distinct trajectories traversing (x,y)
θ	A temporal constraint
C	A minimum connection support
$T(\mathscr{G} \to \mathscr{G}', \text{tra})$	A set of the travel times from g to g' by tra

Given trajectories in Fig. 4.5a, b depicts the st-correlated relation between tra_1 and tra_2. Let $\Delta t_1 \in \mathscr{T}(p_1^1 \cdot g \rightarrow p_2^1 \cdot g, \text{tra}_1)$ and $\Delta t_2 \in \mathscr{T}(p_1^2 \cdot g \rightarrow p_2^2 \cdot g, \text{tra}_2)$ and assume Δt_1 and Δt_2 satisfy a given temporal constraint. According to Rule 1 in Definition 4.2, $p_1^1 \rightarrow p_2^1$ and $p_1^2 \rightarrow p_2^2$ are st-correlated, because $p_1^1 \cdot g$ and $p_1^2 \cdot g$ are spatially close and $p_2^1 \cdot g = p_2^2 \cdot g$. Similarly, let $\Delta t_1' \in \mathscr{T}(p_2^1 \cdot g \rightarrow p_3^1 \cdot g, \text{tra}_1)$ and $\Delta t_2' \in \mathscr{T}(p_2^2 \cdot g \rightarrow p_3^2 \cdot g, \text{tra}_2)$ and assume $\Delta t_1'$ and $\Delta t_2'$ satisfy a given temporal constraint. According to Rule 2 in Definition 4.2, $p_2^1 \rightarrow p_3^1$ and $p_2^2 \rightarrow p_3^2$ are st-correlated since $p_2^1 \cdot g = p_2^2 \cdot g$ and $p_3^1 \cdot g$ and $p_3^2 \cdot g$ are spatially close.

Definition 4.3 (*Connection support*) Given an uncertain trajectory dataset D, a set of cells $\mathcal{G}$, a temporal constraint θ, and two cells $g, g' \in \mathcal{G}$, where $\mathscr{G}$ and $\mathscr{G}'$ are spatially close, *the connection support of the cell pair* $(\mathscr{G}, \mathscr{G}')$ is defined as $|T_1 \cup T_2|$ where $T_1 = \{(\text{tra}_i, \text{tra}_j) | \text{tra}_i'$ and $\text{tra}_j{}'$ are st-correlated, $g \rightarrow g'' \in \text{tra}_i'$, and $g \rightarrow g'' \in \text{tra}_j'$ for some $g'' \in \mathcal{G} - \{\mathscr{G}, \mathscr{G}'\}$, $\text{tra}_i' \subseteq \text{tra}_i$, $\text{tra}_j' \subseteq \text{tra}_j$, and $T_2 = \{\text{tra}_i, \text{tra}_j) | \text{tra}_i'$ and tra'_j are st-correlated, $g'' \rightarrow g \in \text{tra}'_i$, and $g'' \rightarrow \in \text{tra}_j'$ for some $g'' \in \mathcal{G} - \{\mathscr{G}, \mathscr{G}'\}$, $\text{tra}_i' \subseteq \text{tra}_i, \text{tra}_j' \subseteq \text{tra}_j\}$.

For example, in Fig. 4.5b, given $p_1^1 \cdot g$ and $p_1^2 \cdot g$, which are spatially close, the support of the cell pair $(p_1^1 \cdot g, p_1^2 \cdot g) = |T_1 \cup T_2| = 1$ because $T_1 = \{(\text{tra}_1, \text{tra}_2)\}$ and $T_2 = \emptyset$. Similarly, given $p_2^1 \cdot g$ and $p_3^2 \cdot g$, which are spatially close, the support of the cell pair $(p_3^1 \cdot g, p_3^2 \cdot g) = |T_1 \cup T_2| = 1$ because $T_1 = \emptyset$ and $T_2 = \{(\text{tra}_1, \text{tra}_2)\}$.

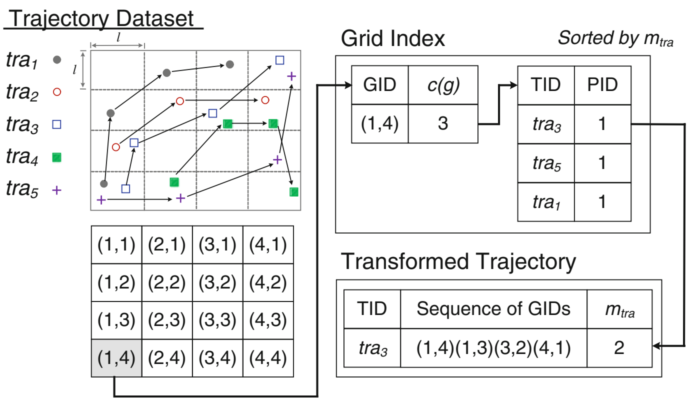
Definition 4.4 (*Neighbor*) Given an uncertain trajectory dataset D, a set of cells $\mathcal{G}$, two cells $\mathscr{G}, \mathscr{G}' \in \mathcal{G}$, a temporal constraint θ, and a *minimum connection support* C, if the connection support of the cell pair (g, g') is greater than or equal to C, g and g' are *neighbors*, denoted as gNg'.

We define a *region* as a connected geographical area as follows:

Definition 4.5 (*Region*) Given a set of cells $\mathcal{G}'$, $\mathcal{G}'$, forms *a* region if for any two cells $\mathscr{G}, \mathscr{G}' \in \mathcal{G}'$, there exists a chain of cells $(\mathscr{G}{=})\mathscr{G}_1 = \mathscr{G}_2 = \cdots = g_k(=g')$ s.t. $g_i N g_{i+1}$ for each $g_i \in \mathcal{G}'$ and $i \in [1, k)$.

To construct regions, a naïve method is that we generate all cell pairs from the set of cells and then compute the connection support of each cell pair by checking other cells in $\mathcal{G}$. We then verify whether the connection supports of cell pairs satisfy the given minimum connection support C to construct regions. However, the time complexity of the method is costly. In this paper, we propose an efficient algorithm to construct regions.

The proposed algorithm utilizes an index structure presented as follows. Given a cell length and an uncertain trajectory dataset D, we build up a *grid index* in which each cell $\mathscr{G}$ has a unique GID, a value $c(g) = |\{\text{tra}|g \in \text{tra}, \text{tra} \in D\}|$, and a corresponding trajectory list. Note that . In the grid index, each GID indexes a list of trajectories that records which uncertain trajectories traverse the cell and which points of these uncertain trajectories locate in the cell by TIDs and PIDs, respectively. To improve the efficiency of the region construction, the trajectories in



(1,1)	(2,1)	(3,1)	(4,1)
(1,2)	(2,2)	(3,2)	(4,2)
(1,3)	(2,3)	(3,3)	(4,3)
(1,4)	(2,4)	(3,4)	(4,4)

GID	c(g)
(1,4)	3

TID	PID
tra_3	1
tra_5	1
tra_1	1

TID	Sequence of GIDs	m_{tra}
tra_3	(1,4)(1,3)(3,2)(4,1)	2

Fig. 4.6 An example of an index structure

a cell's corresponding trajectory list are sorted by m_{tra}, where m_{tra} is the median of $\{c(p \cdot g) | p \in \text{tra}\}$ with given tra.

For instance, given a cell length l and an uncertain trajectory dataset $D = \{\text{tra}_1, \text{tra}_2, \text{tra}_3, \text{tra}_4, \text{tra}_5\}$, Fig. 4.6 shows an example of an index structure. Given a cell (1,4), $c(1,4) = 3$ since three distinct uncertain trajectories (i.e., tra_1, tra_3, tra_5) traverse cell (1,4). The trajectory list of cell (1,4) records these TIDs (i.e., tra_1, tra_3, tra_5) and the corresponding PID of each trajectory. The corresponding PID of tra_1 is 1 since the point of tra_1 that locates in cell (1,4) is the first point of tra_1. As shown in Fig. 4.6, tra_3 traverses four cells (i.e., (1,4), (1,3), (3,2), and (4,1)), and we can calculate that $c(1,4) = 3$, $c(1,3) = 2$, $c(3,2) = 2$, and $c(4,1) = 2$. The median among $\{2, 2, 2, 3\}$ is 2 and thus $m_{\text{tra}3} = 2$.

Before constructing regions, we let $\mathcal{G} = \mathcal{G} - \mathcal{G}'$, where $\mathcal{G}' = \{\mathscr{G} | c(g) = 0, g \in \mathcal{G}\}$. The algorithm of region construction is detailed in Algorithm 1. Note that the term *enclosed* is defined as follows.

Definition 4.6 (*Enclosed*) Given a set of cells $\mathcal{G}$ and a cell $\mathscr{G} \in \mathcal{G}$, the cell $\mathscr{G}$ is *enclosed* if there exists a region $r \subseteq \mathcal{G}\ s \cdot t.\ \mathscr{G}, \mathscr{G}' \in r, \forall g' \in \{g' | g'$ and g are spatially close, $g' \in \mathcal{G}\}$.

In Algorithm 1, we iteratively merge cells to form regions by calculating the connection supports of cell pairs. To efficiently construct regions, we determine an order for the calculation of connection supports of cell pairs according to $c(g)$. Once a cell is chosen, we iteratively pick a trajectory from the cell's trajectory list in Step 4. We then calculate the connection supports of the cell pairs around the points of the trajectory and merge qualified cells from Step 5 to Step 18. An example of this procedure is illustrated in Fig. 4.7. Let a chosen cell be g. Assume the trajectory tra_1 traversed it and $p_2^1 \cdot g = p_4^1 \cdot g = g$ (i.e., $\tau(g) = \{p_2^1, p_4^1\}$). In Step 6, we pick a point (e.g., p_1^1) from but not the point is not in $\tau(g)$. If the cell that the point locates in is not enclosed, the cell would be possibly merged with other cells. The connection supports of the cell pairs of $p_1^1 \cdot g$ and each cell around $p_1^1 \cdot g$ are calculated and the

Algorithm 1: Region Construction

```
Input: An uncertain trajectory dataset $D$, a set of cells $\mathcal{G}$, a temporal constraint $\theta$, and a minimum connection support $C$.
Output: A set of regions $R$.
1. $\mathcal{G}' \leftarrow$ Sort cells in $\mathcal{G}$ in a decreasing order of $c(g)$;
2. Do
3.     $g \leftarrow$ Pop the cell from $\mathcal{G}'$;
4.     Foreach $tra$ traversing $g$ by the order stored in the grid index
5.         $\tau(g) \leftarrow \{p | p.g = g \; and \; p \in tra\}$;
6.         Foreach $p \in tra - \tau(g)$ and $p.g$ is not enclosed
7.             If $p.g$ is contained in some region
8.                 $r \leftarrow$ The region contains $p.g$;
9.             Else
10.                $r \leftarrow \emptyset$;
11.            If $p$ is before $p'$ for all $p' \in \tau(g)$
12.                $r \leftarrow$ CM$(r, p.g, g, \theta, C, Rule1)$;
13.            ElseIf $p$ is after $p'$ for all $p' \in \tau(g)$
14.                $r \leftarrow$ CM$(r, p.g, g, \theta, C, Rule2)$;
15.            Else
16.                $r \leftarrow$ CM$(r, p.g, g, \theta, C, Rule1)$;
17.                $r \leftarrow$ CM$(r, p.g, g, \theta, C, Rule2)$;
18.            $R \leftarrow R \cup \{r\}$;
19. Until $\mathcal{G}'$ is empty or each cell in $\mathcal{G}$ is in some $r \in R$;
20. Return $R$;
```

Algorithm 2: Cell Merging (CM)

```
Input: A region $r$, a cell $p.g$, a cell $g$, a temporal constraint $\theta$, a minimum connection support $C$, and an indicator $I$
Output: A region $r$
1. Let $p.g = (x, y)$;
2. Foreach cell $g' = (x', y') \in \mathcal{G} - r$ where $|x' - x| \leq 1$ and $|y' - y| \leq 1$
3.     If $I$ is Rule1
4.         Verify whether $p.g \; N g'$ is held with given $g$ by rule 1;
5.     Else
6.         Verify whether $p.g \; N g'$ is held with given $g$ by rule 2;
7.     If $p.g \; N g'$
8.         If $r = \emptyset$
9.             $r \leftarrow \{p.g\}$;
10.        If $g'$ is contained in some region
11.            $r' \leftarrow$ The region contains $g'$;
12.        Else
13.            $r' \leftarrow \emptyset$
14.        $r' \leftarrow$ CM$(r', g', g, \theta, C, I)$;
15.        $r \leftarrow r \cup r'$;
16. Return $r$;
```

qualified cell pairs will be merged. Based on $p_1^1 \cdot g$, a region (e.g., the blue cells in Fig. 4.7) is generated in the first round and more cells (e.g., the red cells in Fig. 4.7) are merged into the region around in the second round (i.e., Step 14 in Algorithm 2).

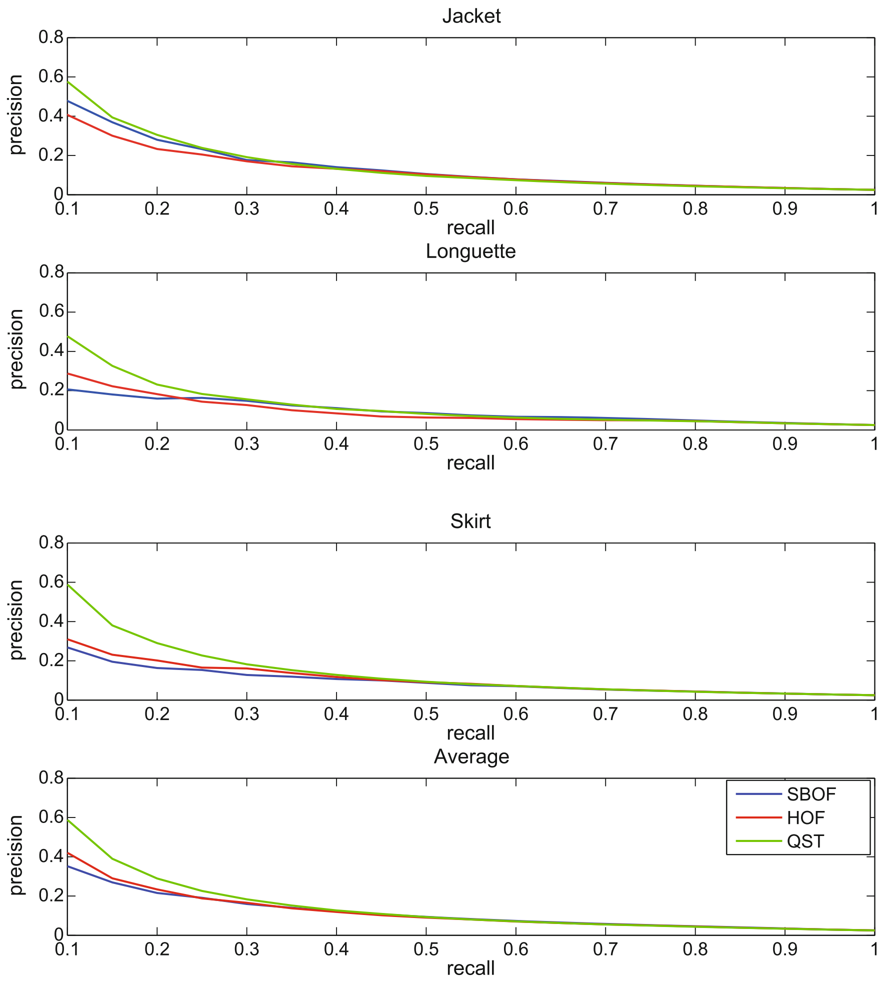


Fig. 4.7 Region construction process

Similarly, for $p_1^1 \cdot g$, a merging process will be stopped if no cell can be merged around $p_1^1 \cdot g$. We then chose other points (e.g., p_3^1, p_5^1) to construct regions around these points in the same way.

Time complexity analysis: Given an uncertain trajectory dataset D and a set of cells $\mathcal{G}$, the time complexity of the naïve method is $o(\mathrm{mn}^3)$ where $|\mathcal{G}| = n$ and $|D| = m$. Similarly, the time complexity of Algorithm 1 is $o(n(\log n+\mathrm{cm}^2))$, where c is the minimal number of the first loop. For Step 1 in Algorithm 1, it costs to sort cells in decreasing order of $c(g)$. In addition, the time complexity of Algorithm 2 is $o(m)$ because there are at most m uncertain trajectories for counting the connection support of a cell pair. Thus the time complexity of the remaining steps in Algorithm 1 is $o(\mathrm{cn\,m}^2)$.

4.3.2 Edge Inference

Once the regions are generated, we then infer edges and derive edge information including moving directions, transition supports, and travel times from historical uncertain trajectories. To generate the edges of a routable graph, we infer the edges within each region, and then infer the edges among regions.

A routable graph is a directed graph $G = (V, E)$ where V is a set of vertices and E is a set of edges. Each vertex represents a geographical area, i.e., a cell. Each directed edge e indicates a transition relationship and has two attributes, *the transition support* $e{\cdot}s$ and *the travel time* $e{\cdot}t$. To derive the transition support of an edge, we record which distinct uncertain trajectories traverse the edge. In other words, an edge has an uncertain trajectory list to record which distinct uncertain trajectories traverse it.

According to the definition of a region, a region is composed of connected cells, and thus we first generate virtual bidirected edges between cells if the cells are neighbors in a region, To infer edges' realistic directions, transition supports, and travel times, we propose *a shortest path based inference approach.*

Given a region and an uncertain trajectory dataset, we utilize the uncertain trajectories traversing the region to derive edge information in the region. For each trajectory traversing the region, we infer the shortest path between any two consecutive points of the trajectory by virtual bidirected edges in the region. We illustrate edge inference in a region in Fig. 4.8. As shown in Fig. 4.8a, four uncertain trajectories pass through the region. For instance, in Fig. 4.8b, an uncertain trajectory $p_1 \rightarrow p_2 \rightarrow p_3$ (blue squares) traverses the region, and we infer the shortest paths from p_1 to p_2 and the shortest paths from p_2 to p_3. As shown in Fig. 4.8b, we find one shortest path from p_1 to p_2, and two shortest paths from p_2 to p_3. After finding the shortest path between two consecutive locations, we divide the travel time evenly and add it to the travel time list of each edge in the shortest path. In addition, each edge of the shortest path adds the trajectory ID into its corresponding trajectory list, and the transition support of each edge in the shortest path is accumulated one. If there are multiple shortest paths between two consecutive locations, we similarly update the information of each edge in these paths.

By using historical uncertain trajectories to infer edge information in a region, we further eliminate the redundant edges in the region and the edges whose transition supports are 0. Given an edge e_1 from the cell g and the cell g', where g and g' are spatially close, the edge e_1 is *redundant* in a region if there exist an edge e_2 from it

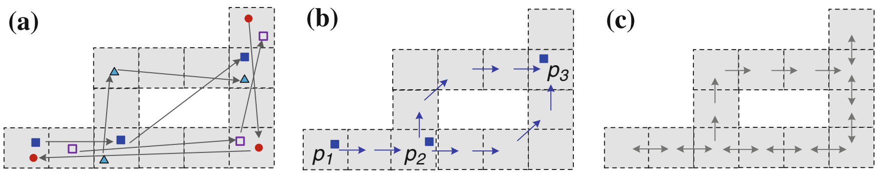


Fig. 4.8 Edge inference in a region

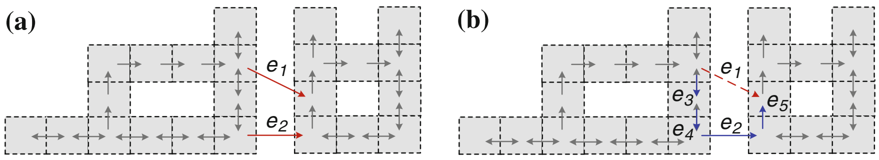


Fig. 4.9 Edge inference between regions

g to g'' and an edge e_3 from g'' to g', such that (1) g and g'' are spatially close and g' and g'' are spatially close, (2) $\frac{e_2 \cdot s e_3 \cdot s}{2} > e_1 \cdot s$, and (3) $\frac{|e_2 \cdot t + e_3 \cdot t - e_1 \cdot t|}{\max\{(e_2 \cdot t + e_3 \cdot t), e_1 \cdot t\}} \leq \theta$ where θ is a given temporal constraint. Figure 4.8c shows the inferred edges in the region after reducing edges. The travel time of each edge is estimated by the median of all the travel times of the edge.

In the following, we construct edges between regions. Similarly, we generate edges between regions by using historical uncertain trajectories first. This means that if an uncertain trajectory traverses a cell of one region to a cell of another region, an edge is constructed between the two cells. Next, we eliminate the redundant edges between regions. Given an edge e_1 from the cell g to the cell g', where g and g' are in different regions, the edge e is a *redundant* edge between the two regions if there exists an alternative route $e_i \rightarrow \cdots \rightarrow e_j$ from the cell g to the cell g' such that (1) $\frac{1}{\text{j}-\text{i}+1} \sum_{k=i}^{j} e_k \cdot s > e \cdot s$, and (2) $\frac{|\sum_{k=i}^{j} e_k \cdot t - e \cdot t|}{\max\{\sum_{k=i}^{j} e_k \cdot t, e \cdot t\}} \leq \theta$ where θ is a given temporal constraint. Figure 4.9 shows an example of edge inference between two regions. As shown in Fig. 4.9a, edge e_1 and edge e_2 are generated between the two regions by historical uncertain trajectories. For instance, in Fig. 4.9b, edge e_1 is a redundant edge if (1) $\frac{1}{4} \sum_{k=2}^{5} e_k \cdot s > e_1 \cdot s$, and (2) $\frac{|\sum_{k=2}^{5} e_k \cdot t - e_1 \cdot t|}{\max\{\sum_{k=2}^{5} e_k \cdot t, e_1 \cdot t\}} \leq \theta$ with a given θ.

Note that the transition information of an eliminated edge is propagated to alternative routes. The travel time of an eliminated edge is evenly propagated to the edges of each alternative route. The trajectory list of an eliminated edge is updated to each edge's corresponding trajectory list in alternative routes.

4.4 Route Inference

Given a location sequence and a time span, we generate the top-*k* popular routes in two phases: route generation and route refinement. In the first phase, we propose routing algorithms to search for the top-*k* coarse routes with the routable graph. We further refine the discovered top-*k* routes to effectively derive specific routes in the second phase.

In the route generation phase: we first generate possible routes between each two consecutive queried locations (called *local routes*) and then search for the top-*k* routes (called *global routes*) from the generated local routes.

A *route* derived in this phase is represented by a sequence of vertices with a given graph $G = (V, E)$. Note that a vertex in the graph represents a cell; thus a route here is regarded as a sequence of cells, denoted as $p{:}g_1 \to g_2 \to \cdots \to g_k$. Given a sequence of query locations and a time span, we search for qualified routes between any two consecutive query locations with the constructed graph. Before searching for routes with the graph, we need to specify the corresponding vertices of query locations. Since a vertex represents a geographical area of a cell, a query location can be mapped to the vertex whose corresponding geographical area overlaps the location. However, it is possible that a query location cannot be mapped to any vertex in the graph. We further select the vertices whose corresponding cells are close to the query location. We adopt the minimum distance (MINDIST) [11] to formulate the distance between a query location and a cell. According to the distance measurement, we specify the cells that are close to such query locations. Thus, a given location sequence is transformed into a sequence of sets of cells. Moreover, we transform a location sequence into cell sequences by combining these cells. After query transformation, we search for the top-k routes according to each cell sequence.

For instance, given a location sequence $q{:}q_1 \to q_2 \to q_3$ in Fig. 4.10a, the locations q_1 and q_2 are mapped to cells g_1 and g_2, respectively. By the minimum distance measurement, the set of cells $\{g_3, g_4\}$ is used to represent the location q_3. Then, the location sequence $q_1 \to q_2 \to q_3$ is transformed into two cell sequences, i.e., $g_1 \to g_2 \to g_3$ and $g_1 \to g_2 \to g_4$.

We generate routes with respect to each cell sequence. Before introducing the routing algorithm, we define the score function for the routes as follows.

Definition 4.7 (*Route score*) Given a graph $G = (V, E)$, a route $\mathscr{P}{:}\mathscr{P}_1 \to \mathscr{P}_2 \to \cdots \to \mathscr{P}_m$, where $\mathscr{P}_i{:}g_{i_1} \to g_{i_2} \to \cdots \to g_{i_j}$, the score of the route is defined as $f(\mathscr{P}) = \sum_{i=1}^{m} \rho(\mathscr{P}_i)$, where $\rho(\mathscr{P}_i) = \frac{1}{j-1}|\cup_{k=1}^{j-1}\{\text{tra}|g_{ik} \to g_{i_{k+1}} \in \text{tra}\}|$.

For each cell sequence, we first search for the top-k local routes between any two consecutive cells in the cell sequence (e.g., g and g') by an A*-like routing algorithm. Furthermore, a possible maximum speed could be derived from historical uncertain trajectories or be determined by difference applications. Given a maximum speed, possible positions between any two consecutive query locations can be restricted in a range if a time interval between the two locations is specified [10]. That is, the possible routes are restricted in the cells overlapping the range. For the A*-like routing algorithm, an estimated score of a route from a cell g to a cell g' is represented as follows.

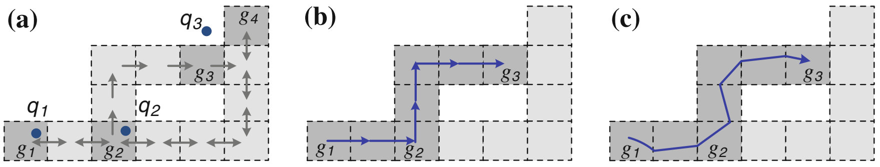


Fig. 4.10 Route inference. **a** Query transformation. **b** Route generation. **c** Route refinement

Given two cells g and g', a current visited cell g'', and a specified range r, an estimated score of a route $\mathscr{P}$:$\mathscr{P}_c \rightarrow \mathscr{P}_f$ from a cell g to a cell g' in a specified range r is $\hat{f}(\mathscr{P}) = \rho(\mathscr{P}_c) + h(\mathscr{P}_f)$, where $\mathscr{P}_c$ is a known route from cell g to cell g'', and $h(\mathscr{P}_f)$ is the score of an estimated route $\mathscr{P}_f$ from cell g'' to cell g'.

Definition 4.8 (*Optimal score*) Given a graph $G = (V, E)$, an uncertain trajectory dataset D, a specified range r, and two cells g and g', the optimal score of the routes from a cell g to a cell g' in r is defined as

$$\hat{h}(\mathscr{P}) = |\{\text{tra}|\text{tra passes through the range } r, \text{tra} \in D\}|$$

for some estimated route $\mathscr{P}$ from a cell g to a cell g'.

Once a local route is generated from the cell g to the cell g' and satisfies the given time span, the score of the local route is calculated. If there are more than k local routes constructed from the cell g to the cell g', the k-th maximum score of these local routes is recorded and incrementally updated. Based on the estimated optimal score function, a branch of searching routes will be stopped if the optimal score of routes generated from the branch is less than the updated k-th maximum score.

Based on the top-k local routes between any two consecutive cells of each cell sequence, we search for the top-k global routes by a branch-and-bound search approach. For instance, given a cell sequence $g_1 \rightarrow g_2 \rightarrow g_3$ in Fig. 4.10a, a global route is derived as a sequence of cells (dark grey) in Fig. 4.10b. To derive a specific route, a route is further transformed into a line by concatenating the centers of any two consecutive cells in the route. Figure 4.10b shows an example of such a route by a blue line. However, it is possible that we search for local routes between two cells belonging to different regions. In the A*-like algorithm, although we search for local routes between two given cells in a restricted range of a graph, the search space is still large if the distance between two given cells is far and they are in different regions. A route between the two cells would possibly pass through several other regions. On the other hand, a lower bound of transition times between any two regions can be estimated by edge information. It helps us stop searching for routes between two regions if the lower bound of transition time between the two regions exceeds the time span. Hence, to improve the efficiency of route generation, we modify the proposed A*-like routing algorithm and introduce a two-layer routing algorithm.

Before searching for local routes between two given cells, we first determine the region sequences to reduce searching space. By utilizing a lower bound of transition times between any two regions, we can generate region sequences with respect to two given cells. According to each region sequence, we search for possible local routes that sequentially traverse these regions. Note that the proposed A*-like algorithm is used for searching for routes between any two regions here. In Fig. 4.11, for instance, given a location sequence $q_1 \rightarrow q_2$, and a corresponding range r, the location q_1 and the location q_2 locate in region R_1 and region R_4, respectively. There are multiple possible region combinations for searching for routes between the two locations.

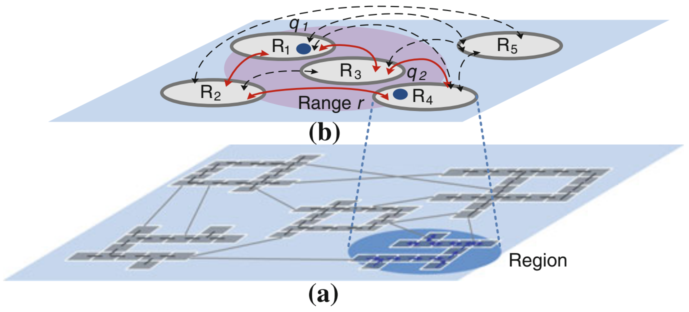


Fig. 4.11 The scenario of the two-layer routing algorithm

Although the searching space of route generation is restricted to the range r (e.g., the red part), the routes from q_1 to q_2 would possibly traverse the regions in different orders. In Fig. 4.11, there are many possible region sequences from q_1 to q_2 (e.g., $R_1 \rightarrow R_4$, $R_1 \rightarrow R_2 \rightarrow R_4$, $R_1 \rightarrow R_3 \rightarrow R_4$, $R_1 \rightarrow R_2 \rightarrow R_3 \rightarrow R_4$ etc.). By utilizing lower bounds of transition times between regions, the possible region sequences would be reduced to satisfy the given time span. For instance, the qualified region sequences are marked by red edges in Fig. 4.11 (i.e., $R_1 \rightarrow R_2 \rightarrow R_4$, $R_1 \rightarrow R_3 \rightarrow R_4$). After deriving region sequences, we search for possible routes which traverse each region sequence.

After route generation, the top-k routes are inferred, and we further refine each route using historical data points. Route refinement has three steps: data point selection, segment formulation, and segment concatenation. First, given an inferred rough route represented by a sequence of cells, we select the historical uncertain trajectories that traverse the cells in the same order as the route. Next, we extract the data points that locate in cells of the rough route from these selected uncertain trajectories, and thus derive a set of points for each cell of the route. To formulate a specific route from selected points, we adopt linear regression for the set of points of each cell to derive a segment. We then concatenate the segments in the same order as an original inferred route. Figure 4.10c shows an example of a refined route.

4.5 Performance Evaluation

In this section, we evaluate the performance of the proposed RICK using real datasets, including check-in records from Foursquare and taxi trajectories. The datasets and experimental setting is presented in Sect. 5.1. In the experiments, we first demonstrate the results using check-in records in Manhattan. To evaluate the effectiveness of our proposed RICK, we use the dataset of taxi trajectories. In the experiments of the performance study, we compare our proposed RICK with the existing method in terms of effectiveness and efficiency. Furthermore, the experiments demonstrate the improvements of routable graph construction and route inference.

4.5.1 Datasets and Settings

4.5.1.1 Real Datasets

In this paper, we use two real datasets to conduct the extensive experiments. One is the check-in dataset from Foursquare. We collected check-in records in Manhattan, and for each user, a series of check-in records recorded in one day is regarded as a trajectory. We pruned the trajectories that contained less than three check-in records. There are totally 6,600 trajectories. The other real dataset contained 15,000 taxi trajectories in Beijing. The average sampling rate of the raw trajectories is less than one minute. To simulate uncertain trajectories, we resampled each raw trajectory such that the time interval between two consecutive resampled points of the trajectory at least exceeded a given sampling rate S. In the experiments, the sampling rate S is set from one minute to five minutes and the default S is five minutes. For example, given $S = 5$, the time interval between two consecutive resampled points is at least five minutes or even more.

4.5.1.2 Metrics

To evaluate the effectiveness of our RICK, we introduce an approach to generate the ground-truth from the raw trajectories to evaluate the effectiveness of the inferred routes. For each query, the raw trajectories that satisfy the query are selected and ranked. To rank these trajectories, a raw trajectory is transformed into a sequence of road segments and the frequency of a road segment is defined as the number of distinct trajectories that traverse it. The score of a transformed trajectory tra:$r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_n$ is defined by $(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{d}(r_i))/\mathrm{tra} \cdot \mathrm{length}$, where r_i is a road segment and $\mathrm{d}(r_i)$ is the frequency of the road segment r_i. Hence, the selected trajectories can be ranked by their scores.

To evaluate the difference between an inferred trajectory and a raw trajectory of the ground-truth, we first apply the length-normalized dynamic time warping distance (NDTW). Given an inferred route p and a raw trajectory tra, we define the NDTW between two trajectories as $\mathrm{NDTW}(p, \mathrm{tra}) = \mathrm{DTW}(p, \mathrm{tra})/p \cdot \mathrm{length}$ for an optimal alignment path. To further reflect the quality of inferred routes, we utilize a maximum distance (MD) between an inferred route and a raw trajectory of the ground-truth according to the discovered NDTW. MD is defined as the maximum value of the distances measured by the optimal alignment path. Therefore, the two measurements for evaluating the inferred top-k routes are defined as follows:

$$\mathrm{NDTW}(\mathrm{T}, \mathrm{T}') = \mathrm{Avg}_{p_k \in \mathrm{T}} \mathrm{min}_{\mathrm{tra} \in \mathrm{T}'} \mathrm{NDTW}(p_k, \mathrm{tra}), \text{ and}$$
$$\mathrm{MD}(\mathrm{T}, \mathrm{T}') = \mathrm{Avg}_{p_k \in \mathrm{T}} \mathrm{MD}(p_k, \mathrm{tra}')$$

where T is the set of inferred top-k routes, T' is the set of top-k raw trajectories, and $\mathrm{tra}' = \mathrm{Arg\,min}_{\mathrm{tra} \in \mathrm{T}'} \mathrm{NDTW}(p_k, \mathrm{tra})$.

In the experiments, the default rank threshold k is 3.

4.5.2 Visualization of Results

In this subsection, we use the check-in dataset in Manhattan to visualize the results derived by RICK. We first demonstrate the constructed routable graph in Fig. 4.12 with given cell length $l = 500$ (m), temporal constraint $\theta = 0.2$ and minimum connection support $C = 3$. In Fig. 4.12, the regions are represented by different colors in Fig. 4.12a, b shows the edges between cells. Note that the edges within a region are drawn by blue lines, and the edges between regions are drawn by black lines. Based on the routable graph, we perform one query and let the span time be one hour for each query. Given a query as "Central Park → The Museum of Modern Art → Times Square → Empire State Building → SoHo", the top-1 route inferred by RICK is depicted in Fig. 4.12c. As shown in Fig. 4.12c, the route does not simply connect the query locations, but passes through other attractions. For example, for the partial route from "The Museum of Modern Art" to "Times Square", RICK constructs this partial route to pass by the "Rockefeller Center" based on users' historical check-in records.

4.5.3 Performance Study

In this section, we evaluate the performance of RICK by taxi trajectories. First, to analyze the effect of queries, the length of query location sequence $|q|$ is set from 2 to 4. In addition, a query location sequence is generated by considering a given distance between any two consecutive query locations, denoted as Δd. For a query, Δt is determined according to Δd. In the experiments, Δd is varied from 1 to 5 (in kilometers), and the corresponding Δt is set from 4 to 20 (in minutes). For each experiment, we perform almost 100 queries and averaged the results.

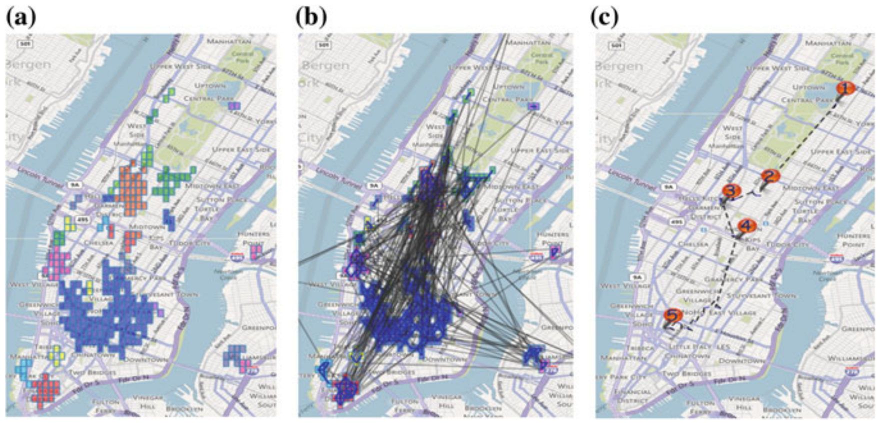


Fig. 4.12 Visualization of results in Manhattan. **a** Regions. **b** Routable graph. **c** Top-1 route

4.5.3.1 Evaluation of Route Inference

We compare our framework with a baseline and analyze the effectiveness of our inferred routes in different aspects.

Baseline: To evaluate the effectiveness of the discovered routes, we compare the proposed RICK with the existing approach (MPR) in [2]. In [2], given two locations (i.e., $|q| = 2$), the most popular rout, which connects the two query locations, is derived. In the experiments, the parameters of MPR are set as $\alpha = 2$, $\beta = 2$, the coherence threshold $\tau = 0.8$, and the cluster size threshold $\varphi = 20$. For RICK, the settings are $l = 300$ (m) and $k = 1$. Figure 4.13 shows the experimental results of MPR and RICK under the Taxi dataset with S and Δd varied. As shown in Fig. 4.13a, the error of MPR increases as S or Δd increases. It is worth mentioning that the error of RICK slightly increases as S or Δd increases, showing that RICK is able to derive the routes from uncertain trajectories. Figure 4.13a shows that RICK is more effective than MPR, although Fig. 4.13b demonstrates that the query time of RICK is slightly higher than the query time of MPR.
Effect of route refinement: In the route inference of the proposed RICK, the top-k routes are derived after route generation and are further refined by route refinement. In this subsection, we compare the effectiveness of the route inference without route refinement (w/o RR) and that of the route inference with route refinement. We set $k = 1$ and $|q| = 2$ in the experiments. Figure 4.14 shows the error of top-1 routes by NDTW and MD. As shown in Fig. 4.14, the errors of inferred routes increase as Δd increases. In addition, a larger l increases the error of routes discovered without route refinement. In Fig. 4.14, with route refinement, the error of the inferred routes is obviously reduced as l increases.

Impact of data sparseness: To study the effect of the data sparseness, we calculate the number of GPS points per km^2 and derive different data sparseness by setting different S. The number of GPS points per km^2 is increased from 77 to 275 while S is decreased from five minutes to one minute. Figure 4.15 shows that the errors (both NDTW and MD) slightly decrease as the data sparseness increases. When the data

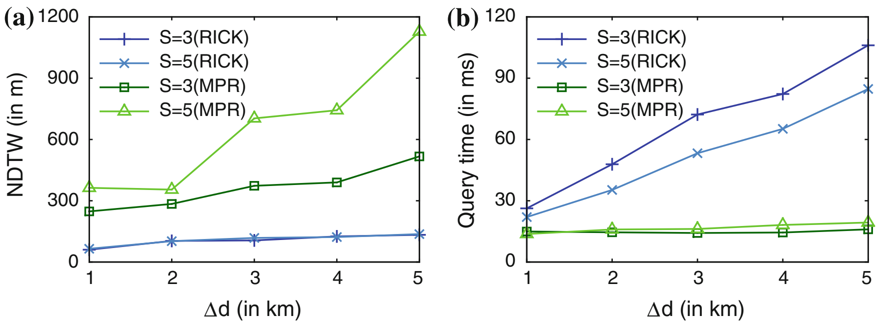


Fig. 4.13 Performance comparison of RICK and MPR

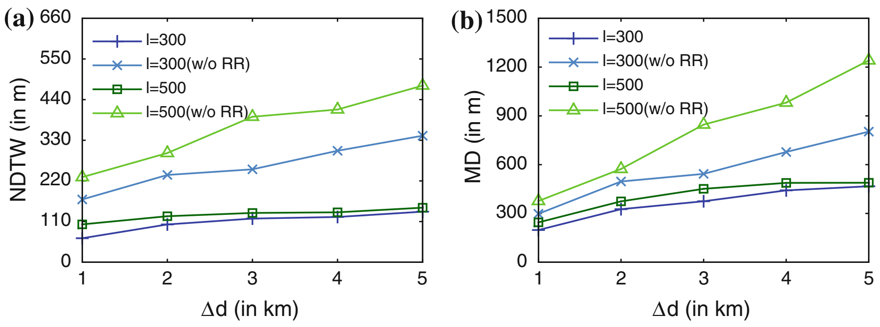


Fig. 4.14 Effect of route refinement

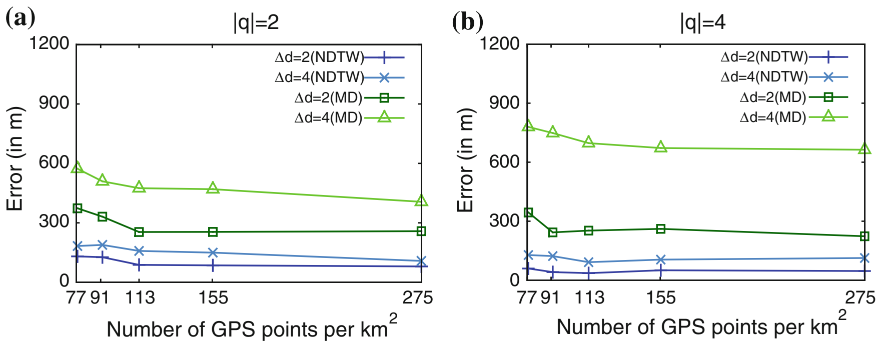


Fig. 4.15 Effectiveness evaluation with data sparseness varied

sparseness is 275 GPS points per km^2, the errors of the inferred routes of at least 4 km (i.e., $|q| = 2$ and $\Delta d = 4$) are less than 500 m and the errors of the inferred routes of at least 12 km (i.e., $|q| = 4$ and $\Delta d = 4$) are less than 800 m. However, NDTW is less than 300 m even though the data sparseness is 77 GPS points per km^2. The proposed framework is effective for inferring the top-k routes.

Efficiency: We investigated the query time of RICK and show the results in Fig. 4.16. In the experiments, $l = 300$ (meters), $\theta = 0.1$, $C = 8$, $S = 5$ (minutes), and $k = 3$. In the route inference, we improve the efficiency of the route generation by a two-layer routing algorithm. To demonstrate the effectiveness of the two-layer routing algorithm, we compare the query time of RICK and the query time of RICK without using the two-layer routing algorithm (denoted by RICK-) in Fig. 4.16a with varied $|q|$ and Δd. As shown in Fig. 4.16a, RICK outperforms RICK-, and the query time is obviously reduced while $|q|$ or Δd is larger. In Fig. 4.16b, the query time of RICK gradually increases as $|q|$ or Δd increases. However, the query time is less than one second.

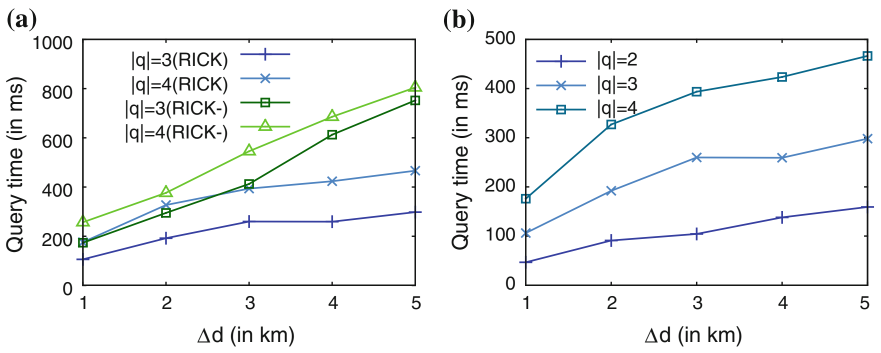


Fig. 4.16 Efficiency evaluation

4.5.3.2 Evaluation of Routable Graph

In the routable graph construction of RICK, we construct the regions referring to the connected areas and further infer and refine the moving directions within the regions. To investigate the impact of exploring shortest path on refining the routable graph, we evaluate the graph built without refinement (denoted as RG), and the graph refined by shortest path based edge inference (denoted as RG+).

To evaluate the correctness of the connectivity in a routable graph, given a raw trajectory dataset D and a graph $G = <V, E>$, the precision of connectivity in G is measured as follows:

$$|\{e|e \text{ is traversed by some tra} \in D \text{ and } e \in E\}|/|E|.$$

The temporal constraint θ and the minimum connection support C are used for constructing a routable graph. Hence, we analyze the precision of connectivity in the graph with varying θ and C.

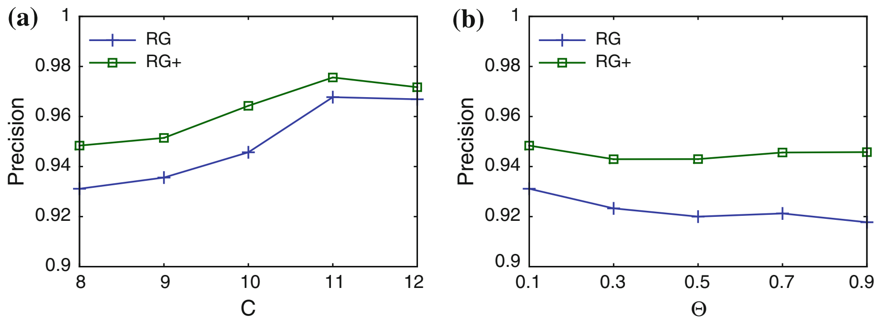


Fig. 4.17 Connectivity evaluation

In Fig. 4.17, S is set as 5 min in the experiments. In Fig. 4.17a, C is varied from 8 to 12 and $\theta = 0.1$. Figure 4.17a shows that the precision of RG and the precision of RG+ increase as C increases. This is because a stricter constraint induces a higher precision (i.e., a higher C). In Fig. 4.17b, $C = 8$ and θ is varied from 0.1 to 0.9. As shown in Fig. 4.17b, the precision of RG and the precision of RG+ decrease as θ increases. The reason is that the precision is reduced as the constraint is loosened (i.e., a higher θ). Figure 4.17 depicts that the precision of RG+ is higher than that of RG, and it demonstrates that the shortest path based edge inference improves the correctness of the explored connectivity in a geographic space.

4.6 Related Work

Route planning based on GPS trajectories: Route planning is widely investigated in [2, 3, 13, 15] with GPS trajectories. The work [15] mainly inferred fastest routes from historical trajectories. In [13], the authors study travel route planning based on searching GPS trajectories. In [3], given a set/a sequence of locations, the top-k trajectories that best connect the given locations are retrieved from existing GPS trajectories. In [2], the authors investigated the problem of popular route planning without road network information. They introduced a transfer network model by exploiting intersections from historical GPS trajectories, and inferred the most popular route between two given locations by the turning probability of each intersection. However, these works were carried out using high sampled GPS trajectories. Given uncertain trajectories, the results obtained by [3, 13] are historical uncertain trajectories and these uncertain trajectories still reveal rough routes. In addition, the trajectories derived by [3] may be far away from the query locations because these trajectories are low sampled. Using a dataset of uncertain trajectories, the accuracy of a transfer network model in [2] would be destroyed and then the effectiveness of inferred routes would be decreased.

Trip Planning based on geo-tagged social media: In recent years, mobile social applications have become popular, generating a huge volume of social media data, such as check-in records or geo-tagged photos. Such social media data can be regarded as sequences of visited locations, thereby revealing users' travel experience in terms of travel routes that link points-of-interest (POIs). Using geo-tagged photos, several studies [6, 12, 14] have investigated the problem of trip planning. However, the recommended trips are represented by a sequence of POIs, and the detailed route between two consecutive POIs is not specified. Different from these works, our method aims to construct the detailed route that is most likely to be taken by people by learning from the uncertain POI sequences in a mutual reinforcement way (e.g., Fig. 4.2).

Uncertain trajectories: The research topics of trajectory uncertainty are studied in [5, 9, 10, 17]. The work [9] introduces the problem of uncertain trajectory clustering, and focuses on the trajectory uncertainty caused by measurement errors. To reduce the uncertainty of an uncertain trajectory, the work [10] formulates an uncertain trajectory in a free space by a given maximum moving speed. However, the indistinct parts of an uncertain trajectory are enclosed in a spatio-temporal range without pointing out specific routes. In addition, the study [5] applies the techniques developed in a free space to model an uncertain trajectory in a road network. The possible routes between two sampled locations of an uncertain trajectory are restricted in a set of road segments by road network information and speed limits. Although the work [17] investigated the problem of discovering the top-*k* possible routes sequentially passing the queried locations from uncertain trajectories, they use road network information to reduce the uncertainty of low sampled trajectories. These works cannot derive routes from uncertain trajectories without road network information.

4.7 Conclusions

In this paper, we proposed RICK to infer the top-*k* routes traversing a given location sequence within a specified travel time from uncertain trajectories. The proposed RICK consists of the routable graph construction and the route inference. We have evaluated the proposed RICK in terms of both effectiveness and efficiency using two real datasets, check-in datasets and driving trajectories. The experiments show three aspects: (1) the inferred routes not only connect user-specified locations but also indicate detailed routes; (2) the proposed routable graph provides a good model of the uncertain trajectory dataset with an accuracy of 0.9; (3) on average, our routing algorithm can find the top-3 routes within 0.5 seconds, with a distance error smaller than 300 meters compared to its corresponding ground-truth. Meanwhile, RICK clearly outperforms the baseline by generating routes 300–700 meters closer (than those of the baseline) to the ground-truth. The experiments demonstrate the effectiveness and the efficiency of RICK. In the future, we will plan routes considering different start times and different user preferences. In addition, we will evaluate RICK by given other uncertain trajectory datasets, e.g., geo-tagged photo trips.

References

1. Cao X, Cong G, Jensen CS (2010) Mining significant semantic locations from GPS data. VLDB 3(1):1009–1020
2. Chen Z, Shen HT, Zhou X (2011) Discovering popular routes from trajectories. In: IEEE ICDE, pp 900–911
3. Chen Z, Shen HT, Zhou X, Zheng Y, Xie X (2010) Searching trajectories by locations: an efficiency study. In: ACM SIGMOD, pp 255–266
4. Giannotti F, Nanni M, Pedreschi D, Pinelli F (2007) Trajectory pattern mining. In: ACM SIGKDD, pp 330–339
5. Kuijpers B, Moelans B, Othman W, Vaisman A (2009) Analyzing trajectories using uncertainty and background information. In: SSDT, pp 135–152
6. Kurashima T, Iwata T, Irie G, Fujimura K (2010) Travel route recommendation using geotags in photo sharing sites. In: ACM CIKM, pp 579–588

7. Li Z, Ding B, Han J, Kays R, Nye P (2010) Mining periodic behaviors for moving objects. In: ACM SIGKDD, pp 1099–1108
8. Liu Q, Ge Y, Li Z, Chen E, Xiong H (2011) Personalized travel package recommendation. In: IEEE ICDM, pp 407–416
9. Pelekis N, Kopanakis I, Kotsifakos EE, Frentzos E, Theodoridis Y (2009) Clustering trajectories of moving objects in an uncertain world. In: IEEE ICDM, pp 417–427
10. Praing R, Schneider M (2007) Modeling historical and future movements of spatio-temporal objects in moving objects databases. In: ACM CIKM, pp 183–192
11. Roussopoulos N, Kelley S, Vincent F (1995) Nearest neighbor queries. In: ACM SIGMOD, pp 71–79
12. Roy SB, Amer-Yahia S, Das G, Yu C (2011) Interactive itinerary planning. In: IEEE ICDE, pp 15–26
13. Wei LY, Peng WC, Lee WC (2013) Exploring pattern-aware travel routes for trajectory search. ACM TIST 4(3):48
14. Yin Z, Cao L, Han J, Luo J, Huang TS (2011) Diversified trajectory pattern ranking in geo-tagged social media. In: SDM, pp 980–991
15. Yuan J, Zheng Y, Zhang C, Xie W, Xie X, Sun G, Huang Y (2010) T-drive: driving directions based on taxi trajectories. In: ACM SIGSPATIAL, GIS, pp 99–108
16. Zheng Y, Zhang L, Xie X, Ma WY (2009) Mining interesting locations and travel sequences from GPS trajectories. In: WWW, pp 791–800
17. Zheng K, Zheng Y, Xie X, Zhou X (2012) Reducing uncertainty of low-sampling-rate trajectories. In: IEEE ICDE, pp 1144–1155
18. Zheng Y, Zhou X (eds) (2011) Computing with spatial trajectories. Springer, New York

Chapter 5
Social Interactions over Location-Aware Multimedia Systems

Yi Yu, Roger Zimmermann and Suhua Tang

Abstract Advancements in positioning techniques and mobile communications have enabled location-based services with a broad range of location-aware multimedia applications. Accordingly, various social multimedia data, relevant to different aspects of users' daily life, is aggregated over time on the Internet. Such location-aware multimedia data contains rich context of users and has two implications: individual user interest and geographic-social behaviors. Exploiting these multimedia landscapes helps mine personal preferences, geographic interests and social connections, and brings the opportunities of discovering more interesting topics. In this chapter, we first introduce some examples of location-aware multimedia data and social interaction data. Then, we report some latest methods related to context detection and location-aware multimedia applications. We further present some analysis of geo-social data. Finally, we point out the trend in the integration of social and content delivery networks. In brief, this chapter delivers a picture of emerging geographic-aware multimedia technologies and applications, with location information as a clue.

Y. Yu (✉) · R. Zimmermann
School of Computing, National University of Singapore,
Computing 1, 13 Computing Drive, Singapore, Singapore
e-mail: yuy@comp.nus.edu.sg

R. Zimmermann
e-mail: rogerz@comp.nus.edu.sg

S. Tang
Graduate School of Informatics and Engineering, The University of Electro-Communications,
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan
e-mail: shtang@uec.ac.jp

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_5

5.1 Motivation and Introduction

Conventionally, content sharing websites [1] and online social networks [2] are separately deployed. Users visit content sharing websites to upload, view, and share their multimedia contents. Users login to social networks to exchange messages and keep in contact with their friends. Recently, various online communities (e.g., Flickr, Foursquare, Facebook, Twitter) have started to provide users with location-based services [3]. In this way, users can record and upload geo-tagged images and videos to these web sites anytime and anywhere with their mobile devices. For example, many fantastic geo-tagged photos taken by users are shared at Flickr. As a result, every day a huge volume of user-generated geo-tagged multimedia data is generated in the Internet.

Location, as an extra information, is playing an important part in complementing content retrieval and recommendation [4–9]. As shown in Fig. 5.1, it also serves as an important element to connect content sharing services and online social services, which facilitates personalized, localized, and socialized multimedia content discovery, retrieval, recommendation, and diffusion across diverse user-generated multimedia datasets. In particular, registered users can check in[1] at various venues and contact their friends nearby to share experience with them. Geographic trajectories of users are associated with their preferences and can be used for personalized location recommendation [10]. Check-in information at business venues can be leveraged for

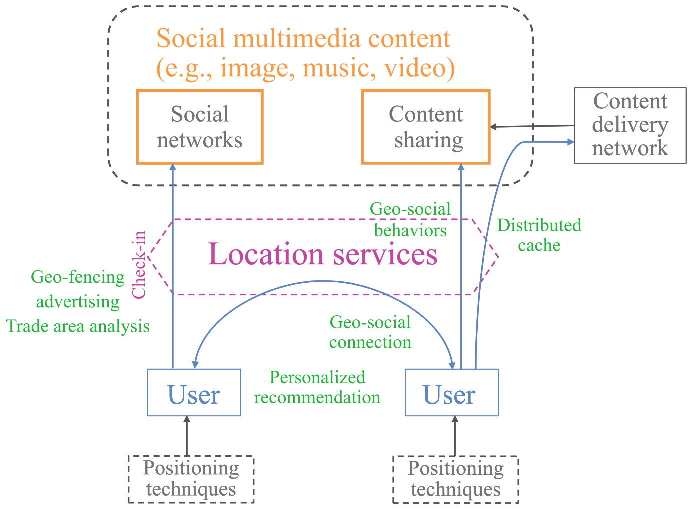


Fig. 5.1 Connecting social networks and content-sharing platforms via location information

[1] Many social networking services allow users to self-report presence (known as check in) to a physical place and share their locations with their friends. Refer to http://en.wikipedia.org/wiki/Check-in.

geo-fencing services [11–14], mobile advertising [15, 16], business analytics, and used to analyze the geo-spatial distribution of users and user social behaviors.

A new trend is the integration of social networks and content sharing platforms [4], as follows: Users share their opinions of multimedia contents or recommend multimedia contents on social networking platforms; This helps to spread multimedia contents and events all over the world through the social connections between users [17]; In addition, the geo-spatial distribution of users and social connections between users can be further exploited to optimize the distributed cache [18] of multimedia contents.

The rest of this chapter addresses different parts in Fig. 5.1. First, we introduce different location-aware media data in Sect. 5.2. Then, we show some methods related to location inference and geo-fencing in Sect. 5.4. Next, we present some location-aware multimedia applications in Sect. 5.5 and the analysis of geo-social data in Sect. 5.6. We also discuss the integration of social networks and content-sharing networks in Sect. 5.7. Finally, we conclude this chapter with Sect. 5.8.

5.2 Geo-Tagged Multimedia Data on Social Networks

Here, we introduce several typical examples of location-aware multimedia data, e.g., Flickr images, Foursquare check-in, Twitter messages. This demonstrates how user-centric location-aware datasets are associated with multimedia contents. Such location-aware social multimedia data with geo-tags can be exploited in later sections to analyze user behavior, especially user interest.

5.2.1 Geo-Tagged Photos on Flickr

Location information is important for remembering where a particular photo came from and showing off user's favorite photos to the world over a map. Online photo sharing website Flickr[2] has created the geo-tagging[3] function to let users geo-tag their photos, as shown in Fig. 5.2. According to the location names, these geo-tagged photos can be classified and displayed on a map.

Flickr acts as a repository of all kinds of photos together with geo-tags. Through crowdsourcing from Flickr's geo-tagged photo collections, geographic discovery can be studied to discover knowledge about different aspects of information on the surface of the Earth, for example, classifying the land-use into classes [19] of academic, sports and residential according to both images and their geo-tags.

[2] https://www.flickr.com/.

[3] Geo-tagging is the process of adding geographical identification metadata to various media data. Refer to http://en.wikipedia.org/wiki/Geotagging.

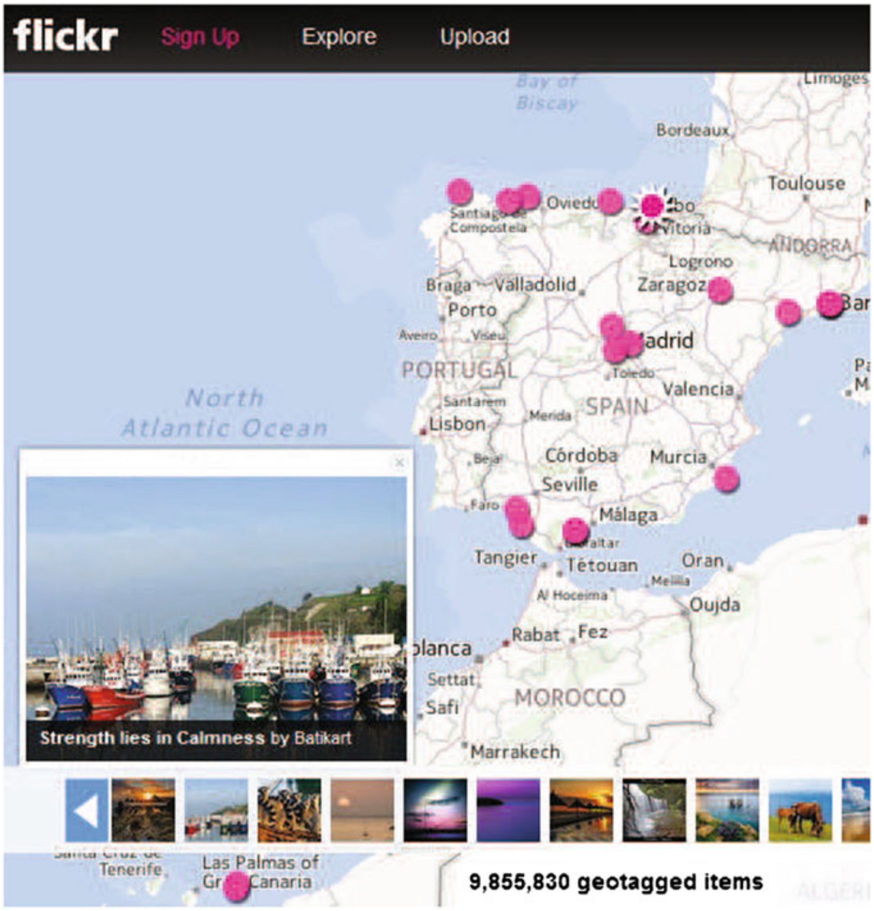


Fig. 5.2 Geo-tagged Flickr photos shown on the map

5.2.2 Geo-Social Data on Foursquare

Foursquare[4] implements a location layer for the Internet, which is an intersection of virtual social networks and physical world to help connect people with their friends around the world. In addition, Foursquare provides an API to map location information to geo-categories. Specifically, with a given location (latitude and longitude), Foursquare returns venues nearby with metadata (geo-category etc.). From Fig. 5.3, we can see top 10 check-in countries and top 10 check-in geo-categories in Foursquare.

User-generated geographic data may be shared on social networking platforms. For example, checking-in at a venue via Foursquare, Foursquare will tell you who and

[4] https://foursquare.com/.

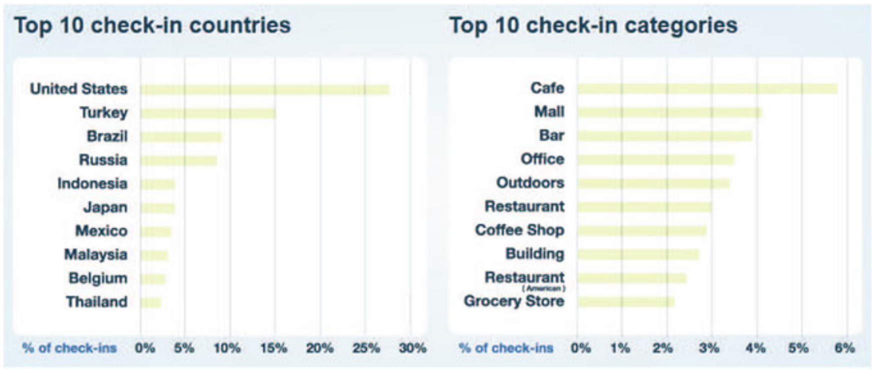


Fig. 5.3 Top check-in countries and categories in Foursquare, reported by http://gnip.com/foursquare/

what are nearby and broadcast your location to your friends and update your Twitter and Facebook status. Foursquare also can serve as a metadata of local business information. When users check-in at the stores, the check-in data provides a spatial distribution of users visiting these stores, and can be used for analyzing the primary trade areas of these stores [15]. Check-ins in Foursquare also can provide user visit information [10].

Data about the geographic positions of users can be made publicly available, together with their online social connections. For example, many Foursquare users choose to automatically push their check-ins to Twitter messages. Although Foursquare does not provide unauthorized access to user friends list, each tweet provides a URL to the Foursquare website, where information about the geographic location of the venue can be acquired. Twitter provides a public API to search and download these tweets. Then, friendship ties and location information can be acquired from tweets. These datasets are publicly available and can be used to study social and geographic networks of users.

5.2.3 Location-Aware Messages on Twitter

As music plays an important role in our life, users often tweet music-related topics on Twitter.[5] Through crowdsourcing in Twitter, tweets with geospatial coordinates can be leveraged for estimating artist similarity, popularity, and local music trends. In addition, geographic music listening pattern inferred from all music tweets can be visualized on an electronic map [20].

Some social media systems utilize and provide location information at various accuracy levels and run over different geographical scopes (e.g. a street, a suburb,

[5] https://twitter.com/.

a city, a country), and work with different social web sources (e.g. Twitter, Facebook, etc.). For example, Crisis tracker[6] is a web-based system that automatically tracks sets of keywords on Twitter, and filters stories based on location information.

5.3 Location and Context-Awareness

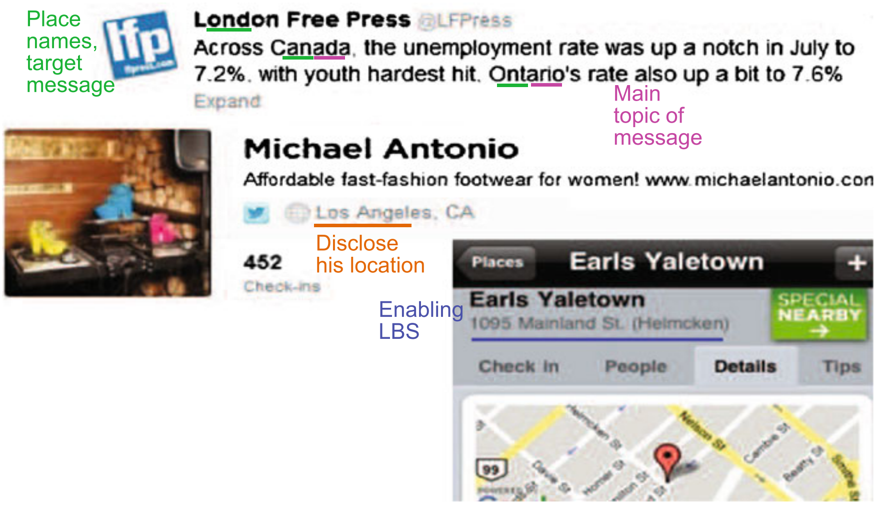
Location-based services have experienced different generations. The first generation location-based services were released around 2000 [3]. Various icons are used to represent different categories of point of interest on an electronic map. The preferred application was the delivery of nearby points of interest (such as restaurants and bars). Advancements in positioning techniques and mobile communications have enabled the second generation location-based services with a broad range of new and sophisticated applications. Here, we mention two typical applications as examples. (i) Social community platforms like Facebook and Foursquare have enabled location sharing for the mutual exchange of location data between users. A special form of location sharing is the check-in function. It is used to explicitly acquire user locations at certain venues. (ii) Locating people to provide special offers or discounts has attracted much attention in mobile marketing. In this way, advertisers could catch the attentions of users by providing advertisements matching their needs. The area of mobile marketing is the next big thing in the mobile Internet [16]. Particularly, we explain in detail the geo-fencing application [13], which is a promising technique for user-centric mobile location-based services.

5.3.1 Location Inference from Social Messages

Social media messages contain different types of location information, such as place names appearing in the message, a location from which the message was sent, and so on. Four types of locations, shown in Fig. 5.4, can be inferred from social web data. *Location in text* is a location type for place names described in a target message (for example, London, Canada, Ontario). *Targeted location* is relevant to the main topic of the target message (for example, Canada, Ontario). *User location profile* is a location type that a user discloses in his profile (for example, Los Angeles). The user's *current location* is a location type that is obtained from location-based service in physical world (for example, 1095 Mainland St.).

When we geo-locate a message, we should consider which location type is appropriate. A framework is proposed in [21] for classifying location elements and a method for their extraction from social web data. This work is related to location inference from text messages. Usually the inputs are the messages and the outputs are the locations. There are two components in the system: location name recognition

[6] http://irevolution.net/2012/07/30/collaborative-social-media-analysis/.



London Free Press @LFPress
Across Canada, the unemployment rate was up a notch in July to 7.2%, with youth hardest hit. Ontario's rate also up a bit to 7.6%
Expand

Michael Antonio
Affordable fast-fashion footwear for women! www.michaelantonio.con
Los Angeles, CA
452 Check-ins

Places Earls Yaletown
Earls Yaletown
1095 Mainland St. (Helmcken)
SPECIAL NEARBY
Check in People Details Tips

Fig. 5.4 Location information that can be inferred from social media data

and toponym resolution. The system extracts terms that are possibly location names as location candidates, and resolves whether or not they are location names in the toponym resolution component. A confidence score for each location instance is calculated by multiplying the location popularity and region context scores [21]. After these calculations, the location instance with the highest confidence score is selected as the result of toponym resolution. Finally, the detected location is assigned coordinates.

5.3.2 Location Inference from Tweets via the SAGE Model

Term distribution of tweets written by a given user depends on several factors such as user preference, region distribution and topic distribution. A user has his preferences over regions where he usually spends his time, and preferences over topics that he often tweets about. In addition, at a specific region, the tweets may contain localized keywords such as an airport, a park, a mall, a city, etc. Moreover, the content of tweets may be associated with the topics at a region and can be classified as sports, politics, travel, daily life, etc. Therefore, a tweet is composed of a bag of words from topic, region and background language models. Then, given a tweet, its location can be inferred by using these language models.

We first give some preliminaries in Fig. 5.5 on how to model term frequency in the log space. For a term v in a model ϕ, its term frequency is β_v, and its log frequency is defined as $\phi_v = \log \beta_v$. Then, the term distribution can be computed by normalizing β_v, and more importantly, it is equivalent to computing the term distribution directly

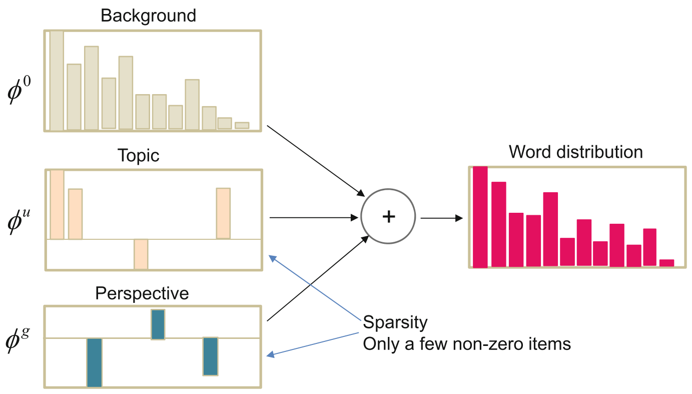


Fig. 5.5 Modeling term frequency in the log space

from the log frequency ϕ_v (Eq. 5.1).

$$p(v|\phi) = \frac{\beta_v}{\sum\limits_v \beta_v} = \frac{\exp(\phi_v)}{\sum\limits_v \exp(\phi_v)}. \tag{5.1}$$

Now consider the sparse additive generative model (SAGE) [22], where several models ϕ^0, ϕ^u, ϕ^g are added together (Eq. 5.2). Their addition in the log space is equivalent to the multiplication of term frequency. In this process, ϕ^0 is a basic reference model (β^0 is the term frequency distribution), ϕ^u is the difference between one model and the reference model (β^u is the rate by which term frequency is increased in this model), and ϕ^g is the difference between another model and the reference model.

$$p(v|\phi^0 + \phi^u + \phi^g) = \frac{\exp(\phi_v^0 + \phi_v^u + \phi_v^g)}{\sum\limits_v \exp(\phi_v^0 + \phi_v^u + \phi_v^g)}. \tag{5.2}$$

The above SAGE model can be used to represent multiple facets involved in automatic generation of text messages. For example, here, use ϕ_0 to denote the log value of term frequencies of a background model. Other components, such as ϕ_u and ϕ_g, are used to describe the topic model and perspective model, which only record the difference from the background model. The SAGE model has two properties. First is sparsity-inducing for a specific model. In other words, only the difference of term frequency of a subset of terms is modeled. For example, in Fig. 5.5, ϕ_u and ϕ_g only have a few non-zero items. Second is to combine generative facets through simple addition in log space. For each term, the non-zero values in all models are added together, and then normalized to get the distribution of terms.

Next, we introduce how a tweet is automatically generated using the SAGE model, based on the term distribution, regional language models, global topics, user

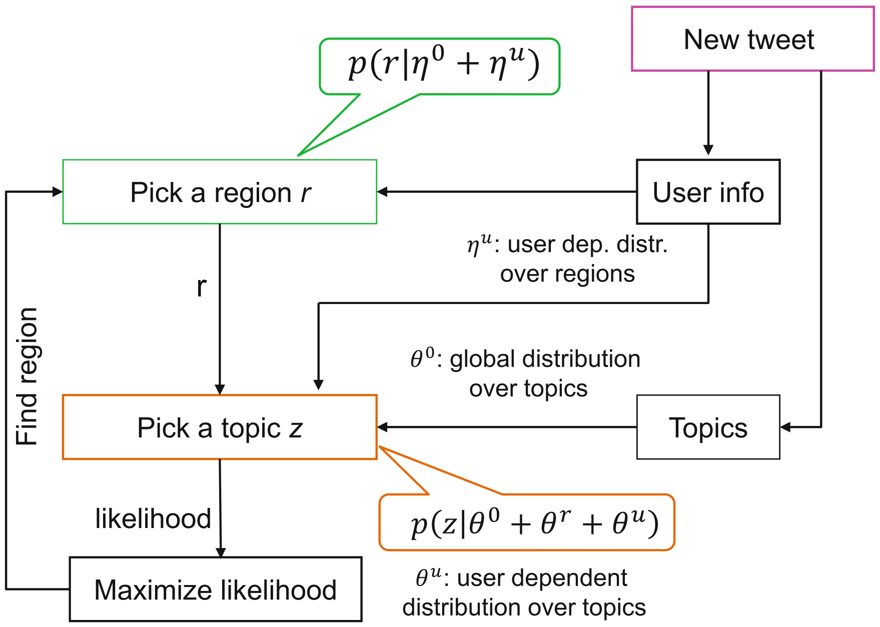

New tweet
$p(r|\eta^0 + \eta^u)$
Pick a region r
User info
η^u: user dep. distr. over regions
r
Find region
θ^0: global distribution over topics
Pick a topic z
Topics
likelihood
$p(z|\theta^0 + \theta^r + \theta^u)$
Maximize likelihood
θ^u: user dependent distribution over topics

Fig. 5.6 Location prediction via tweets using the SAGE model

preferences etc. [23]. A tweet is generated by several steps. In the first step, using both global distribution over latent regions η_0 and user dependent distribution over latent regions η_u, a region r is drawn from the mixed region model $p(r|\eta_0 + \eta_u)$. In the second step, using global distribution over topics θ_0, regional distribution over topics θ_r, and user dependent distribution over topics θ_u, a topic z is drawn from the mixed topic model $p(z|\theta_0 + \theta_r + \theta_u)$. In the third step, each word w in the tweet is successively generated by drawing from the aggregate distribution $p(w|\phi_0 + \phi_r + \phi_z)$, where ϕ_0 parametizes a global distribution over terms, ϕ_r describes the region-dependence of terms, and ϕ_z is a topic-specific distribution of terms.

Although Twitter provides location service, currently only 1 % of tweets are geotagged (latitude and longitude). The previous tweet generation model can be used for location prediction of tweets [23], as shown in Fig. 5.6. Location prediction for a new tweet is based on the words used in the tweet and its user information. User information gives the user dependent distribution over latent regions (η_u). The additive model for region gives a guess of a region r from the model $p(r|\eta_0 + \eta_u)$. Words in the tweet are related to regional distribution over topics (θ_r) and user dependent distribution (θ_u) over topics. On this basis, the additive model $p(z|\theta_0 + \theta_r + \theta_u)$ for topic gives a guess of topic which maximizes the probability. This probability is associated with the region r. Further maximizing this probability with respect to different regions gives the most proper region for the tweet. This is a rough estimation of the location for the tweet.

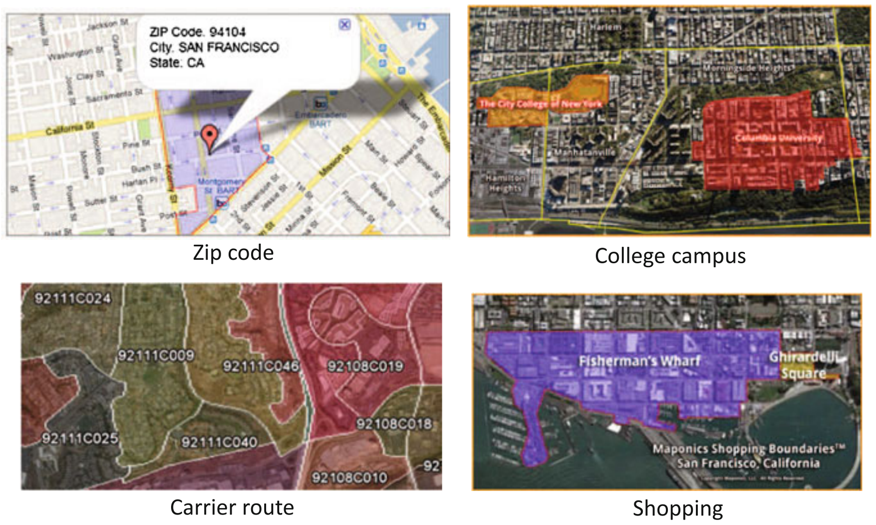


Fig. 5.7 Semantic geo-boundaries in real life reported from http://www.maponics.com/

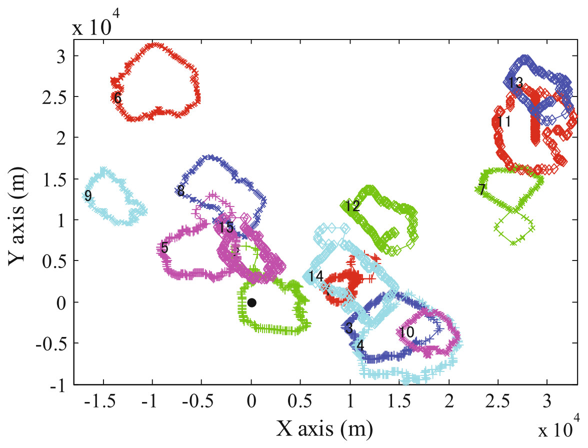
5.3.3 *Context Awareness via Geo-Fencing*

More and more location-based social services want to locate, reach and interact with users on-the-go and provide various services. To this end, the geo-fencing service [13] (e.g., placecast, sensewhere, zentracker) is introduced to respond to personal user needs, and recent years have seen a growing need for user-centric geo-fencing technique in location-based services.

A geo-fence is a virtual perimeter for a real-world confined geographic area. This area can be the coverage of a particular radio cell or a Wi-Fi access point, or specified by a geographic shape. As a result, geo-fences may have different shapes, e.g., circles, rectangles, polygons, which are specified by geographic coordinates. The basic idea behind geo-fencing is very intuitive: when users enter or exit geo-fences based on geo-fencing-enabled location preferences, notifications are sent out to users or their networks of friends.

Various semantic geo-fence boundaries can be predefined to target very specific geographic areas and customers, visualize business opportunities and help to make more informative decisions. Figure 5.7 shows example geo-fences corresponding to zip code boundaries, college campus boundaries, carrier route boundaries, shopping boundaries, respectively.

Geo-fencing is a big feature for user-centric location-based social networks. It mainly deals with pairing a point (a user's coordinate) with a polygon (a semantic geo-fence boundary). In other words, the task is to estimate whether a point is INSIDE or WITHIN a distance of a polygon. Each point has multiple instances each with a unique sequence number, i.e., points can be moving. Each polygon has



edges of 15 polygons (200)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
285	255	235	196	264	250	240	239	226	226	242	153, 15	152, 20	250	217

Fig. 5.8 Polygons, points and edges from training dataset of ACM SIGSPATIAL GIS Cup 2013

multiple instances each with a unique sequence number, i.e., the shapes and positions of polygons may change as well. A point may appear in several polygons (in the overlapping area of polygons). Sequence numbers of points and polygons belong to the same space and have no overlapping. Sequence number works as timestamp and a large sequence number means a recent time. When the sequence number of a point is given, all instances of polygons up to that time should be examined. Figure 5.8 shows examples of polygons. Here, we can find polygons usually are irregular, and each polygon on average contains around 200 edges. Two polygons (12 and 13) further have inner rings, whose numbers of edges are equal to 15 and 20, respectively.

5.3.4 Efficient Geo-Fencing

Geo-fencing is broadly applied in location-based services, e.g., advertisements, child location service. It can be well solved by using the crossing number algorithm [11] (or the winding number algorithm [12]). However, with the rapid increase of geospatial datasets, the geo-fencing technique is required to process millions of points and hundreds of polygons or even more in real-time. So, how to efficiently pair points with polygons is becoming a very important task. Here, we introduce a simple but effective and efficient geo-fencing algorithm [14], which is one of the top winners in ACM SIGSPATIAL GIS Cup 2013.

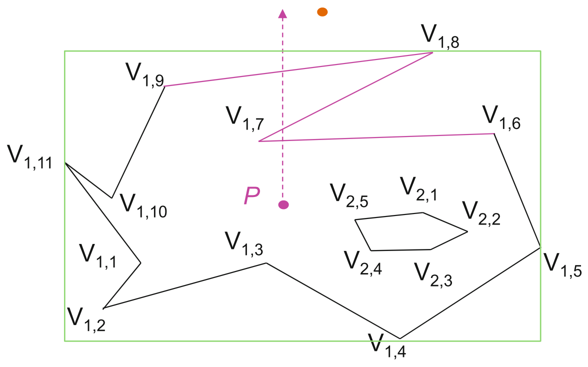


Fig. 5.9 Geo-fencing: detecting whether a point is inside a polygon

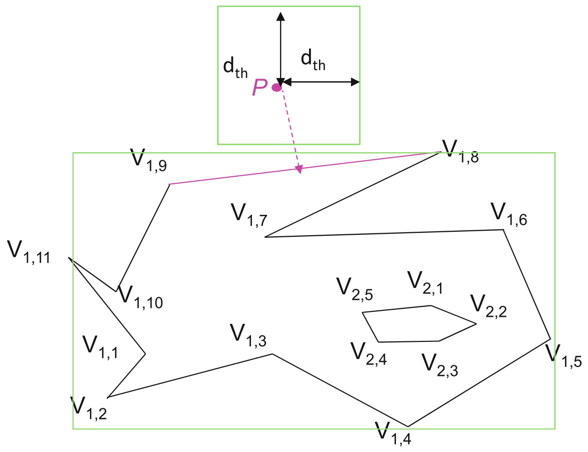


Fig. 5.10 Geo-fencing: detecting whether a point is within a distance d_{th} of a polygon

According to the crossing number algorithm [12], the number of intersections for a ray passing from a point to the exterior of a polygon, if odd, indicates that the point lies inside the polygon, as shown in Fig. 5.9. But this crossing number algorithm requires checking all edges, and becomes inefficient when each polygon contains many edges. Actually, this problem can be simplified by two steps [14]: First, by exploiting the minimum bounding rectangle (MBR) of a polygon, a point outside the MBR of a polygon is surely outside the polygon. An R-tree is further used to quickly detect whether a point is inside the MBR of a polygon. Second, when the point is inside the MBR, instead of an exhaustive search, an edge-based locality sensitive hashing (LSH) scheme is proposed to adapt to the crossing number algorithm. As for the WITHIN detection in Fig. 5.10, a point might be outside the MBR of a polygon but still within a distance d_{th} of the outer-ring of the polygon. In this case, a rectangle centered at the point, with an edge length being $2d_{th}$ is constructed. If this rectangle does not overlap with the MBR of the polygon, the point is surely not within a distance d_{th} of the polygon. Applying LSH in the WITHIN detection is a little more

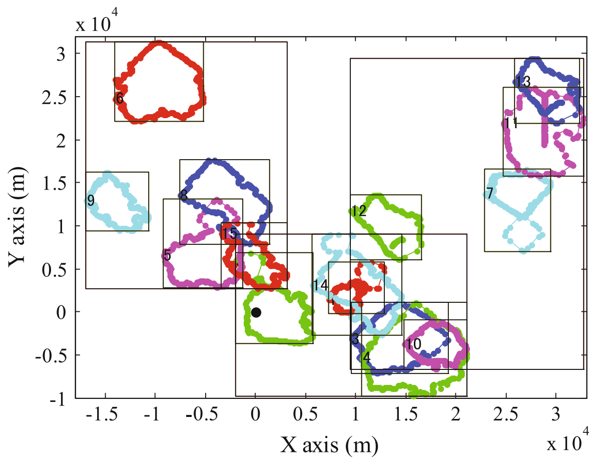


Fig. 5.11 MBR of polygons are organized in the R-tree

complex. A probing scheme is suggested to locate adjacent buckets so as to check all edges near to the target point.

Figure 5.11 shows an example of the relationship between an input point and its latest instances of polygons. In this figure, each polygon has its own MBR, and 15 basic MBRs are further divided into three groups in a higher level in an R-tree. In this way, MBRs that contain the given point are quickly found instead of exhaustive search. The corresponding polygons are regarded as candidates and are further examined.

5.4 Localized and Personalized Search

Personalization has been a trend of web searching. A method to personalized search is to exploit the location information. As is known, there is a geographic locality in user's interest and culture. So, for the same query, people in different areas may expect different results. These days, search engines can return most relevant local results to users according to the location information in user's profile, while filtering out irrelevant information. For example in Fig. 5.12, users searching for a pasta restaurant in Kyoto and Singapore may get local relevant results. From these search results, it is obvious that Google search engine may personalize results based on users' location information. In this way, location information, as an important dimension, complements multimedia retrieval and recommendation.

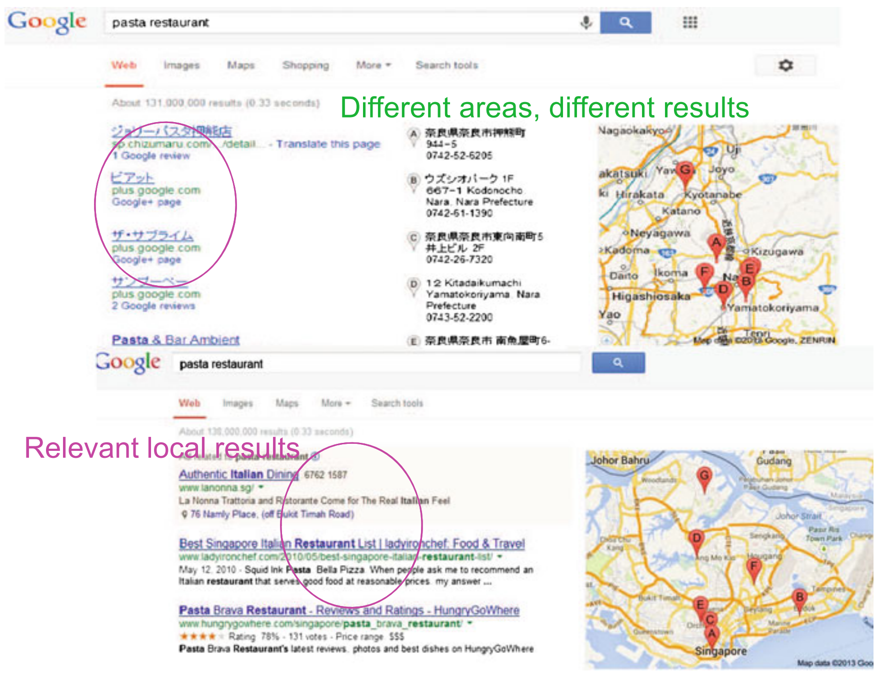


Fig. 5.12 Personalized search via exploiting location information

5.5 Location-Aware Multimedia Applications

Here, we introduce emerging geo-tagged multimedia applications and techniques (e.g., land-use classification, geo-tagged image retrieval). Their common part is to incorporate geographic information as a context in multimedia information processing.

5.5.1 Music Geo-Listening Patterns

Twitter streaming API can be leveraged to retrieve tweets with geo-spatial coordinates. Further using music-related hashtags helps to extract music listening-related tweets. Then, artist information can be extracted by parsing and analyzing the content of these tweets. Music-related tweets often contain patterns, for example artist name followed by song title. In some cases, the artist name might appear as a valid song title, which results in some ambiguity. Generally, pattern-based approaches can be used to match potential artist names against the artist dictionary. Track information can be used to help distinguish artist names, by exploiting the musicbrainz database as a knowledge base for artist names and related song titles.

These music related tweets are classified according to artist genre information [20]. More specifically, the genre tags available for each artist are collected from last.fm, and further refined by using a list of known genres from freebase. Then, the artists (and genre tags) are split into k clusters, k ranging from 10 to 20. Next, each tweet is assigned an artist cluster number based on its included artist information. Further exploiting the coordinates of tweets, the number of tweets per artist cluster per area can be computed as music listening pattern and visualized on a world map.

5.5.2 Geo-Tagged Images for Land-Use Classification

Next example is exploring geo-tagged images for land-use classification. In this application, the problem of geographic discovery, particularly land-use classification, is investigated through crowdsourcing of geographic information from geo-tagged photo collections in Flickr. The geo-tagged photos are represented by their visual or text features to perform land-use classification. This is formulated as a supervised classification problem, in which support vector machine (SVM) [24] is used. Three land-use classes are considered in [19]: academic, residential, and sports. To generate a predicted land-use map, the target area is divided into multiple sub-regions, each separately classified.

Visual features and text features are main components in land-use classification model [19]. An intuitive question in the classification is how to model proximate sensing from visual features or textual features contained in geo-tagged images. Bag of visual words (BoW) with a soft-weighing scheme is used to extract a BoW feature from each image, and a dictionary of 500 visual words is used. Flickr images commonly have user-supplied text associated with them. A dictionary of terms is created based on the words extracted from the title, descriptions, and tags associated with each image. The text analysis is performed at the group level since there is typically not enough text associated with the individual images for effective classification. Each of the text components associated with an image is parsed into a set of terms, and each group of images is represented by a histogram of terms among the dictionary. Then, pLSA (probabilistic latent semantic analysis) [25] is used as a tool to reduce the dimensionality of the term histogram of each image group.

5.5.3 Geo-Visual Image Similarity

Sometimes, it is necessary to identify geo-tagged images that contain similar views of identical objects so as to retrieve similar images taken at the same location. The geographic location of the photo image is measured where the picture is captured, not where the object is located. So, the position in the geo-tag is not the position of the captured object (but of the camera position where the object is taken). Images having identical objects are defined as orthologous images, for example, in Fig. 5.13,

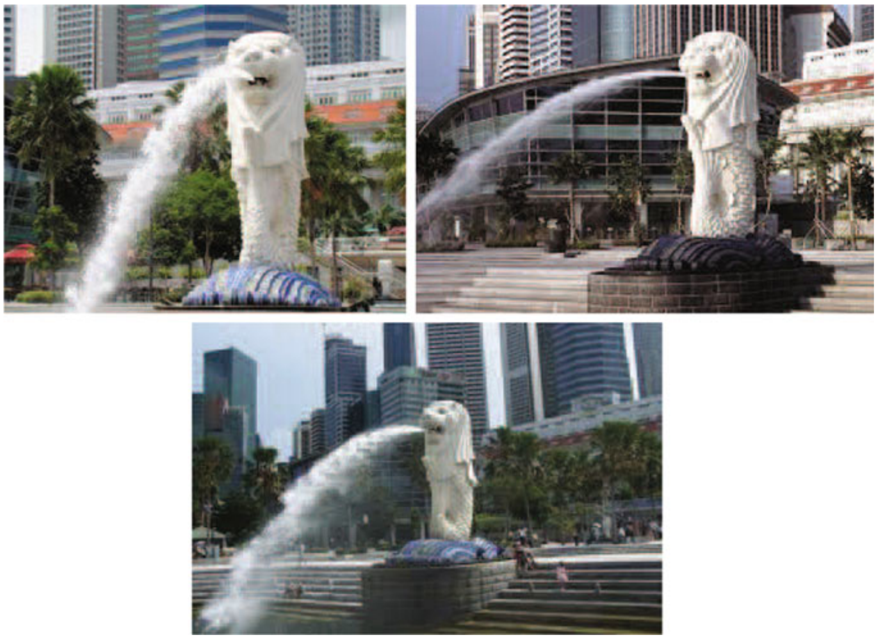


Fig. 5.13 Different photos showing the identical Merlion

three photos are similar to each other. Then, an orthologous identity function (OIF) [26] is used to estimate the degree to which two images are similar. OIF is a similarity rating function that uses both the geographic distance and image distance of photos.

5.5.4 Geo-Location and Context-Based Pedestrian Detection

In previous examples, we introduced classification of immovable objects such as land-use. Now we discuss the classification of movable objects, pedestrian detection related to geo-location.

Pedestrian detection can be conducted by different models with different complexities [27]. In the conventional model, the detection problem can be simply formulated as computing the posterior probability $p(P|V)$, where P denotes the pedestrian label and V denotes visual appearance of image or image batch. The second model adds the geographic location G as $p(P|V, G)$. Different locations will influence both the visual appearance and pedestrian presence probability. Therefore, the third model further exploits the environment context E, considering that different environments will influence the visual appearance of pedestrians. This model involves all factors including geographic location and environment context. Its context-based posterior probability $p(P|V, G, E)$ means the probability that an image contains a pedestrian given the visual appearance V, the location G and the environment E.

By leveraging a vast amount of web images, a contextual image database is constructed, in which each image is automatically attached with geographic location (i.e., latitude and longitude) and environment information (i.e., season, time and weather condition). Two pre-trained classifiers are exploited: a time classifier to decide whether an image was taken in the daytime or at night, a season classifier to decide which season an image was taken in. There is no any hint on weather condition in image metadata. Therefore, the weather condition is divided into three classes (snow, fog and normal), and a weather classifier is trained as well. By incorporating visual feature, geographic location (i.e., latitude and longitude) and environment context (i.e., season, time and weather condition), a context-based pedestrian detection method [27] can be realized by the probabilistic model discussed above.

5.5.5 Soundtrack Recommendation for User-Generated Videos via Context Information

Most user-generated videos, taken outdoor, lack suitable soundtracks. Adding a matching soundtrack to a video can make the video much more attractive for sharing. Generally, different geo-locations convey different affective atmospheres. For example, a busy city has a different atmosphere from a majestic mountain view. In this sense, each geo-category is associated with a mood. Based on mood similarity, a soundtrack can be recommended to a video scene.

Geo-locations can be classified into geo categories through leveraging Foursquare API. A geo-category is further associated with an atmosphere, or a mood at a venue. Table 5.1 shows a potential mapping from geo-categories to moods [7]. User study is conducted to identify which mood should be associated with each geo contextual

Table 5.1 Relationship between geo-categories and moods

Geographic category	Related mood(s)
Arts and entertainment	Quiet, calm
Colleges and Universities	Quiet, calm
Food	Sweet, happy
Great outdoors	Dreamy
Nightlife spots	Funny, intense, playful
Professional and other places	Aggressive, heavy
Residences	Sweet, sleepy
Shops and services	Happy
Travel and transport	Melancholic, bittersweet, funny

category. By using the relationship in this table, a system can automatically rank mood categories for a given geo-location.

The whole soundtrack generation system [7] has two parts: smartphone application and server side. User generated videos are captured by smartphones together with continuous streams of geo-sensor (GPS) information. These geo-sensor data streams are mapped to a set of ranked, textual geo-tags. Geo-tags are further classified to geo-categories via the API provided by Foursquare, and then mapped to mood tags according to a predetermined geo-mood mapping table (refer to Table 5.1). Mood-tags provide the input into a music retrieval engine, which returns a music soundtrack most matching these tags. Finally, the music soundtrack is associated with the video and the new video is ready for sharing.

The performance of soundtrack recommendation for user-generated videos can be further improved by exploiting visual features as well. Especially, the classification results from geo-feature and visual feature via SVM [24] are late-fused to generate a more robust result, as discussed in [8, 9].

5.6 Analysis of Geo-Social Data

Social media are user-centric, and designed for the interactions and communications between users all over the world. Social networking platforms provide ways to create and exchange user-generated contents while sustaining human contact at the same time. Social media have different forms and languages, which include, e.g., videos, images, audio songs, comments, reviews, ratings. Users participate in social networking platforms via different devices, e.g., using desktop PCs, tablets, smart phones, or game consoles. Conventionally, user interface approaches address the user's interaction with devices, the interactions between a user and a software or application. In comparison, online social interactions [2] are the communications between users via the help of social interface. They are established through the self-reinforcing activities of participating users. Social interactions reveal what is going on, what an application or site is about, and reflect psychological views of an identity, the self, interpersonal relationships, and social structures.

In the following, we show with examples some approaches related to the analysis of geo-social data, especially personal preference mining, social knowledge discovery, and geographic distribution of social activities.

5.6.1 Analysis of Social Expertise Based on Number of Check-Ins

A user visits different venues and generates different check-ins online. These check-ins reflect user's location history in the physical world. Foursquare has a hierarchical

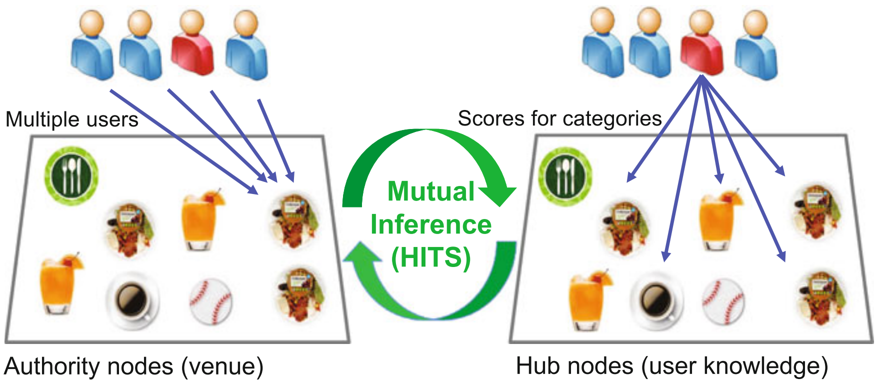


Fig. 5.14 Iterative model for social expertise discovery

category structure which includes 9 top categories and 410 sub-categories. Using the API provided by Foursquare, the geographic trajectory of a user can be converted to a series of geo-categories, which contains user's personal preferences. A user's location history is regarded as a document and categories or sub-categories are considered as terms in the document. By exploiting the TF-IDF (term frequency inverse document frequency) method [28], features can be computed at different levels, using either categories or sub-categories as vocabularies. When computing the similarity between two users in terms of the trajectories, a similarity score can be computed at each level of the hierarchical category, and their weighted sum gives a total similarity between two users [10].

An example is used to explain how to make use of an iterative model for social expertise discovery [10]. In the model shown in Fig. 5.14, each user has his scores for different venue categories, and a venue category is associated with multiple users. For a specific category m, a user's knowledge, $u_m.h$, can be represented by the sum of the authority scores ($v_m.a$) of the venues visited by the user (Eq. 5.3). On the other hand, the authority score of a venue, $v_m.a$, can be represented by the hub scores ($u_m.h$) of the users who have visited this venue (Eq. 5.4).

$$u_m.h = \sum_{u.v \in m} v_m.a. \tag{5.3}$$

$$v_m.a = \sum_{u \in U} u_m.h. \tag{5.4}$$

Then, a user with a high score in a category is regarded as a local expert of that category. To identify the local experts of a venue category, for example, Italian food, based on category information recorded in the user's location history, a user's expertise in each category in different cities can be computed by an iterative model, known as mutual inference [29]. In this process, the initial authority and hub scores are set as the number of user's visits.

5.6.2 Analysis of Business Venues Based on Check-Ins

As mentioned before, Foursquare has 9 top-categories and 410 sub-categories. However, in some cases, for example, in trade area analysis, shopping habits are desired [15]. Top 9 categories cannot effectively distinguish check-in patterns. 410 sub-categories can be more valuable in user profiling but with too high dimension.

The LDA (latent dirichlet allocation) method [30] can be used to identify hidden check-in patterns as topics from the histogram of user check-ins in terms of sub-categories. LDA is widely adopted in document topic modeling. It assumes that each document contains a mixture of topics and each topic has certain probability of mentioning a word. LDA identifies topics and calculates the proportion of different topics in each document by examining word distributions in the documents.

More specifically, the distribution of different topics is calculated for a document. Each user is treated as a document, and each topic is regarded as a term. By analyzing the distribution of topics of customers of a store and computing the histogram of the main topic of all customers, the stores can be profiled as well in terms of potential topics [15].

Another example is shown in Fig. 5.15 for business attractiveness discovery, where the size of an icon is proportional to the popularity of the corresponding business. Consider a number of customers C_1 to C_n and a number of competitor venues V_1 to V_m. a_{ij} represents the number of visits of customer C_i to venue V_j. Then, the probability that a venue V_j is visited can be calculated via

$$P(V_j) = \frac{\sum_{i=1}^{n} a_{ij}}{\sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik}}. \tag{5.5}$$

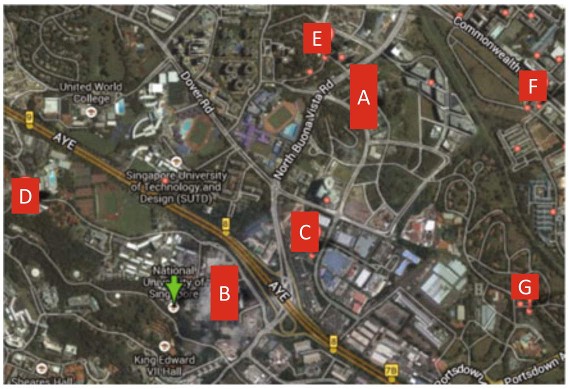


Fig. 5.15 Business attractiveness, the size of an icon is proportional to the popularity of the corresponding business

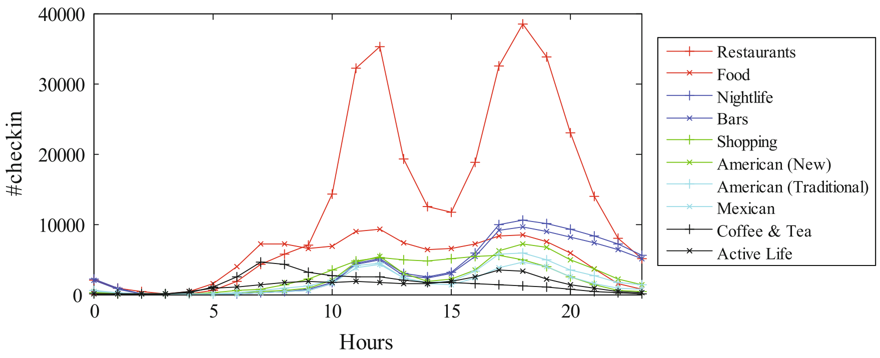


Fig. 5.16 User activities on weekdays in Yelp

This probability is an indicator of local popularity of a business venue. In other words, the probability of venue V_j being visited by all customers in an area reflects its business attractiveness compared with other competitor venues.

5.6.3 Analysis of User Check-Ins in Yelp

In this section, we investigate user behaviors in physical world, by using experimental check-ins data. This data includes 11,537 businesses and 8,282 sets of check-ins from March 2005 to January 2013 in Phoenix, which was provided by Yelp.

Figures 5.16 and 5.17 respectively show user activity patterns across the 10 most popular categories of Yelp on weekdays and on weekends. Top 7 categories are the same in the two figures, which indicates that user business activities on weekdays and weekends have no significant differences in the area of Phoenix. However, we still can find that more people like to travel and go to grocery stores on weekends than on weekdays.

Figure 5.18 shows the CCDF (complementary cumulative distribution function) of the number of per-user check-ins. The number of check-ins varies greatly among users. 50 % users have a check-in count no more than 20. On the other hand, 1 % users have more than 1,000 check-ins.

5.6.4 Analysis of Interest Focus and Entropy in Foursquare

User-generated geo-social data contains user behaviors in physical world and also reflects geographic reach and interest of a geo-category context or a multimedia content across the globe. Here, some methods and examples are given to address

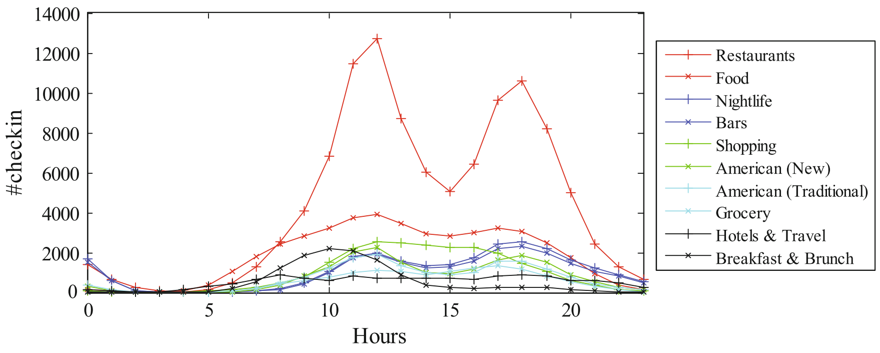
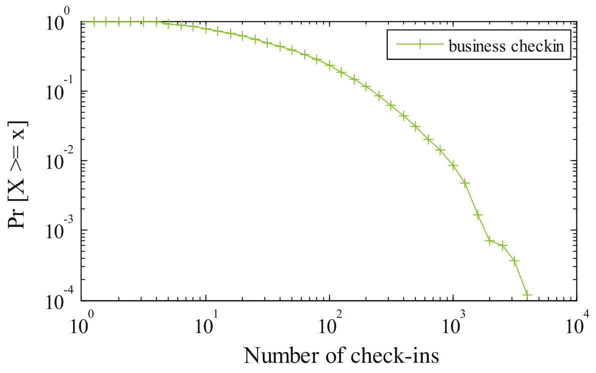


Fig. 5.17 User activities on weekend in Yelp

Fig. 5.18 CCDF of the number of check-ins at business venues in Yelp



geographic distribution of social activities related to geographic popularity of photos, tips and videos.

We investigate the distribution of social media data of LA and NYC, crawled from Foursquare, where 2,728,411 venue photos and 1,212,136 tips are used in the experiments. Since Foursquare is a location-based social networking platform, large volumes of tips and photos are posted in this community. Each user has different interest over all the geo-categories, reflected in the variations of the number of per-category tips or photos. In other words, the distribution of a user's visit in terms of geo-category would likely exhibit a non-uniform distribution, with a large fraction of visits in only a few categories. The distribution of user interest can be measured by two metrics, interest focus and interest entropy.

Figure 5.19 shows the distribution of the number of visits of a user in each category, which is usually non-uniform. Let v_{ik} represent the number of visits of a single user i to a category k. Then, interest focus of a user is defined as its highest fraction of visits, as follows:

$$F_i = \max_j \frac{v_{ij}}{\sum_k v_{ik}}. \tag{5.6}$$

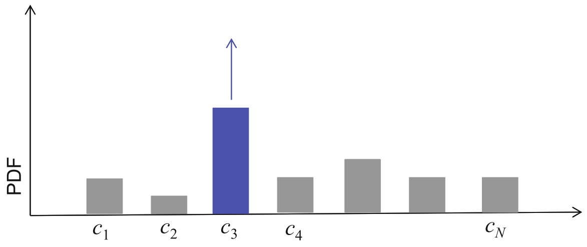


Fig. 5.19 Distribution of the number of visits to different categories for a user

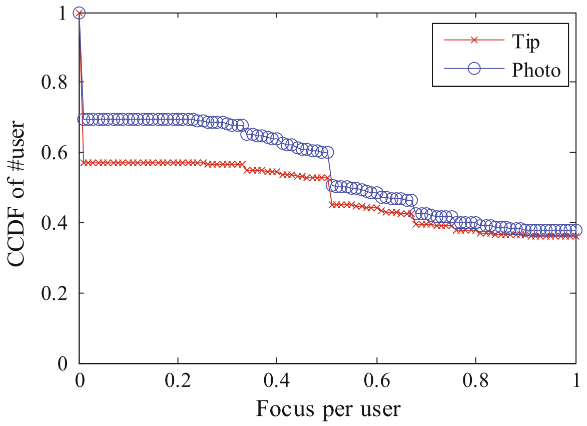


Fig. 5.20 CCDF of interest focus in terms of photos and tips in Foursquare

A higher interest focus means that the interest of a user is more limited to a specific category.

CCDF of per-user interest focus is shown in Fig. 5.20, where the visit is defined in terms of tips (messages) or photos. In this figure, we used the top-9 categories. Nearly 50 % users have an interest focus greater than 0.5, which indicates that many users have a primary interest (in terms of geo-category).

Interest entropy is the other metric for evaluating how user interests are distributed over different categories. With the fraction of visits to a category in Fig. 5.19 as a probability, interest entropy is computed as a standard entropy, as follows:

$$H_i = -\sum_k p_{ik} \log_2 p_{ik},$$
$$p_{ik} = \frac{v_{ik}}{\sum_j v_{ij}}. \tag{5.7}$$

It reflects how user interests are distributed over different categories. A higher interest entropy means a more uniform distribution of visits to different categories while a lower value means interests are focused in fewer categories.

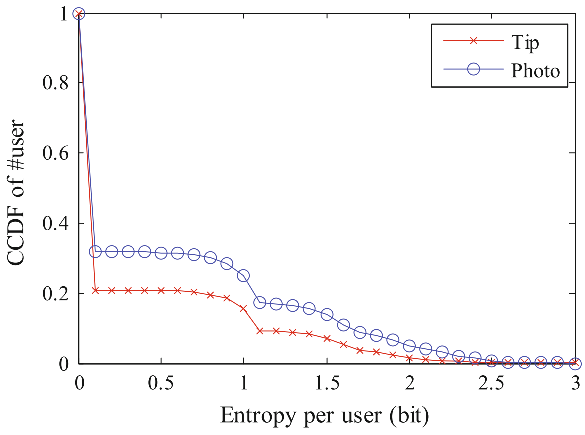


Fig. 5.21 CCDF of interest entropy in terms of photos and tips in Foursquare

Figure 5.21 shows the CCDF of per-user interest entropies, in terms of tips and photos. Only 20 % users have an interest entropy of photo greater than 1bit, or the number of categories being frequently visited is equal to 2. The interest entropy of tips is lower.

5.7 Integration of Social and Content Networks

These days, users find videos from the Internet by different methods. Some of the videos are directly searched via the web sites, some other videos may be recommended between users through their social connections. As a result, social connection has a significant impact on video views. In addition, the effect of social sharing is becoming more important as more users are involved in the social networks.

5.7.1 Geo-Social Networks

Every day, a huge volume of Internet traffic is generated by online multimedia sharing platforms such as YouTube, Flickr, Last.fm. These platforms often rely on content delivery networks [1] to distribute their content from storage servers to multiple locations over the planet. Servers exchange content in a cooperative way to maximize the overall efficiency. Recently, content diffusion is also fostered by web-links shared on online social networks [2]. This may generate large amounts of requests to the provider through the cascading across a user's social links. Content discovery heavily depends on the web search. Web search services like Bing and Google Web Search now are an integral part of our daily life. Google Search alone receives 12.8 billion queries[7] every month from U.S. users. People use web search for a couple of reasons, including listening to music, watching baseball, and making purchase decisions.

Lots of applications and services on the Internet have been developed to make use of location information to meet users' daily needs. The increase of various social media services requires a global platform for sharing user-generated contents, such as videos, images, music, blogs and tweets. Location-enabled tagging for social contents via smart phones and social media services reflects geo-spatial logs of user activities. Users can link their presences and multimedia contents (for example, video, image) to a particular place. Geo-spatial footprints generated by users provide interesting information about the spatio-temporal dynamics of online memes [31], which have important implications for a variety of multimedia systems and applications [8, 20, 32]. On the one hand, geo-spatial data contains personal physical logs of each individual user. On the other hand, it also reflects social behaviors related to the community as data of more users is aggregated. These geo-spatial data could be very useful for studying various lifestyle patterns [33], e.g., public health, cultural identification, urban computing.

5.7.2 Graph Representation of Geo-Social Networks

Next we introduce how to model social networks via a graph [17]. Online users are located over the 2-dimensional surface of the Earth. The great-circle distance is adopted as the metric. The distance in Fig. 5.22 between any two nodes is calculated as a great-circle distance from their geographic coordinates. The social tie between two nodes is represented by a link between them.

A social network can be represented as an undirected graph G, with a node set N and a link set K. When there is a social tie between two users i and j, a link is established between them. The link length is associated with the great circle distance l_{ij}. It is useful to find how friends of a user are geographically distributed. One useful metric is node locality (Eq. 5.8). Considering node i and all its neighbors in a set Γ_i, the geographic closeness between two nodes is measured by a function of normalized

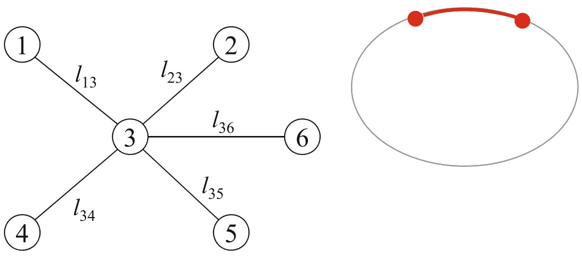


Fig. 5.22 Friendship and great circle distance

[7] http://bit.ly/ThGnOc.

distance using a parameter β. The average over all nodes in the neighbor set Γ_i gives node locality [17] of node i.

$$L_i = \frac{1}{|\Gamma_i|} \sum_{j \in \Gamma_i} e^{-l_{ij}/\beta}. \tag{5.8}$$

In this way, the node locality represents the average closeness between a user and his friends, and decreases as the actual distance gets larger. It is useful when exploiting social connections to recommend multimedia contents, as is discussed later.

The node locality can be investigated by using the cascade of Twitter messages. Twitter messages are shown on the author's personal page and also sent to the author's followers. A node is used to represent a user with a geographic location. Then, a directed graph of users can be extracted from the dataset of tweets, and node locality of each user can be calculated.

A more interesting phenomenon is the spreading of YouTube video links via tweets. A cascade over a social network begins when the first user shares some content and becomes the initiator of the cascade. After this event, some of his contacts will share the same content again, and the cascade will recursively spread over the social links.

5.7.3 Geo-Social Multimedia Content Delivery

Now we introduce some methods related to multimedia content diffusion, for example, geo-social cascades, caching policies and distributed cache. The popularity of multimedia content over the Web can be driven by public media coverage. This type of phenomena often results in globally popular items, which should be widely replicated throughout a content delivery network. Alternatively, content may become popular over social networking platforms because people share it and talk about it. In this way, content may easily spread from a small set of users to a vast audience through social connections, for example, 700 YouTube video links are shared on Twitter every minute.[8]

Social sharing also has a large impact on content delivery network. The latter is a system of networked servers holding copies of data items, placed at different geographic locations as shown in Fig. 5.23. Its performance is influenced by the geographical distributions of the requests. Then, it would be very useful to understand whether an item becomes popular on a planetary scale or just in a particular geographic area. A globally popular content item should be replicated at every location, since it receives many requests from all around the world. On the other hand, when content is only locally popular, it should be cached only in specific locations.

As for standard caching policies [34] used in content delivery networks, each policy assigns a priority $P(v)$ to a video v, and the video with the lowest priority is

[8] http://www.streamsend.com/.

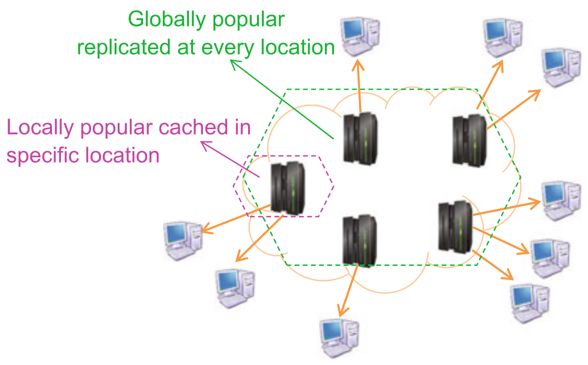


Fig. 5.23 Content delivery networks handling locally popular and globally popular contents differently

chosen for deletion when the cache buffer gets full. There are three typical caching policies. In Least-Recently-Used (LRU) policy, $P(v)$ equals $clock(v)$. $clock(v)$ is the last time that the video v is watched, and it involves the simple aging effect. In Least-Frequently-Used (LFU) policy, $P(v)$ equals $Freq(v)$, where $Freq(v)$ is the number of times video v has been requested since it was stored in the cache. The mixed policy combines LRU and LFU, and the priority of video v is given by $P(v) = clock(v) + Freq(v)$, in order to balance both temporal and popularity effects.

The above caching policies can be further augmented by exploiting geo-social information. Twitter fosters the popularity of YouTube, since users tend to tweet about videos they like, triggering a spreading of the video. This provides us some opportunities to investigate how geographic information is extracted and used to improve caching of multimedia files. There are two augment caching policies [18] based on the characteristics of the geo-social cascades involving one video. One is Geosocial (shown in Fig. 5.24), the extra weight of video v, which is added to its priority, is the sum of the node locality values of all the users that have posted a message about the video, even though they are not involved in a social cascade. The other is Geocascade (shown in Fig. 5.25), the extra weight of video v that is added to the priority is the sum of the node locality values of all the users participating in the video's social cascade. In this way, exploiting social connections helps in finding whether a video becomes popular and helps in optimizing the cache management.

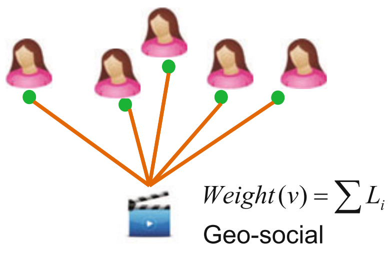


Fig. 5.24 Geosocial in geographic social networks: users access the same video

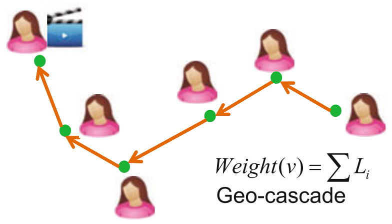


Fig. 5.25 Geocascade in geographic social networks: users are in the same social cascade

5.8 Summary of This Chapter

With an overwhelming amount of social multimedia content on the Internet, it is difficult to find what users are really interested in. For example, a search for "pasta" may return hundreds of millions of social media items. In addition, sometimes, an incomplete query may lead to results of different meanings, where more accurate search requires further information on user preference. For example, a search for "apple" returns the fruit apple and the apple brand. Personalization has been a trend of web searching. Recently, many social networking platforms have provided location-based services, by either explicitly letting users choose their places or implicitly enabling geo-tagging to associate multimedia content with latitude and longitude. Location information has been a very important aspect that helps better understand online social media contents and people activities in physical world.

This chapter takes user location as a clue to discuss broad topics over location-aware multimedia systems. We have talked about fundamental components related to geographic-aware social media, mobile users, social activities and multimedia content delivery. More specifically, geo-social data contains rich context and has two aspects of implication: individual user interest and geographic-social behaviors. We have shown some examples of geographic-aware social media and social interaction data, and reported latest geographic-aware multimedia applications and methods, for example, how to leverage tweets with geospatial information for mining music listening patterns, how to map geo-categories to moods. We also have discussed some location-enabled advanced topics and approaches. Particularly, we explained geo-fencing in detail, which is a promising technique for user-centric mobile location-based services. We showed some approaches related to personal preference mining, social knowledge learning, geographic distribution of social activities and multimedia content diffusion. To sum up, exploiting location information to mine user preference and social links to predict content popularity will greatly affect the form of content retrieval and delivery, which are attracting, and will continue to attract much research interest.

Acknowledgments The work presented was in part supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

References

1. Pallis G, Vakali A (2006) Insight and perspectives for content delivery networks. Commun ACM 49(1):101–106
2. Boyd DM, Ellison NB (2007) Social network sites: definition, history, and scholarship. J Comput-Mediat Commun 13(1):210–230
3. Junglas IA, Watson RT (2008) Location-based services. Commun ACM 51(3):65–69
4. Brodersen A, Scellato S, Wattenhofer M (2012) YouTube around the world: geographic popularity of videos. In: WWW'12, pp 241–250
5. Shen ZJ, Arslan Ay S, Kim SH, Zimmermann R (2011) Automatic tag generation and ranking for sensor-rich outdoor videos. In: ACM Multimedia'11, pp 93–102
6. Ma H, Zimmermann R, Kim SH (2012) HUGVid: handling, indexing and querying of uncertain geo-tagged videos. In: ACM SIGSPATIAL'12, pp 319–328
7. Yu Y, Shen, ZJ, Zimmermann R (2012) Automatic music soundtrack generation for outdoor videos from contextual sensor information. In: ACM Multimedia'12, pp 1377–1378
8. Shah RR, Yu Y, Zimmermann R (2014) User preference-aware music video generation based on modeling scene moods. In: ACM MMSys'14, pp 156–159
9. Shah RR, Yu Y, Zimmermann R (2014) ADVISOR-personalized video soundtrack recommendation by late fusion with heuristic rankings. In: ACM Multimedia'14
10. Bao J, Zheng Y, Mokbel MF (2012) Location-based and preference-aware recommendation using sparse geo-social networking data. In: ACM SIGSPATIAL'12, pp 199–208
11. Shimrat M (1962) Algorithm 112: position of point relative to polygon. Commun ACM 5(8):434
12. Hormann K, Agathos A (2001) The point in polygon problem for arbitrary polygons. Comput Geom 20(3):131–144
13. Kupper A, Bareth U, Freese B (2011) Geofencing and background tracking—the next features in LBS. In: INFORMATIK11
14. Yu Y, Tang SH, Zimmermann R (2013) Edge-based locality sensitive hashing for efficient geo-fencing application. In: ACM SIGSPATIAL'13, pp 576–579
15. Qu Y, Zhang J (2013) Trade area analysis using user generated mobile location data. In: WWW'13, pp 1053–1064
16. Quercia D, Di Lorenzo G, Calabrese F, Ratti C (2011) Mobile phones and outdoor advertising: measurable advertising. IEEE Pervasive Comput 10(2):28–36
17. Scellato S, Noulas A, Lambiotte R, Mascolo C (2011) Socio-spatial properties of online location-based social networks. In: ICWSM'11
18. Scellato S, Mascolo C, Musolesi M, Crowcroft J (2011) Track globally, deliver locally: improving content delivery networks by tracking geographic social cascades. In: WWW'11, pp 457–466
19. Leung D, Newsam S (2012) Exploring geotagged images for land-use classification. In: GeoMM'12, pp 3–8
20. Hauger D, Schedl M (2012) Exploring geospatial music listening patterns in microblog data. In: 10th international workshop on adaptive multimedia retrieval
21. Ikawa Y, Vukovic M, Rogstadius J, Murakami A (2013) Location-based insights from the social web. In: WWW'13 companion, pp 1013–1016
22. Eisenstein J, Ahmed A, Xing E (2011) Sparse additive generative models of text. In: ICML'11, pp 1041–1048
23. Hong L, Ahmed A, Gurumurthy S, Smola A, Tsioutsiouliklis K (2012) Discovering geographical topics in the Twitter stream. In: WWW'12, pp 769–778
24. Joachims T, Finley T, Yu CN (2009) Cutting-plane training of structural SVMs. Mach Learn 77(1):27–59
25. Hofmann T (1999) Probabilistic latent semantic indexing. In: ACM SIGIR'99, pp 50–57
26. Kamahara J, Nagamatsu T, Tanaka N (2012) Conjunctive ranking function using geographic distance and image distance for geotagged image retrieval. In: GeoMM'12, pp 9–14
27. Liu Y, Shi Z, Wang G, Guan H (2012) Find you wherever you are: geographic location and environment context-based pedestrian detection. In: GeoMM'12, pp 27–32

28. Robertson S (2004) Understanding inverse document frequency: on theoretical arguments for IDF. J Doc 60(5):503–520
29. Zheng Y, Zhang L, Xie X, Ma WY (2009) Mining interesting locations and travel sequences from GPS trajectories. In: WWW'09, pp 791–800
30. Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. J Mach Learn Res 3:993–1022
31. Kamath KY, Caverlee J, Lee K, Cheng Z (2013) Spatio-temporal dynamics of online memes: a study of geo-tagged tweets. In: WWW'13, pp 667–678
32. Yamasaki T, Gallagher A, Chen T (2013) Personalized intra- and inter-city travel recommendation using large-scale geotags. In: GeoMM'13, pp 25–30
33. Yu Y, Aizawa K, Yamasaki T, Zimmermann R (2014) Emerging topics on personalized and localized multimedia information systems. In: ACM MM'14
34. Wang J (1999) A survey of web caching schemes for the Internet. ACM SIGCOMM Comput Commun Rev 29(5):36–46

Chapter 6
In-house Multimedia Data Mining

Christel Amato, Marc Yvon and Wilfredo Ferré

Abstract Multimedia Data Mining research conducted by the European IBM Human Centric Solutions Center group is presented by the in-house multimedia project, whose objective is creating a framework for in-house smart care monitoring for the aging population. The chapter presents an overall framework for collecting, aggregating, and preprocessing multimedia data in real-time and uploading it to a backend system. As a result, this combination of data from diverse sensors provides a wide range of opportunities for applications, which can be built in the prospect of *Smart Care*. The objective of these applications is to get insights from sets of raw data. In the in-house multimedia implementation we demonstrated how multimedia data collected over time does provide valuable insights about a particular person behavior and wellness. Collecting data from multiple sensors and combining them is essential for better understanding people behavior and helps to avoid misleading conclusions.

6.1 Introduction

Digging in the mass of multimedia information becomes a key challenge and opportunity for business and individuals. More and more devices and connected objects are equipped with sensors participating in the collection of many kinds of data [1]. Machine-to-Machine communications are developing more and more with smart

C. Amato (✉)
IBM France Laboratory, 17 Avenue de l'Europe, 92275 Bois Colombes Cedex, France
e-mail: christel.amato@fr.ibm.com

M. Yvon
IBM Human Centric Solutions Center, 17 Avenue de l'Europe,
92275 Bois Colombes Cedex, France
e-mail: yvon@fr.ibm.com

W. Ferré
IBM Integrated Health Services, 17 Avenue de l'Europe,
92275 Bois Colombes Cedex, France
e-mail: wil_ferre@fr.ibm.com

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_6

applications as described in [2]. This is quite representative of the big data phenomenon with a strong heterogeneity as described in [3, 4]. They feed systems with multimedia information making it very challenging to identify hidden meaning. Multimedia data can be gathered and integrated in real-time and in mobility to a backend system.

Dealing with data has to go through several consecutive steps that we are going to describe in this chapter through a development related to an in-house multimedia for the aging population project. Helping and accompanying elderly people is a rising concern since connected objects are opening up new possibilities. As described in [5], a vision-based smart home care system aims at monitoring elderly persons remotely. Such a system is dedicated to accidental fall down only and is intrusive with installed cameras in the house. In [6] an intelligent, robust, costless, flexible, and real-time home monitoring system is described that records the basic home activities and timely response, when there is a change in the regular daily activity of the elderly person. This system records the usual activity (on or off) of many devices such as TV set, oven, etc., by installing an electronic card inside the device.

Our purpose was not only to detect abnormal behaviors of aging people according to their habits (the normal) but also to supervise their activity regarding wellness without being intrusive. In this way we have equipped houses with connected sensors so that data can be collected and analyzed.

The first step we accomplished was related to data acquisition. It is not as simple as it may look. Several aspects to take care of are the following:

- Managing the acquisition devices: sensors and indicators.
- Ensuring a proper digital format to facilitate further processing.
- Managing the transmission system.
- Filtering and aggregating data.

Once data are gathered, the next phase is apply analytics.

This is currently a booming area in IT. It is about finding meaningful information for the purpose: improving the client value-proposal, detecting not yet obvious trends in the society, customizing a treatment for a specific patient, improving the accuracy of predictions, etc.

There is one aspect to take care of before applying methods and statistics to the data: clean up the collected data. The challenge here is to identify data of interest and to isolate it from “noise”.

6.2 Smart Care in Bolzano: In-house Multimedia Project

The trend of aging explosion of the population has no parallel in human history. The growth of population over the age of 60 happens with a simultaneous and gradual decline of the population under the age of 15. By 2050 the elderly population is predicted to surpass the teenage population for the first time in human history. Life expectancy has increased on average up to nearly 82 years.

The growth of aging population is already heavily influencing all areas of day-to-day human life, and its influence will continue. In the economic area, the aging population will affect economic growth, savings, investment and consumption, labor markets, pensions, taxation and the transfers of wealth, property, and care from one generation to another.

Aging population will continue to affect health and health care, family composition and living arrangements, housing, and migration.

The Smart Care in Bolzano Project has the following goals:

- Meeting high quality health care needs of elderly people.
- Maintaining support of elderly people at manageable levels for society.

The potential offered by technology also extends to other domains, including deeper immersion of old people in everyday social life and support for active aging in the context of work/employment.

The City of Bolzano aimed to become a model in developing new approaches to assistance and support to aging population by introducing innovations in social services through technology and establishing new economically sustainable models. This project target was to remotely monitor health of citizens of age 75+ at home across their daily activities.

The "Secure Living Project," sponsored by the City of Bolzano was part of an initiative to enhance the quality of life of the elderly, providing greater independence and integration into society, while, at the same time, lowering public spending through more accessible and less invasive technologies.

Data from each home were collected and analyzed, and if necessary, if immediate action was required, a first alert was sent to a panel of relatives/or so called "angels" and subsequently to a dedicated team from the Bolzano Social Services Department.

6.3 Data Acquisition

6.3.1 Managing the Acquisition Devices

Managing the acquisition devices is a crucial step in any project. Each device must be chosen not only according to the needs but also to the constraints. Each device must be selected for captured data taking into account the requirements for accuracy and frequency of data stream, and technical requirements for output connections, power consumption, size, noisiness, and disturbance.

In our in-house multimedia project 30 elderly people between 66 and 80-years old and in need of in-home assistance were equipped with sensors and monitoring equipment in their homes as part of the pilot project. The main challenges from a technical setting were not to be intrusive in the houses with residents inside. Elderly people do not like when their everyday life is being disturbed. All devices were

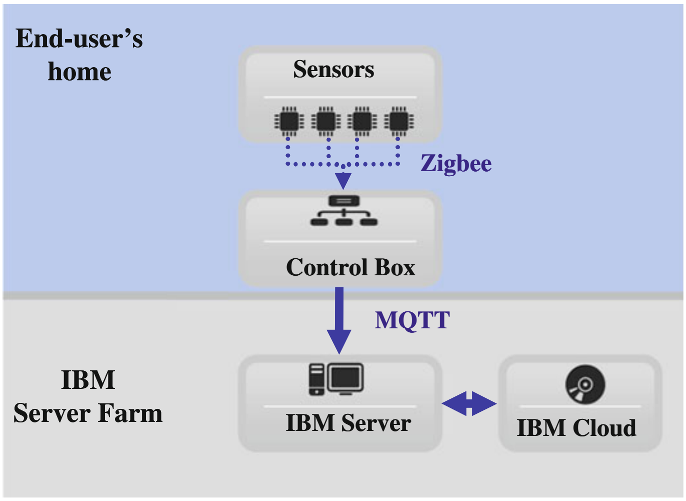


Fig. 6.1 Structure of the device system inside and outside the end-user's home

selected to be discrete and autonomous. They have to be launched and to connect themselves. Figure 6.1 shows the overall structure of the system inside and outside the user's home.

Each home was equipped with sensors to monitor constantly the following home parameters: water leakage, light, CO, CO_2, smoke, temperature, and humidity. All the sensors were running on battery power so that they did not need to be connected to the electrical network. Water sensors were used to detect abnormal water on the floor. They were installed in the kitchen under the sink so that it remained discreet. The other sensors were put in the main room. All the multimedia sensors were coming from several manufactures. We preferred devices that met needs, size, and easy way to install.

All sensors were sending data over a wireless network to a control box, which role was to send collected data to the IBM servers. Data stored in the IBM cloud were available immediately on a dashboard for supervising in real-time or for analyzing them.

6.3.2 Ensuring a Proper Digital Format of the Multimedia Data

In multimedia project, the collected data come from several devices of different types. As they can be selected from distinct manufacturer, the output data usually have different format. The output format can be in a manufacturer format (for example, binary data and strings that contain data about localization, time, etc.) or in a proprietary format.

Due to the IBM Cloud able to store unstructured data, all these heterogeneous data can remain in the initial form without any modifications. In this context of data manipulation, it means one needs to know the initial format for each type. As the collected data is not necessarily explored and analyzed by the same team that collected it, it is very important to standardize this data. Thus, the technical challenge in standardizing the collected data is to keep the most relevant information in the data.

In the in-house multimedia project, the control box gets the data from each sensor and attaches additional data, such as the id of the house, where the data were collected, the sensor id and type, etc. When the timestamp is not contained in the input frame, the control box added it from its own internal clock. Before doing this, all the existing equipment clocks were synchronized with the control box.

6.3.3 Managing the Transmission System

The selection of a network must be in compliance with local conditions and data flow bandwidth. The transmission between node devices is adapted and has different characteristics than transmission to the cloud server.

The technical challenges in the in-house multimedia project were to create a wireless network in a building made of several apartments without any Internet connection. The best practice was to install a Zigbee communication system. We had to cope with this issue and transmit data from the standard telecom network by positioning the control box unit (the component receiving all data from all sensors) at the highest floor for better network connection.

The solution was interconnected in two ways:

- The sensors were interconnected between themselves via a Zigbee mesh-network for broadcasting data about one unit, i.e., apartment and assisted person.
- Aggregated data was broadcasted to the IBM server using a 3G network.

6.3.4 Filtering and Aggregating Data

Collecting data periodically every minute or every second may generate large amount of data in one week. The collected values may be redundant and not worth to be stored. Some sensors do not vary often; their consecutive values may be constant during hours. Filtering the data consists of eliminating redundancies according to a defined precision. If the difference between the last collected value and the new one is less than the precision, then the new value is ignored.

Once all the devices are installed, switched on, and connected with each other, the entire solution can be tested from the terminal device to the IBM Cloud. The test consists of verifying that all the chains are running well and that the sensors are varying depending on environmental conditions.

Table 6.1 Volume of collected data during the in-house multimedia project

Sensor type	Total readings	Type of data	Units of measurements
CO_2	69,272	Numeric	Part per million
Temperature	50,621	Numeric	Degree celsius
Humidity	35,441	Numeric	Percentage
Light	31,466	Numeric	Lumens
Water	19,831	Numeric	Percentage
CO	17,931	Numeric	Part per million
Smoke	8,804	Numeric	Part per million

In the in-house multimedia project the test consisted of verifying that data was arriving to the cloud in real-time without being disturbred by the flow of data coming from the 30 houses in the 3G network. Then we had to verify the water leak sensor by spilling water on it and to notice that the collected value increased. The CO_2 sensors were tested by blowing into the sensors.

Once the overall system has been tested, the experiment begins. The pilot had been up and running for 8 months. During this time 238,965 data values were collected. Table 6.1 shows the volume of the collected data for each sensor.

As we can see "CO_2" is the most variable parameter while "smoke" detector is the most stable one. "CO_2" is varying each time people are coming in the room or leaving it. "Smoke" value is varying when a person is cooking. Smoking is forbidden inside apartments.

6.4 Data Analytics

6.4.1 Cleaning Up the Data

Cleaning data is an important step that helps to alleviate the load on the limited available computing power. But first we have to decide what is considered as noise. Obviously, it is made of data that is not relevant to your specific purpose. However, what is considered as noise for one purpose could be useful for another one.

We used filtering for the following targets:

- Eliminating information not relevant to the pursued goal.
- Assessing the quality of collected data.

The quality assessment of the collected data is a major feature. It is important for further analytics accuracy to know what is the confidence associated with to each collected piece of data. It depends on the quality of the sensor, the position of the source, and several other factors. And it is very important to treat it as a dynamic process—a sensor may break down or degrade at some point of time.

6.4.2 Analyzing the Data

Analyzing data consists of operating multidimensional methods on data. Most of them come from statistics or machine learning. Two kinds of analyses are distinguished: descriptive analysis and predictive analysis.

Descriptive analysis describes the features of a collection of data. It can be applied to one parameter calculating mean, range, regression, variance, pattern detection, etc., or include several data sources identifying dependencies, correlation, etc. It comes with a learning phase.

We have applied the Multimedia Data Management method to the collected data in the in-house multimedia project, which allowed us to detect patterns over a certain time. Figure 6.2 shows a correlation between the participant's sleep pattern, daily routine, and the safe levels of carbon monoxide (CO). The CO level is very low and stable while the person is asleep until 7 a.m. We can notice little or no activity in the home. The level of CO noticeably rises when the person wakes up at 7 a.m and uses the kitchen for breakfast. The level drops again and rises around noon for lunch. The pattern is then repeated at 4 p.m and in the evening. CO then falls back to the very low level again once the occupant goes to bed at night.

Once we have pattern associated to each people, divergencies can be notified and alerts can be sent.

Having CO increased in many times is dangerous for the people living in the house. It can be due to the gas cooker being left on. In that case, someone must interact in emergency. In another case, CO remaining flat during the day is significant and people must be closely watched.

Having regular meals indicates that the person is active and in good shape. Missing meals or irregular behavior could be an early indicator of change in the person's physical or psychological health.

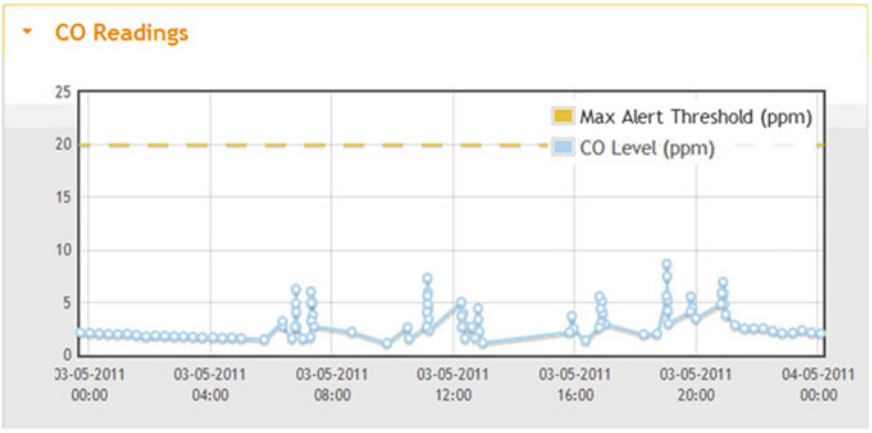


Fig. 6.2 CO readings in a period of one day

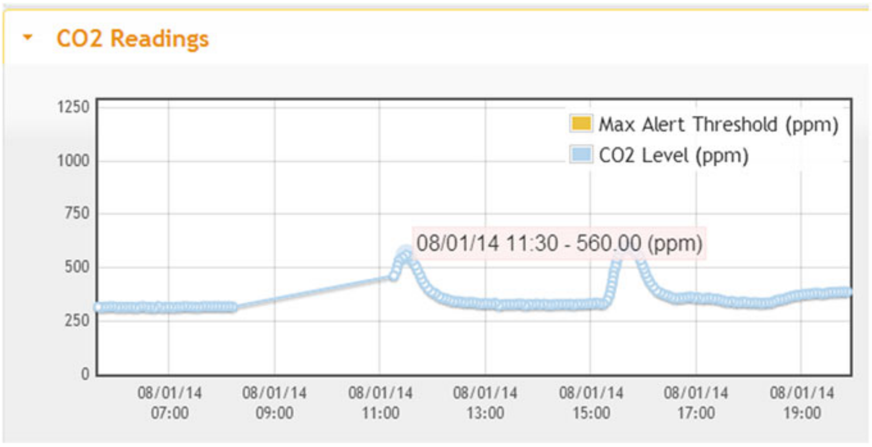


Fig. 6.3 CO_2 readings in a period of one day

Figure 6.3 shows another example of behavior that is correlated with sensor values. It shows a correlation between visits to a participant and the safe level of carbon dioxide (CO_2). The CO_2 level is low and stable until 11 a.m when a nurse brings medicines to the participant. Another visit occurred at 3 p.m and it increased the CO_2 level.

When a person has guests at home mostly on weekends the CO_2 increases. The person can then be supervised without being intrusive (we never know “who”). Regular visits indicate that the person is not alone and can talk to someone. Depriving visits could be an early indicator of change in the person’s physical or psychological health.

When we first demonstrated the system, there were 20 people in one of the rooms and as a consequence the CO_2 rapidly increased over the alert limit. The administrator received its first real alert.

6.5 Conclusion

In the Smart Care Bolzano implementation we demonstrated how multimedia data does provide valuable information about a given person’s wellness by analyzing records accumulated over a certain period of time. Using several data streams combined together helps to improve understanding. Looking at only one data stream could not be enough or misleading for making conclusions. For instance, if the CO level rises up very quickly with no activity, it would require immediate attention from the administrators, while confirmed with another parameter it might validate a normal activity. The system emphasizes that the correlation of parameters is important in the surveillance setting. As people may change their way of doing things, either by change in their habits or by an evolution of technology and equipment, a smart system must be able to take into account of these new factors.

Another aspect of multimedia data mining that has not been studied here is that it has been exceeding most storage capacity with the emergence of *big data* and requiring specific processing such as the parallelized computing techniques. Information has been known as a representation and an interpretation: this is exactly what we need to tackle. On one hand we have representation of multimedia datasets and on the other hand we have to deliver a consistent interpretation. Once we embark in the world of cognitive systems, we develop progressively the capacity for the computing system to find by itself the meaning of the data and take autonomous action. This is a new frontier for computing science that will open a new world of possibilities, challenges, opportunities, and risks.

References

1. Swan M (2012) Sensor Mania! The internet of things, wearable computing, objective metrics, and the quantified self 2.0. J Sens Actuator Netw 8:217–253
2. Chen M et al (2012) Machine-to-machine communications: architectures, standards and applications. KSII Trans Internet Inf Syst 6(2):480
3. Zaslavsky A et al (2012) Sensing as a service and big data. In: Proceedings of the international conference on advances in cloud computing (ACC), Bangalore, India, July 2012
4. Kumar P, Pandeya K (2013) Big data and distributed data mining: an example of future networks. Int J Adv Res Innov 2:36–39
5. Keshavarz A et al (2006) Distributed vision-based reasoning for smart home care. In: Proceedings of the 4th ACM international workshop on video surveillance and sensor networks, pp 145–154
6. Suryadevara NK et al (2011) Wireless sensors network based safe home to care elderly people: behaviour detection. In: Proceedings Eurosensors XXV. Athens, 4–7 September 2011

Chapter 7
Content-Based Privacy for Consumer-Produced Multimedia

Gerald Friedland, Adam Janin, Howard Lei, Jaeyoung Choi and Robin Sommer

Abstract We contend that current and future advances in Internet scale multimedia analytics, global inference, and linking can circumvent traditional security and privacy barriers. We therefore are in dire need of a new research field to address this issue and come up with new solutions. We present the privacy risks, attack vectors, details for a preliminary experiment on account linking, and describe mitigation and educational techniques that will help address the issues.

7.1 Introduction

The growth of multimedia as demonstrated by social networking sites such as Facebook and YouTube combined with advances in multimedia content analysis (face recognition, speaker verification, location estimation, etc.) provides novel opportunities for the unethical use of multimedia. In small scale or in isolation multimedia analytics have always been a powerful but reasonably contained privacy threat. However, when linked together and used on an Internet scale, the threat can be enormous and pervasive. The multimedia community therefore has an *obligation* to understand these risks, mitigate the effects, and educate the public on the issues.

Imagine a future where multimedia query engines *just work*. You can search by topic, location, person, camera identity, and time—even when the uploader did not

G. Friedland (✉) · A. Janin · H. Lei · J. Choi · R. Sommer
International Computer Science Institute, 1947 Center Street, Suite 600,
Berkeley, CA 94704-1159, USA
e-mail: fractor@icsi.berkeley.edu

A. Janin
e-mail: janin@icsi.berkeley.edu

H. Lei
e-mail: hlei@icsi.berkeley.edu

J. Choi
e-mail: jaeyoung@icsi.berkeley.edu

R. Sommer
e-mail: robin@icir.org

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_7

explicitly include such information. An unscrupulous attacker could query for videos recently recorded at resorts and then find videos taken with the same camera in nearby wealthy residential neighborhoods. This would produce an ideal "hit list" of targets who are likely away from home, which the thief could then refine. As reported in previous work (see Sect. 7.2), cybercasing already occurs, but with a multimedia query engine, simple methods of anonymizing posts and suppressing metadata will no longer be enough. Rather, the multimedia community must work to educate the public about the risks of inferencing at the Internet scale, invent methods to identify when information (such as the "identity" of the camera) is being unintentionally leaked, and develop mitigation techniques to reduce the potential harm.

After defining the topic and presenting prior work (Sect. 7.2), we outline existing and future multimedia content analysis and linking techniques that could support unethical use and describe possible attack vectors (Sect. 7.5). Next, we describe some preliminary experiments providing evidence that multimedia analytics can circumvent one aspect of privacy by linking accounts (Sect. 7.7). Finally, we outline mitigation and educational techniques (Sect. 7.12) and conclude that this is a new topic to be explored (Sect. 7.15).

7.2 Definition and Prior Work

Privacy is a concept that is hard to define. As a consequence, many definitions exist, including "privacy is the right to be left alone" [1] and more modern definitions, such as U.S. President Barak Obama's "Framework for Protecting Privacy" [2]. Merriam Webster defines privacy as "(a) the quality or state of being apart from company or observation and (b) freedom from unauthorized intrusion" [3]. While all of the definitions aim at the same goal, they are too broad for our engineering purposes. Therefore, in this paper we restrict ourselves to a more technical definition. We define privacy as "practically securing the implications of communication", which sets it apart from the field of *secure communication*, which is "securing the properties of the communication itself" [4] through methods of cryptography, steganography, identity hiding, and other well-known computer science topics. In other words, our privacy research is not about securing a communication line between several parties; it is to make sure that publicly available information conveys only the data the author intended. We acknowledge that this goal, like the aims of secure communication, will most likely never be achieved perfectly. However, improvements in methods can make communication "more private". Given that even our narrower definition is still a very broad goal, we will limit ourselves to attack vectors that pose an actual criminal threat and/or directly influence life-changing decisions.

While the scientific community has investigated correlation between different data sets in terms of privacy implications, most of these efforts have focused on de-anonymizing or compromising a single data set with the help of auxiliary information. Except for the few exceptions described below, efforts have mostly concentrated on structured data, ignoring multimedia content analysis.

7.3 Work on Structured Data

In 1997, Sweeney [5] showed that anonymously published medical records can be de-anonymized when correlated with external data, triggering a large body of follow-up work on designing anonymous statistical databases as well as understanding their limitations [6–10].

More relevant to the multimedia community, Narayanan et al. present an algorithm and proof for de-anonymizing sparse datasets [11]. They apply their algorithm to anonymized Netflix movie ratings: given knowledge of a subset a person has rated (e.g. learned from a lunch conversation or public ratings), the system is able to identify *all* movies in the database that the user has rated. In [12], the same idea is used to de-anonymize a social network graph by leveraging a graph from a second network with real identities as auxiliary data. Researchers from Parc investigated inference using web search engines in order to analyze whether anonymized (or obfuscated) private documents that are going to be released publicly can be de-anonymized [13, 14]. They do not consider multimedia content nor inference between information that is already publicly available.

Griffith et al. [15] correlate public birth, death and marriage records from the state of Texas to derive the mother's maiden name of more than 4 million Texans. Balduzzi et al. [16] automatically query 8 social networks with a list of 10 million e-mail addresses to retrieve the associated user profiles. They then correlate that profile information across the networks and are able to identify mismatches between them. (i.e. they find users who chose different names, age, etc. in different networks). More generally, Bishop et al. [17] discuss the need to go beyond "closed worlds" when sanitizing a data set and consider external knowledge explicitly.

With geo-location information being a popular key to image and video retrieval, another area of related research is locational privacy. The Electronic Frontier Foundation published an overview of locational privacy aspects [18]. Locational privacy in vehicular systems, e.g. toll collection, is addressed in [19, 20]. Zhong et al. [21] present protocols for secure privacy preserving location sharing. The upcoming HTML 5 standard will include APIs to query a client's location. The *Cree.py* [22] application uses geolocation data from social networks and media hosting services to track a person's movements.

Several web sites highlight the potential of information leakage users might not be aware of:

`Sleeptime.org` estimates sleep patterns of Twitter users.

`Stolencamerafinder.co.uk` crawls for digital camera serial numbers in online photos in order to find pictures taken with stolen cameras.

`Icanstalku.com` published geotags found in tweets.

`pleaserobme.com` used status updates form social networks to locate users who were currently not at home but had published their home address.

7.4 Work on Multimedia Data

The above section was presented to outline current work on structured data. History has shown that work on multimedia data follows in the footsteps of structured data with a delay (for example, work on compression, messaging capabilities, or even World Wide Web content itself). As a result, we see an initial growth in multimedia articles that present work on privacy. We see this early work as evidence for our hypothesis of a new field of research.

In a recent effort [23], we analyzed the privacy implications of *geotagging*, i.e. high-accuracy location information attached as meta-data to audio, image, and video files. Specifically, we examined the risk that such geotags pose for what we termed "cybercasing": using online data and services to mount real-world attacks. Moreover, we showed that geo-tags are not needed as they can be replaced by multimedia analytical location estimation techniques [24].

In [25] Lukas et al. propose a method for the problem of digital camera identification from images based on the sensor's pattern noise. For each camera under investigation, they first determine its reference pattern noise, which serves as a unique identification fingerprint. This is achieved by averaging the noise obtained from multiple images using a denoising filter. To identify the camera from a given image, they consider the reference pattern noise as a spread-spectrum watermark, whose presence in the image is established by using a correlation detector. Experiments on approximately 320 images taken with nine consumer digital cameras are used to estimate false alarm rates and false rejection rates.

Many researchers have worked on automatic video blurring (for example [26–28]); however, [29] showed that many of the proposed techniques are not effective. In response to this problem, [30] has presented an initial framework to validate video privacy.

7.5 Privacy Risks and Possible Attacks

In this section, we describe some existing and future multimedia analytic techniques that pose a privacy risk including how these risks could be exploited. This is by no means an exhaustive list.

Location Estimation. Multimedia location estimation formed the genesis of our interest in privacy in multimedia, and was reported in previous work (see Sect. 7.2). Using multimodal methods, state-of-the-art algorithms can estimate the location of about 40 % of Flickr videos with an accuracy better than 100 m, and over 50 % with an accuracy better than 1 km. This extends the amount of exactly trackable multimedia by a significant factor without requiring actual GPS sensors.

Time Estimation. The date and time that a multimedia document was recorded can be estimated using cues such as sun location or measuring shadow lengths. More powerfully, if you can determine that Video A was recorded at the same time and place as Video B, and you know or can infer Video A's time, you now know Video B's time. Just excluding time/date metadata from *your* vacation video does not protect you if somebody else includes it in theirs.

Person Detection. In the image realm, this is usually known as face detection; in audio, speaker recognition. While the uploader can take active methods to anonymize the foreground participants if privacy is an issue (e.g. replacing their face with a black box, replacing their audio with a bleep sound), the privacy of background participants is problematic because the uploader may not care about incidental privacy breaches of the background participants.

Object Detection. Detecting an iphone in a person's hand might make them a more desirable robbery target. Marketers could target people based on the furniture quality in the background of a video. Note that mitigation techniques are particularly problematic with object detection, since one cannot simply remove *all* objects from a multimedia document without severely impacting the document's content.

Environmental Acoustic Noise. Uploaders often recognize the need to obscure faces. However, when recording video data they often forget that the audio track includes a unique signature that might break their anonymity. This has been shown in several studies, including our previous work (Sect. 7.2). Also, the combination of such linking methods with other methods such as location estimation leads to even more powerful privacy invading possibilities.

Sensor Detection. It is already possible to narrow down or even uniquely identify what camera was used to record a video or what microphone was used to record audio based on the artifacts of the sensor. For example, pixel noise is unique to a particular camera; the exact frequency response of a microphone might be used to narrow down the possible microphones. This provides a whole new avenue of linking, completely bypassing other means of anonymization.

3D Recordings. Time-of-flight cameras, light field camera, stereo cameras, and microphone arrays are all becoming more pervasive. It is clear that similar devices will continue to be developed. Each comes with its own sets of issues, and have the potential to capture even more unwanted data. Since this trend will only accelerate, it is necessary for the multimedia community to address these issues.

Exotic Sensors. Everything from air pressure sensors to heart rate monitors are becoming more common, and it is likely data from these sensors will be incorporated into multimedia documents much as GPS is now. Since users often have no real notion of what is being collected or how accurate it is, they have little or no intuition on the privacy implications. A prominent historic example is GPS—it was only recently that the profound privacy implications of geotagging became commonly known.

We outline a small number of specific attacks that can now or could shortly be used to invade privacy in detrimental ways using Internet scale multimedia analytics and linking.

Today, one can readily access much of the *structured* information available online via programmatic interfaces: major services like Google, Facebook, Twitter, Flickr, YouTube, and LinkedIn all offer extensive APIs that make automatic retrieval trivial. These APIs often offer more comprehensive access than the corresponding web interface, and their availability is the primary driver behind the wide range of 3rd party "apps" that constitute a key part of today's social networking space.

We contend that as multimedia retrieval technology matures, it will eventually become part of such APIs, making the capabilities available to everybody able to write a few lines of Python code. For example, Google already provides simple forms of image and video search, and rumor has it that face recognition is ready for mass deployment as part of their Goggles service. Facebook has already integrated face recognition into their platform, and though it is not yet exposed via the Facebook API, third party companies such as face.com are already providing programmable access to face recognition of Facebook content.

Having large-scale multimedia retrieval at one's fingertips provides an opportunity for amazing next-generation online services. However, we believe that it will also open up a new dimension of privacy threats that our community has not yet understood.

The availability of Internet-scale multimedia retrieval capabilities allows a wide range of attacks that threaten users' privacy. Whereas today's search queries remain limited to mostly textual information, attackers will eventually query for audio and video *content*. Criminals could leverage that to reliably locate promising targets. For example, they may first identify individuals owning high-value goods within a target area and then pinpoint times when their victims' homes are unattended.

Another threat is background checks becoming much more invasive than today: many companies have strong incentives to examine their customers' private life for specifics impacting business decisions. An insurer, for example, might refuse payment to a customer receiving disability where the insurer finds Facebook photos of the customer skiing. Likewise, an employer seeking new hires might check a candidate's Twitter followers for potentially embarrassing information that could be used against the company in the future and refuse to hire such candidates.

A whole new realm of marketing techniques are enabled by multimedia retrieval and linking. A company could extract all videos of people wearing branded merchandise, cluster them by location and time, and target that location for direct marketing. The privacy implications of such broad and automatic analysis have been insufficiently studied.

The new capabilities make *stalking* easier by providing the means to not only quickly locate the victims, but also profile their typical behavior patterns, friends, relatives, and acquaintances.

7.6 Example

In this section, we will exemplify the power of multimedia retrieval in combination with structured-data retrieval in a mockup scenario adopted from [31].

Consider the following business: Fred works for Schooner Holdings and wishes to gain (possibly illicit) inside information on future profits at the chipmaker Letin. Fred hires Eve, who runs an "expert network". Eve puts Fred in touch with Bob, a

Letin employee. In the process of consulting for Fred, Bob is encouraged to reveal information about Letin's upcoming products.[1]

Currently, the greatest limit on this process is Eve *finding* experts like Bob who (perhaps unknowingly) possess potential insider information and are willing to act as consultants. Eve would greatly improve her business if she could find "corruptibles": individuals in the business of interest who might be favorable to legitimate or illegitimate offers.

Thus Eve starts searching social networks for individuals who are compatible with her desired level of (il)legality. She instructs her crawler to begin with LinkedIn and web searches, crawling the names and contact information for personnel at companies of interest.

Then her crawler shifts to Facebook, Twitter, other social networks, and blogs, beginning with all candidates found in the first pass. This crawler does not just look at the candidates but also at friends of candidates.

She also searches any media, including images and videos, for links to other people that the social network might not provide directly. Face recognition for example can provide probable connections to other profiles. She also examines media for any compromising material, such as illegal acts, drug paraphernalia, or party photos. Eve knows that her automated content analysis does not need to be perfect: she leverages crowdsourcing services like Mechanical Turk [33] to validate potential candidate matches using human labor at a very low cost.

Eve's crawler also queries further public and semi-public records. There are commercial services that map an email address to a mailing address. Her crawler uses these to discover where candidates live and how much their property is worth (e.g. by using Zillow.com's access to property tax data and sales history).

With all this data, Eve's crawler can now create "inference chains" which estimate the probability that any given candidate in her set has a potential weakness, enabling Eve to search for possible points of corruptibility. An individual who is dating someone with a reputation as a gold digger, or who purchased their house at the height of the real estate bubble, might have financial problems. Such candidates could be honestly corrupted by offering consulting positions, allowing Eve to expand her expert network.

Eve might also contract with those operating outside the law. Then blackmail becomes an attractive option, especially if considering guilt by association. Someone with a security clearance may be vulnerable if his associates are drug abusers, or if he is having an affair that can be inferred through social patterns.

Nothing in the preceding scenario is unrealistic: every step Eve takes can be constructed using today's technology. It is simply a matter of putting all the pieces together to collect and analyze the reams of data which exist on today's social networks and other databases.

[1] In many countries, this practice is possibly illegal but exists in a gray area and is seemingly routine practice. The Galleon insider trading trial [32] was based largely on the use of expert network consultants.

Unfortunately, there is also hardly any protection in place against somebody like Eve. Furthermore, while structured data still plays a dominant role in this scenario, it is easy to see how multimedia data will blur the boundaries even more. For example, if we assume that face recognition technology reaches close to perfection, user names will no longer provide a boundary as long as a face photo is part of the website. Moreover, speaker recognition, location estimation, and other techniques described in Sect. 7.9 will add even more possibilities. Finally, note that the methods need not be perfect—Eve needs only a small number of likely hits to follow up on to allow nefarious actions to proceed

7.7 Preliminary Experiments

This section presents technical details on a preliminary experiment on matching user accounts based on consumer-produced videos, demonstrating that multimedia retrieval can circumvent traditional security and privacy barriers, such as the assumption that different account names will separate the same persona.

Consider the following scenario: A professor at a University is proud to present lectures to a very large audience on a public video distribution site. These lectures contain her voice, her face, and her name in the credits, making their authorship anything but anonymous. At the same time, she is dating online and follows the dating site's suggestion to provide introduction videos of herself to make her profile more personable. The suggestion comes with the assurance that, unless the author of the introduction video identifies herself, the video will remain anonymous.

In the following, we will provide evidence that this promise of anonymity is hard to keep in the face of increasingly accurate multimedia retrieval technologies.

7.8 Dataset

We begin by describing the data sets used in this experiment. The audio tracks are extracted from the videos distributed as training and test sets for the Placing Task of MediaEval 2011 [34], a multimedia benchmark evaluation. The Placing Task involves automatically estimating the location of each test video using one or more of: metadata (e.g. textual description, tags), visual/audio contents, and social information. The videos are not pre-filtered or pre-selected in any way to make the data set more relevant to the user-verification task, and are therefore likely representative of videos selected at random.

A total of 10,857 Creative Commons licensed Flickr videos, uploaded by 2,943 Flickr users, were used in our experiments. Flickr requires that an uploaded video must be created by its uploader (if a user violates this policy, Flickr sends a warning and removes the video). This policy generally ensures that each uploader's set of videos is "personal" in the sense that they were created by the same person and therefore likely have certain characteristics in common, such as editing style, recording device, or frequently recorded scenes/environments, etc.

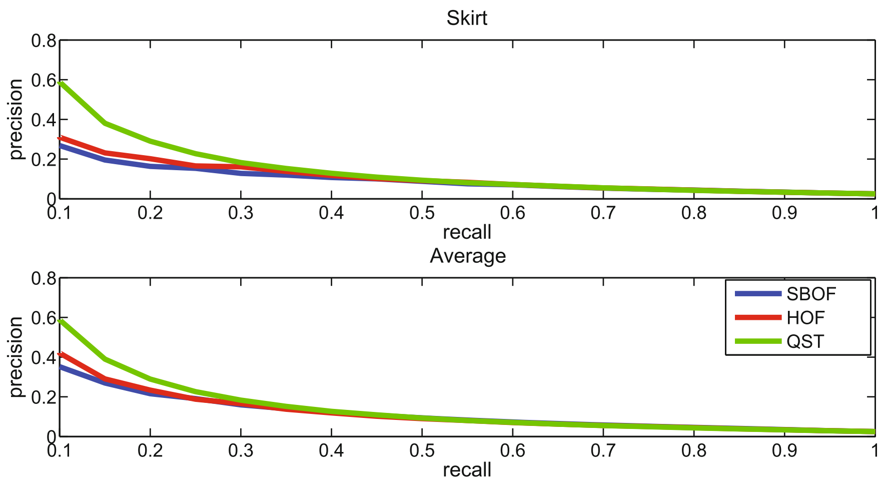


Fig. 7.1 A histogram visualizing the duration of the videos of the data set used in our experiments

From a by-hand examination of 123 short-duration videos from the data set, we found that most of videos' audio tracks are quite "wild". 59.3 % of the videos are home-video style with ambient noises. 47.2 % of the videos had heavy ambient noises such as crowds chatting in the background, traffic noise, and wind blowing into microphone. 25.2 % of the videos contained music, either played in the background of the recorded scene, or inserted at the editing phase. 59.3 % of the videos did not contain any form of human speech at all, and even for the ones that contained human speech, 64 % were from multiple subjects and crowds in the background speaking to one another, often at the same time. Although we found that 10.5 % of videos contained audio of the person behind the camera, there is no guarantee that the owner of the voice is the actual uploader; it is possible that all videos from the same uploader were recorded by different people (such as family members).

Figure 7.1 displays a histogram of the lengths of the 10,857 videos used in our dataset. All videos are limited to 90, accounting for the peak at 90. 71.8 % of videos have less than 50 of playtime, while 50 % have less than 30 of playtime.

7.9 Technical Approaches

This section describes the multimodal user verification experiments based on audio and a set of five visual features. Note that the task of user verification is to determine if two videos are uploaded by the same user or different users. The i-vector-based approach [35], which is currently the state-of-the-art in the field of speaker recognition, is used to perform classification and audio-visual feature combinations. The approach involves extracting a set of low-dimensional vectors to represent the user identity of each video. The vectors can be derived from either the audio or visual features.

To extract the audio-based low-dimensional vectors, which are known as the i-vectors in [35], a total variability matrix T is first trained to model the variability (both user-, acoustic environment-, and acoustic channel-related) of the high-dimensional Baum-Welch statistics obtained from the MFCC $\mathrm{C0} - \mathrm{C19} + \Delta + \Delta\Delta$ (60 dimensions total) audio feature vectors of each video. The matrix acts as a projection matrix used to obtain the low-dimensional vectors, which characterize the user of each video based on its audio. Specifically, for each video, the audio track is first extracted, and a vector of first-order Baum-Welch statistics M of the audio feature vectors, centered around the means of a GMM world model, is obtained. The statistics can be decomposed as follows:

$$M = m + T\omega \tag{7.1}$$

where m is the GMM world model mean vector, and ω is the low-dimensional vector. The GMM contains 1,024 mixtures, and each mixture contains 60 mean dimensions corresponding to the dimensionality of the MFCC features. Hence, the total dimensionality of M is 61,440, which the T-matrix projects onto a set of 400 dimensions to form the low-dimensional audio-based vectors.

The visual-based low-dimensional vectors are obtained from the result of a Principal Components Analysis (PCA) projection of a set of pre-extracted visual features onto a small set of its eigen-dimensions. The visual features are extracted using the open source library LIRE [36]. The features used include Tamura (TAM), Gabor (GAB), Auto Color Correlogram (ACC), Color and Edge Directivity Descriptor (CEDD), and Fuzzy Color and Texture Histogram (FCTH). The TAM feature is a texture-based feature. For our experiments, 24 dimensions are used to represent the low-dimensional vectors for the GAB, ACC, CEDD, and FCTH features, and 12 dimensions are used for the TAM feature.

The audio and visual features are combined by concatenating the corresponding low-dimensional vectors (in this way, the combined-feature experiments use more parameters than the standalone-feature experiments). The system performs a Within-Class Covariance Normalization (WCCN) [37] on the resulting vectors, which whitens their covariance via a linear projection matrix. A generative Probabilistic Linear Discriminant Analysis (pLDA) [38] log-likelihood ratio is then used to obtain a similarity score between the low-dimensional vectors of each training and test video. The generative pLDA log-likelihood ratio for similarity score computation is shown below:

$$\begin{aligned}\mathrm{score}(\omega_1, \omega_2) = \log N\left(\begin{bmatrix}\omega_1\\ \omega_2\end{bmatrix}; \begin{bmatrix}\mu_1\\ \mu_2\end{bmatrix}, \begin{bmatrix}\Sigma_{\mathrm{tot}} & \Sigma_{\mathrm{bc}}\\ \Sigma_{\mathrm{bc}} & \Sigma_{\mathrm{tot}}\end{bmatrix}\right)\\ - \log N\left(\begin{bmatrix}\omega_1\\ \omega_2\end{bmatrix}; \begin{bmatrix}\mu_1\\ \mu_2\end{bmatrix}, \begin{bmatrix}\Sigma_{\mathrm{tot}} & 0\\ 0 & \Sigma_{\mathrm{tot}}\end{bmatrix}\right)\end{aligned}$$

where ω_1 and ω_2 are the vectors for a pair of training and test videos, $N(\cdot)$ is the normal Gaussian probability density function, and Σ_{tot} and Σ_{bc} are the total and between-class scatter matrices computed from the training vectors. Hence, one user-similarity score is obtained for each training versus test video using the above approach.

The Brno University of Technology's (BUT's) Joint Factor Analysis Matlab demo [39] is used to assist in the system development, and the open-source ALIZE toolkit [40] is used to train the UBM. The HTK Library [41] is used for MFCC feature extraction.

7.10 Experiments and Results

A set of 1,268 Flickr users in the corpus were designated as training users, and 2,851 were designated as test users, with roughly 1,200 users in common with the training users. Each training user is associated with one video in the training set, and 4,869 videos are associated with the 2,784 test users. Overall, a set of 6,251 videos were used for training and testing. A separate set of 146 users with 4,605 videos were used to train the T-matrix, PCA projection matrices, and the total and between-class scatter matrices used in the system. 2,302 videos from the 146 users were used to train the GMM world model. A total of 6 million similarity scores were computed between video pairs from the training and test users, with 3,385 of the scores from pairs with the same user. Table 7.1 shows the Equal Error Rate (EER), and the Miss Rates at 1 % and 0.1 % False Positive (FP) rates for the 6 million scores of the system. A Miss occurs when a pair of same-user videos are classified as having different users, and a FP (false positive) occurs when different-user videos are classified as having same users, given a particular scoring threshold. For the Miss rate at 1 % FP, the threshold is set such that 1 % of the different-user pairs are classified as having same users. User verification results for both audio and visual features, standalone and in combination, are shown. Also shown are the number of dimensions used in the low-dimensional vectors used to compute the user-similarity scores for each feature, or combination of features.

Results in Table 7.1 indicate that the audio-based MFCC feature has the best standalone performance—26.1 % EER, 65.6 % Miss at 1 % FP, and 86.6 % Miss at 0.1 % FP. If the MFCC features are combined with the top-four standalone visual features in terms of EER (ACC,CEDD,GAB,FCTH, and TAM), then the performance improves to 24.0 % EER, 59.2 % Miss at 1 % FP, and 78.4 % Miss at 0.1 % FP. This represents an 8.0 % relative EER improvement, a 9.8 % relative improvement of Miss at 1 % FP, and a 9.5 % relative improvement of Miss at 0.1 % FP. The results demonstrate the effectiveness of combining the audio and visual modalities for this task. The standalone visual features perform significantly worse than the MFCC feature.

Table 7.1 User matching results for audio and visual features standalone and in combination

Feature	EER (%)	Miss at 1 % FP (%)	Miss at 0.1 % FP (%)	Vector dims
ACC	35.1	84.9	96.0	24
CEDD	35.0	82.1	91.4	24
FCTH	34.9	82.2	91.5	24
GAB	44.4	97.3	99.6	24
TAM	33.9	87.6	98.8	12
GAB+CEDD+ACC+FCTH+TAM	33.0	76.6	91.5	108
ACC+CEDD+FCTH+TAM	32.4	74.9	89.7	84
MFCC	26.1	65.6	86.6	400
MFCC+ACC+CEDD+GAB+FCTH+TAM	24.1	60.0	79.3	508
MFCC+CEDD+ACC+FCTH+TAM	24.0	59.2	78.4	484

Similarity scores were computed on 6 million pairs of videos, with a total of 1,268 training users and 2,784 test users as described in Sect. 7.10

7.11 Summary of Experimental Results

The outcome of the above experiment for user matching is certainly not yet a reason for panic as user matching based on content is still very preliminary. However, given that our best approach was able to match random, short consumer-produced videos with an equal error rate of 24 % (compared to 50 % for chance), it means that a future can be foreseen where attacks like this become feasible. Moreover, many attacks are not targeted at matching one particular user. When finding victims from a large pool, the miss and false alarm rates are more important. The above experiments show that at 1 % false alarm, we would only miss about 60 % of the true positives. Given a scenario where the 1 % false alarm does not represent many videos, one can search through the 40 % of the non-missed true positives for a pair of videos containing the same user uploader.

7.12 New Topics for Research

Countering the attacks described above is not straight-forward since filtering out sensitive information from audio and video content is fundamentally harder than with structured text data. We therefore propose a new topic in multimedia devoted to considering both privacy research as well as education.

7.13 Mitigation Research

A major challenge for conserving privacy in consumer produced videos is the development of methods to identify the foreground information information that the user considers important from the background information. It is this background data that has the highest risk of incidentally leaking private information.

We believe that machine learning will play a key role in detecting such unnoticed information leaks. For example, one can label who is an "extra" in a movie by the number of times they appear and the number of lines they speak. The extras form the semantic background to the movie—they are noticeable, but not directly relevant. A machine learning algorithm could use "star" versus "extra" as ground truth, and learn models to distinguish the two. Applied to consumer-produced videos, the system could then identify foreground versus background participants using the trained model.

Once the information that is breaking privacy is identified, it must also be removed or distorted sufficiently to reduce the threat. This is difficult with most existing multimedia analysis algorithms, since they are statistical in nature. If we understood the specific cues the statistical methods learn, we could obscure those cues, hopefully without distorting the rest of the content. For example, if the background semantic "bird call of a Nene" is detected, you are leaking location information (Hawaii). Just damping that sound may be enough to obscure the location. This sort of cue detection is in the nascent stages for some methods (e.g. concept detection as in TrecVID MED), and nearly non-existent for others. It is incumbent on the multimedia community to develop an understanding of the cues so that mitigation techniques can be developed.

For other methods, more direct mitigation may be possible. For example, an upload tool could blur semantically background faces in a video (however, this might not be enough, see also discussion in Sect. 7.2). A query tool could refuse to perform speech recognition and indexing on background voices. This would be very similar to today's common practice for copy machines to refuse the copying of bank notes. A key component of such a system would be to ensure, possibly with the interaction of the uploader, that foreground content is not compromised.

7.14 Education on Privacy

Independent of any technological protection, we believe a key ingredient to comprehensive mitigation must be *education*. University electrical engineering and computer science curricula usually include an abundance of material on how to improve retrieval based on the underlying multimedia content analysis but only rarely talk about the negative impacts of these technologies. Privacy content is mostly limited to traditional topics in secure communications such as steganography, encryption, and other well-known techniques and/or even removed from consideration, as ethical concerns are considered not to be part of engineering. Therefore, even when acknowledged as a problem, many new technologists lack the knowledge of how to react to society's concerns and even mitigate easy-to-address risks. An argument often heard from students is: "We'll deal with privacy and social issues later—right now we need to focus on development." The truth, however, is that, for example, if privacy and security had been a concern in the early stages of developing the Internet, many of today's issues, such as spam and phishing email, would most likely be much less of a problem. Undergraduate and graduate engineering education curriculums should therefore include a strong component on privacy that makes future technologists aware of the societal implications of their research and development.

The second line of education should concern users, especially young people. Among the groups most affected by privacy concerns are high-school students [42]. They are the most frequent users of social-networking sites and apps, but often do not have a full understanding of the potential consequences their current online activities might have later in their lives. For example, a Facebook posting that a high-schooler's friends think is "cool" might be seen by a much larger audience than she or he expected—including perhaps future employers who wouldn't agree with the high-schooler's judgement. In addition, not understanding—or not thinking about—the consequences of posting often leads to oversharing information about other people, including friends and relatives. Consequently, users can take steps to protect themselves once they realize the power that modern content analysis tools yield in the hands of adversaries. They might then even choose not to post certain content in the first place.

Figure 7.2 shows a preliminary mockup for a teaching tool that we created as part of a project for social media privacy education for teenagers [43]. The input for the web-based tool is an arbitrary image that has been published on the web. The image

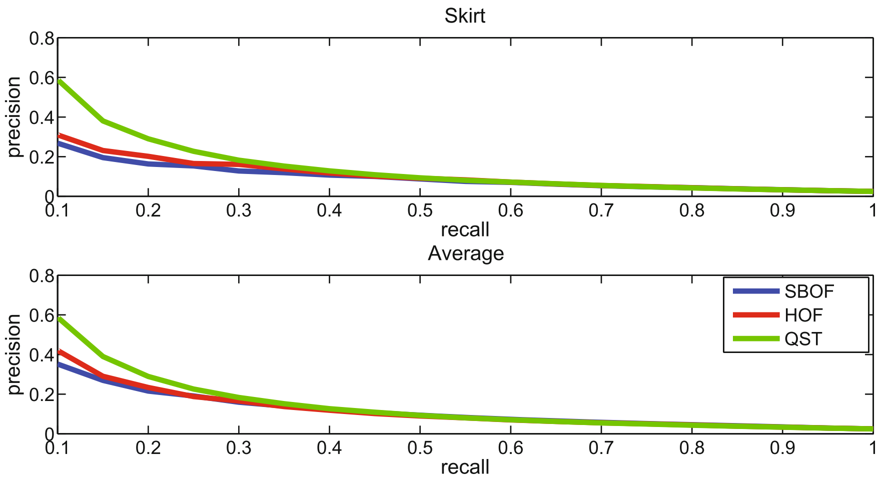


Fig. 7.2 Education is part of the new topic. A mockup of an educational browser tool showing that online image often includes meta-data that allows inference beyond the content of the image

is than analyzed for EXIF data. If found, the data is displayed textually. Furthermore, if the EXIF data contains geo-tags, the location for the image is shown on a map and all Twitter feeds that belong to that location are also shown. We saw that people are often shocked, how much information an indoor image like the one shown conveys and at the same time, how much can be inferred from that location, e.g. when a photo that does not contain any faces actually maps back to their own twitter feeds.

Building effective educational components that transfer knowledge on privacy protection and the consequences of multimedia retrieval to younger adults who are not yet capable of understanding deep research results constitutes a new domain for research. Here, educational research needs to team up with HCI and other multimedia-related fields to attack this part of the new topic. The question is how to enable educators to master an up-to-date, scientifically-informed understanding of privacy, without having to rely on (often exaggerated) newspaper articles.

7.15 Conclusion

The growth of multimedia as demonstrated by social networking sites such as Facebook and YouTube combined with advances in multimedia content analysis (face recognition, speaker verification, location estimation, etc.) provides novel opportunities for the unethical use of multimedia. The article surveyed the field and showed that awareness of the issue is focused on structured data but does not extend to multimedia retrieval. Using a scenario, a taxonomy of attacks, and a preliminary experiment, we outlined how multimedia retrieval adds a new quality to privacy and security research. We believe that mitigation is both a question of research as well as education. In summary, we believe the diversity of attacks and the complexity of solving the privacy issues with multimedia content will require creative thinking of a community of researchers and therefore spawn a new field in multimedia content analysis. We believe web-scale multimedia privacy is not only a new topic, but also a necessary new field.

Acknowledgments This material is based upon work supported by the National Science Foundation under Grant No. CNS-1065240. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank Michael Ellsworth for his rewording suggestions.

References

1. Prosser D (1960) Privacy. In: California law review, vol 48, p 383
2. House TW (2012) Consumer data privacy in a networked world. http://www.whitehouse.gov/sites/default/files/privacy-final.pdf
3. Webster M (2013) Privacy. http://www.merriam-webster.com/dictionary/privacy
4. Wikipedia (2013) Secure communication.http://en.wikipedia.org/wiki/Secure_communication

5. Sweeney L (1997) Weaving technology and policy together to maintain confidentiality. J Law Med Ethics 25:2–3
6. Aggarwal C (2005) On k-anonymity and the curse of dimensionality. In: Proceedings of the international conference on very large data bases
7. Dinur I, Nissim K (2003) Revealing information while preserving privacy. In: ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems (PODS)
8. Dwork C (2006) Differential privacy. In: 33rd international colloquium on automata, languages, and programming (ICALP)
9. Dwork C (2008) Differential privacy: a survey of results. In: Agrawal M, Du D, Duan Z, Li A (eds) Theory and applications of models of computation., Lecture notes in computer scienceSpringer, Berlin, pp 1–19
10. Sweeney L (2002) k-anonymity: a model for protecting privacy. J Uncertain Fuzziness Knowl-based Syst 10(5):557–570
11. Narayanan A, Shmatikov V (2008) Robust de-anonymization of large sparse datasets. In: Proceedings of the IEEE symposium on security and privacy
12. Narayanan A, Shmatikov V (2009) De-anonymizing social networks. In: Proceedings of the IEEE symposium on security and privacy
13. Chow R, Golle P, Staddon J (2008) Detecting privacy leaks using corpus-based association rules. In: Proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining
14. Staddon J, Golle P, Zimny B (2007) Web-based inference detection. In: Proceedings of 16th USENIX security symposium
15. Griffith V, Jakobsson M (2005) Messin' with texas deriving mother's maiden names using public records. In: Proceedings of the international conference on applied cryptography and network security (ACNS)
16. Balduzzi M, Platzer C, Holz T, Kirda E, Balzarotti D, Kruegel C (2010) Abusing social networks for automated user profiling. In: 13th international symposium on recent advances in intrusion detection, 09 RAID'2010
17. Bishop M, Cummins J, Peisert S, Singh A, Bhumiratana B, Agarwal D, Frincke D, Hogarth M (2010) Relationships and data sanitization: a study in scarlet. In: Proceedings of the workshop on new security paradigms
18. Blumberg A, Eckersley P, On locational privacy, and how to avoid losing it forever. Electronic frontier foundation
19. Hoh B, Gruteser M, Herring R, Ban J, Work D, Herrera J-C, Bayen AM, Annavaram M, Jacobson Q (2008) Virtual trip lines for distributed privacy-preserving traffic monitoring. In: Proceeding of the 6th international conference on mobile systems, applications, and services MobiSys'08
20. Popa RA, Balakrishnan H, Blumberg A (2009) VPriv: protecting privacy in location-based vehicular services. In: Proceedings of the USENIX security symposium
21. Zhong G, Goldberg I, Hengartner U (2007) Louis, lester and pierre: three protocols for location privacy. In: Proceedings of the privacy enhancing technologies symposium
22. H-Security Cree.py application knows where you've been. http://www.h-online.com/security/news/item/Cree-py-application-knows-where-you-ve-been-1217981.html
23. Friedland G, Sommer R (2010) Cybercasing the joint: on the privacy implications of geo-tagging. In: Proceedings of the USENIX workshop on hot topics in security, August 2010
24. Friedland G, Choi J (2011) Semantic computing and privacy: a case study using inferred geo-location. Int J Semant Comput 5(01):79–93
25. Lukas J, Fridrich J, Goljan M (2006) Digital camera identification from sensor pattern noise. IEEE Trans Inf Forensics Secur 1(2):205–214
26. Dufaux F, Ebrahimi T (2006) Scrambling for video surveillance with privacy. In: computer vision and pattern recognition workshop. CVPRW'06. In: Proceeding of the conference on, pp 160–160
27. Fan J, Luo H, Hacid M-S, Bertino E (2005) A novel approach for privacy-preserving video sharing. In: Proceedings of the 14th ACM international conference on information and knowledge management, CIKM'05. ACM, New York, pp 609–616

28. Koshimizu T, Toriyama T, Babaguchi N (2006) Factors on the sense of privacy in video surveillance. In: Proceedings of the 3rd ACM workshop on continuous archival and retrival of personal experiences, CARPE'06. ACM, New York, pp 35–44
29. Neustaedter C, Greenberg S, Boyle M (2006) Blur filtration fails to preserve privacy for home-based video conferencing. ACM Trans Comput-Hum Interact 13(1):1–36
30. Dufaux F, Ebrahimi T (2010) A framework for the validation of privacy protection solutions in video surveillance. In: IEEE international conference on multimedia and expo (ICME), pp 66–71
31. Friedland G, Maier G, Sommer R, Weaver N (2011) Sherlock Holmes evil twin: on the impact of global inference for online privacy. In: Proceedings of the new security paradigms workshop (NSPW), September 2011
32. Cohan P (2009) Why executives risk their job to tip a hedge fund. http://meet-the-street.blogspot.com/2009/10/expert-networks-what-every-iro-needs-to.html,
33. Amazon.com Mechanical Turk, https://www.mturk.com/mturk/welcome
34. Mediaeval web site. http://www.multimediaeval.org
35. Burget L, Oldřich P, Sandro C, Glembek O, Matějka P, Brümmer N (2011) Discriminantly trained probabilistic linear discriminant analysis for speaker verification. In: Proceedings of ICASSP, Brno, Czech Republic
36. Mathias L, Chatzichristofis SA (2008) Lire: Lucene image retrieval—an extensible Java CBIR library. In: Proceedings of the 16th ACM international conference on multimedia, pp 1085–1088, October 2008
37. Hatch AO (2006) Generalized linear kernels for one-versus-all classification: application to speaker recognition. In: Proceedings of ICASSP, Toulouse, France
38. Ioffe S (2006) Probabilistic linear discriminant analysis. In: Proceedings of ECCV, pp 531–542
39. Joint factor analysis matlab demo. http://speech.fit.vutbr.cz/software/joint-factor-analysis-matlab-demo/
40. Bonastre J, Wils F, Meignier S (2005) Alize, a free toolkit for speaker recognition. In: Proceedings of the ICASSP, vol 1, pp 737–740
41. Hmm toolkit (htk). http://htk.eng.cam.ac.uk/
42. Facebook users by age.http://en.wikipedia.org/wiki/Facebook#Reception
43. Tools for teaching privacy to k12 and undergraduate students.http://teachingprivacy.icsi.berkeley.edu

Part III
Biometric Multimedia Data Processing

Chapter 8
Large-Scale Biometric Multimedia Processing

Stefan van der Stockt, Aaron K. Baughman and Michael Perlitz

Abstract The field of Biometrics analyses organic signals from people to identify or verify an identity using a combination of physiological, behavioural or cognitive characteristics such as voice, fingerprints, eye color, facial features, iris, handwriting or other characteristics. Large-scale biometric identification systems can benefit from modern optimisation, classification and parallel computation techniques to reduce cost and increase accuracy. This chapter discusses recent and novel developments by the authors in the approaches taken to enable large-scale biometric identification. The authors present an overview of different techniques to perform the tasks of search space reduction, feature selection and parallel processing of biometrics data. Topics covered are: support vector machines and hyperspace transformations for effectively searching extremely large fingerprint databases to identify individuals; evolutionary computing to perform efficient facial feature selection for identification purposes; and cloud and high-performance designs for biometric systems.

8.1 Introduction

Traditionally, the field of Biometrics applies pattern recognition and analyses organic signals from people to identify or verify an identity using a combination of physiological, behavioural or cognitive characteristics such as voice, fingerprints, eye color, facial features, iris, handwriting or other characteristics [1]. A biometric should maximise the following criteria:

- *Universal*: such that everyone has the signature
- *Unique*: such that true positives are maximised

S. van der Stockt (✉)
IBM Corporation, 70 Rivonia Road, Sandton, Johannesburg 2196, South Africa
e-mail: stefanvd@za.ibm.com

A.K. Baughman
IBM Corporation, 3039 Cornwallis Road, Research Triangle Park, NC 27709, USA
e-mail: baaron@us.ibm.com

M. Perlitz
IBM Corporation, 2300 Dulles Station Blvd, Herndon, VA 20171-6349, USA
e-mail: michael.perlitz@us.ibm.com

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_8

- *Acceptable*: such that society adopts the use
- *Permanent*: such that the trait minimally changes over time
- *Collectable*: such that the signal can be acquired

Biometric samples that are universal, unique, permanent, acceptable and collectable, or some combination thereof, are a mechanism to uniquely and accurately identify or verify people. Verification, or a one-to-one (1:1) match, involves the confirmation or authentication of a person's identity. Identification, or a many-to-many match (1:M or N:M) where M is the gallery size and N are the probe biometrics, requires the person's identity to be looked up in a database containing the biometrics of many individuals. Verification is considered easier to implement and is less challenging in large-scale systems [1].

One of the most well-known biometrics used for identification and verification is the fingerprint, which satisfies all of the required criteria above. Forensic scientists use fingerprints to identify and verify identities. The pattern on the skin at the tip of the finger phalanx is referred to as a fingerprint [1, 2]. Even fingerprints between identical twins are different [1]. The singularities in the ridge pattern of a fingerprint are known as minutiae [3]. Figure 8.1 shows an example of fingerprint minutia. The combination of fingerprint ridge flow, n-furcations, and minutiae are known as features and are used in criminal court cases to refute or support identification around the world such as the 1999 US versus Byron Mitchell, 2000 US versus Hilerdieu Alteme, and the Commonwealth versus Owen McCants (2914).

Biometric data is growing at an accelerating rate and is creating an unimpeded data avalanche. Smart devices with cameras, microphones and esoteric sensors combined with maturing networking capabilities is making applications that utilise biometrics viable on a previously unimagined scale. Recent hand-held device advances such as the Apple iPhone 5 S allows fingerprint data to be gathered and potentially incorporated into mobile apps. Social media sites such as Facebook are beginning to use facial recognition to automatically tag humans in images. Further, large-scale biometric systems can be exemplified by India's National ID programme, Aadhaar, with goals to enrol multi-modal biometrics from 1.2 billion people from the country's general population [4]. Interest in using biometrics is exploding in areas such as criminal

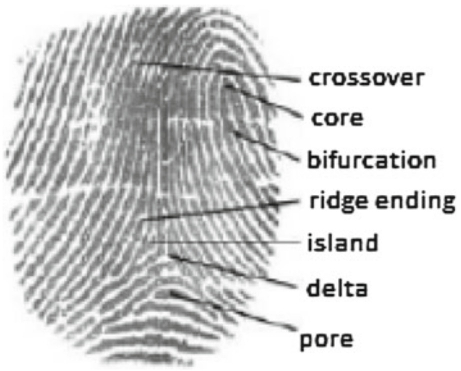


Fig. 8.1 Example fingerprint images and minutia

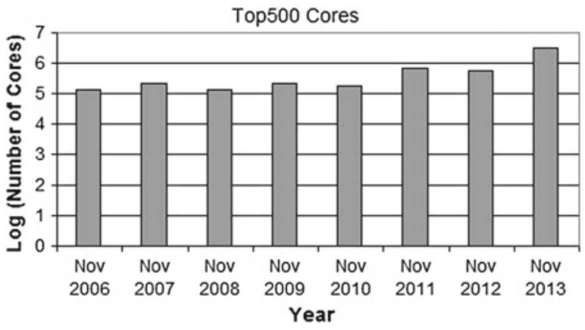


Fig. 8.2 The growth of cores on top supercomputers since 2006

investigation, customs control, health care, banking, insurance, industrial and residential access control. These applications typically require candidate gallery matches within a few seconds from the submission of a biometric probe identification job.

The alarming growth of biometrics data has implications for existing techniques. On the algorithm level, the process of feature extraction or templatisation and the process of performing a match can be difficult to reach high levels of specificity, sensitivity or precision. Identification is more challenging than verification when applied against a large database of biometrics or gallery where a probe could contain several biometric samples [1]. To increase performance, many techniques have been used in biometric identification research that focus searches on the area of the data that most likely contains the identity in question [1, 2, 5–9]. As such, the penetration of a large-scale biometric database is reduced. A few of these techniques include binning and hashing functions [1]. Large-scale biometric identification systems can benefit from modern optimisation, classification and parallel computation techniques to reduce cost and increase accuracy.

The field of biometrics is a natural fit for parallel computing. Parallel computing allows a single problem to be broken down into many smaller pieces that can be addressed simultaneously. In the past, this kind of technology was only available in supercomputers, but recent advances in processor design and lower costs have made it possible for firms of modest size to have access to hundreds or even thousands of processor cores. Figure 8.2 depicts the number of cores on the fastest supercomputer since 2006.[1] New tools and methods have emerged that enable rapid and cost-effective deployment of massively parallel data intensive applications. This chapter focuses on techniques and designs to address large-scale biometric identification.

While the process of identification may not necessarily be implementable as a parallel operation, the use of search space reduction techniques together with relevant parallel computing techniques enables the same amount of hardware to perform more simultaneous searches.

This chapter discusses recent and novel developments by the authors in the approaches taken to enable large-scale biometric identification. The authors present

[1] www.Top500.org.

an overview of different techniques to perform the tasks of search space reduction, feature selection and parallel processing of biometrics data. Section 8.2 discusses how support vector machines and hyperspace transformations can be used to effectively search extremely large fingerprint databases to identify individuals. Section 8.3 discusses the use of evolutionary computing to perform more efficient facial feature selection for identification purposes. Section 8.4 discusses new cloud and high-performance designs for biometric systems.

8.2 Large-Scale Fingerprint Identification Using SVM

This section shows how large-scale fingerprint identification can be performed using support vector machines. A SVM is used to reduce the search space needed to identify the person.

8.2.1 Related Work

The quantity and magnitude of fingerprint databases are increasing in size. Generally, the community is converging towards a definition of large-scale databases. A readily accepted source defines a very large biometric database with 1 million records [2].

Several country citizen repositories pass the extremely large database category as shown in Table 8.1. All of the repositories below are part of the very large database classification. The US Visit programme, operated by the US Department of Homeland Security (DHS), contains over 100 million people's fingerprints. The Federal Bureau of Investigation (FBI) maintains over 54 million sets of 10 fingerprints each. The Unique Identification Authority of India maintains over 5 Billion fingerprints.[2] The database sizes of a number of countries are listed in Table 8.1. As database sizes increase, the number of false positives increases quadratically with the size of the database [2].

Technological advances have improved all layers within the large scale biometric identification problem. Automated fingerprint identification can be decomposed into fingerprint capture, feature extraction, file partitioning or binning, pre-screen matcher, secondary matcher and decision logic [3]. Numerous feature extraction, pattern recognition and template matching algorithms have been developed and analysed with respect to speed and a receive operator curve (ROC) [2, 10, 11]. File partitioning, indexing or binning algorithms reduce the search space of a biometric database. Baughman et al. investigated the viability of using a novel hyperspace structure with a plurality of kernel functions within a Support Vector Machine (SVM) as a method to perform search space reduction [6].

[2] https://portal.uidai.gov.in/uidwebportal/dashboard.do.

Table 8.1 International biometric repositories [2]

Country	Programme	Database size	Biometric
India	National ID	5 B	Fingerprint
UAE	National ID	103 M	
Thailand	National ID	15–40 M	
Peru	Voter registration	13 M	
Hong Kong	National ID	7 M	

A SVM kernel can utilise fingerprint minutiae and ridge flow information. Fingerprint minutiae points are ridge endings, bifurcation or n-furcation of the fingerprint. Within a Cartesian grid, the (x, y) position give the location of the point while the θ point defines the angle of incidence to the ridge flow [12, 13]. A delta point is the area on a fingerprint where the ridge flow diverges while the core is the point where ridge flow is symmetric [12, 13]. Fingerprint ridge flows can be classified into six main classes: scar, whorl, left loop, right loop, arch and tinted arch, with many derivations [12, 13].

Yao et al. used SVMs with the NIST-4 special database [9]. The fingerprints were a priori classified into whorl, right loop, left loop, arch and tented arch. A bank of SVMs was trained on the classes for one versus all, pair wise and error-correction scheme experimentation [9]. Each of the classifiers found the support vector between two fingerprint classifications. With the error-correction scheme, the algorithm achieved 89.3 % classification accuracy with 1.8 % rejection rate during feature extraction [9].

In 2008, Jin-Hyuk et al. extended Yao's SVM work and used the NIST-4 special database. The SVM was used to vector fingerprints according to an image's ridge flow [14]. The one versus all scheme was dynamically ordered with naive Bayes classifiers to break ties that occur with multi-class classification systems. The algorithms produced 90.8 % classification accuracy [14]. This paper extends both works by utilising a plurality of kernels within a hyperspace construct.

As an alternative to SVMs, a few works have addressed fingerprint indexing. Within Germain's work, an algorithm of transformation parameter clustering attempts to build fingerprint database index maps [1]. Triangular minutiae constellations are created from

$$C(n, 3) = \frac{n!}{3!(n-3)!} \tag{8.1}$$

minutia points, where n is the total number minutiae points within a fingerprint. A bound is established so that not every triangular shape is computed. The full index or key consists of nine components: length of each side, ridge counts between the pair and angles between the pair [1]. Each index is placed into a multi-map or container. If the key is already within with a container, the entry is added. Through a search, accumulation of evidence for a potential match is generated based upon the members that are found within the probe key set. On average, the algorithm achieved 7.3 μsec per print with a 1/10 False Negative Rate (FNR) and 1/1,000,000 False Positive Rate (FPR) [1]. This paper presents an alternative dynamic and modular SVM for search space reduction.

8.2.1.1 Support Vector Machines

A Support Vector Machine (SVM) [15, 16] is a supervised learning pattern recognition algorithm that is commonly implemented within the pattern recognition field. An SVM projects feature vectors into a linear or non-linear state space using kernel function(s) and attempts to maximise the margin between classes [16]. The projection of low-dimensional feature vectors into a higher dimensional hyperspace structure helps to provide sparse separable clusters of data, which in turn makes classification easier. SVMs perform well on very high-dimensional data (1,000s of columns). Biometric data such as fingerprints that have been transformed into feature vectors are good candidates for SVM classification.

8.2.2 Fingerprint Identification Using SVM

The aim of using an SVM for fingerprint identification is to reduce the search space by returning a smaller result set to search for a matching fingerprint. The process starts by converting fingerprint position and ridge flow pattern classifications into feature vectors. The feature vectors in turn serve as input to one or more kernel functions. When multiple kernel functions and SVMs are defined, a hyperspace structure encapsulates the space. A composite kernel of fingerprint position and a ridge pattern classification heuristics maps fingerprint data into a hyperspace structure. A hyperspace structure provides a construct that groups and creates relationships between kernels and SVMs. The result is an infinite projection space for kernel functions. As samples are projected into a hyperspace construct, fingerprint identification velocity improves while system performance will increase or remain constant.

8.2.2.1 Kernel Definition

A parametric function projects low-dimensional data onto high-dimensional spaces vector data into similar sparse groups [16]. A function,

$$\mathbf{y} = f(\mathbf{x}, \mathbf{w}) \tag{8.2}$$

is a mapping of input vector $\mathbf{x}$ to output vector $\mathbf{y}$ with $\mathbf{w}$ weights. A training phase selects $\mathbf{w}$ that minimises classification error $E(.)$ [16]. From Eq. 8.3, the error measure is equal to a distance measure $l(.)$ between a prediction $f(\mathbf{x}, \mathbf{w})$ and target value $\mathbf{y_t}$.

$$E(f(\mathbf{x}, \mathbf{w})) = \sum_{t=1}^{m} l(f(\mathbf{x}, \mathbf{w}), \mathbf{y_t}) \tag{8.3}$$

The kernel function selected and trained within the paper includes multiple independent kernels [16]

$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \tag{8.4}$$

where k is a kernel function defined by the summation of two separate kernel functions k_1 and k_2, $\mathbf{x}$ is the feature vector, and $\mathbf{x}'$ is the projection weights vector. The first kernel maps fingerprints into finger digit positions. The right thumb is position one while the left thumb is position ten, i.e.

$$k_1(\mathbf{x}_i, \mathbf{x}') = p(\mathbf{x}_i') = r, r \in [1, 10] \tag{8.5}$$

where the function $p(\mathbf{x}_i')$ projects a feature vector $\mathbf{x}_i$ into a range r that maps to fingerprint digits 1 through 10. A kernel k_2 defines a bootstrapped c-means clustering function from the $\mathbf{x}'$ weights or initial cluster epicentres. Using the minimum squared difference between normalised features $q(\mathbf{x}_j)$ and epicentre $\mathbf{x}_k'$, the cluster membership is selected as follows:

$$k_2(\mathbf{x}_j, \mathbf{x}') = c(\mathbf{x}_j, \mathbf{x}') = \min \| \sum_{k=0}^{N} (q(\mathbf{x}_j) - \mathbf{x}_k')^2 \| \tag{8.6}$$

The selected minimum cluster(s) will become the group value for $q(\mathbf{x}_j)$. The feature vector of a fingerprint feature vector $q(\mathbf{x}_j)$ is transformed into normalised values as

$$q(\mathbf{x}_j) = s\mathbf{x}_j, s \in \mathbb{R} \tag{8.7}$$

where s is any real value that normalises feature values from potentially different feature extractors. A function M defines the higher dimensional mapping

$$k(\mathbf{x}, \mathbf{x}') = M(r + c) \tag{8.8}$$

where $r = k_1(\mathbf{x}_i, \mathbf{x}')$ and $c = k_2(\mathbf{x}_j, \mathbf{x}')$. Working towards a linear dual representation,

$$k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}') = (r_i, c_i) \tag{8.9}$$

leads to dimensional mapping [16], which is used in this chapter.

8.2.2.2 Feature Classification

A bootstrapped C-Means clustering algorithm, as defined in algorithm 1, separates fingerprint scar, whorl, left loop, right loop, arch and tinted arch classes. Initially, each data element was itself a cluster [17]. Sequentially, the data elements were evaluated for cluster membership by the Euclidean distance between its feature vector, a single classification point, and each cluster's epicentre. If the sample's smallest distance from a cluster is equal to a threshold of 0 metric units, the data sample was placed into the cluster. The threshold for clustering was 0 because the classification of fingerprint patterns was integers in the range $[1, 6]$ [13]. If a cluster does not exist for a fingerprint classification, a new cluster is formed. When the probability is lower than a majority vote of 50 %, the fingerprint is boosted for all patterns [18]. The

Algorithm 1 Bootstrapped C-means clustering pseudocode

```
A priori define $(\tau, t)$ where $\tau$ is an empirically determined threshold and $t$ is the time requirement for biometric identification
Step 1: For all fingerprints in the set of all fingerprints, $\forall f_i \in F$, all fingerprints $\sum_{k=0}^{N}(f_i \in C_j)^2$ are assigned to cluster $C_j$ where $0 \le j \le |\mathbf{C}|$
Step 1a: For each fingerprint $\forall f_i \in F$ and $i - |F|$
Step 1b: Determine the minimum Euclidean distance, $c = \min \| \sum_{k=0}^{N}(q(\mathbf{x}_j) - \mathbf{x}'_k)^2 \|$, to a cluster where $\mathbf{x}_j = u(f_i)$ and $u(.)$ is the feature extraction algorithm for fingerprints.
Step 2: If $c > \tau$ create a new cluster $\mathbf{x}'_{k+1} = u(f_i)$.
Step 3: If $c \le \tau$ recalculate an existing cluster $\mathbf{x}_k = \sum_{i=0}^{N+1} \frac{\mathbf{x}_i}{N+1} + \frac{N(\mathbf{x}'_i)}{N+1}$
Step 4: Check to determine if the cluster epicentres have changed, $\mathbf{b} = \sum_{k=0}^{N} \mathbf{x}'_{kt} == \mathbf{x}'_{kt+1}$.
Step 4a: If $\mathbf{b}\&\mathbf{c} == \sum_{i-0}^{N} 2^i$ where $\mathbf{c} = 1_1 \cup \cdots \cup 1_N$ then the c-means algorithm stops.
Step 4b: If $t_c > t$, where $T_c$ is the cumulative algorithm time and $t$ is the a priori time then the c-means algorithm stops.
Step 5: Continue step 1 if both step 4a and step 4b are not true.
Step 6: Return the cluster space, $\forall f_i \in \mathbf{x}'$
```

clustering algorithm continued until none of the epicentres moved or began thrashing within the cluster space.

8.2.2.3 Hyperspace Structure

Below, the following automaton defines the hyperspace construct:

Hyperspace construct: H = Hyperspace, Λ_i = Sub-Universe, Ω_w = World, $\phi_{i,j}$ = Dimension, $\rho_{i,j}$ = Policy, $\delta_{i,j,k}$ = Operator, A_l = Cluster, $\alpha_{l,i,j}$ = Bin Member, where i = sub-universe number, j = dimension number, k = operator number, l = cluster number, m = world number, $A_l = \forall\alpha_{l,i,j}$

A cluster is defined as the set of all cluster members. Recall that a cluster is related to SVM kernel $k_2(\mathbf{x}_j, \mathbf{x}')$ from Eq. 8.6. Each cluster member is vectored into a n-dimensional space. From kernel $k_1(\mathbf{x}_i, \mathbf{x}')$ from Eq. 8.5, each dimension represents a fingerprint position. A sub-universe is defined as a set of dimensions or set of fingerprint positions:

$$\Lambda_i = \{\forall\phi_{i,j}\} \tag{8.10}$$

For each dimension there exists an associated policy and a set of operators. The policy provides an association between operators or heuristics and a dimension.

$$\phi_{i,j} = \exists\rho_{i,j} \cup \{\forall\delta_{i,j,k}\} \tag{8.11}$$

Within the context of fingerprint identification, the heuristic determines which fingerprint feature vector belongs to each fingerprint classification such as whirl, tented arch, left loop, etc. and to which cluster based on Cartesian distance. The world automaton, Ω_w, specifies the set of all data elements or fingerprints which belong to all clusters within a sub-universe and the sub-universe itself:

$$\Omega_w = \{\{\forall \alpha_{l,i,j}\}\varepsilon\{\Delta_i\}\} \cup \{\Delta_i\} \tag{8.12}$$

The world automaton could span multiple SVMs if required. Within this chapter, we only use 1 SVM with 2 kernels. Finally, the hyperspace automaton defines the universe of a problem domain and contains the SVM:

$$H = \{\forall \Omega_w\} \tag{8.13}$$

8.2.3 Experimental Procedure

All experiments that were performed used the SVM technique as described in Section 8.2.2. The dataset comprised of a 6,278 sub-sample of 24,000 records in the National Institute for Standards and Technology (NIST) special database 14. All of the fingerprint images were encoded with the Wavelet Scalar Quantisation (WSQ) specification [19]. The file name contained the fingerprint position label. Several open source algorithms from NIST were implemented to produce the features required for the hyperspace structure [13]. MINDTCT, BOZORTH3 and PCASYS were respectfully used as templatisers, matchers and pattern classifiers [13]. The implementation of PCASYS as single layer Probabilistic Neural Network (PNN) was chosen for fingerprint classification to vector a fingerprint into the appropriate clustering algorithm within the Hyperspace structure. The PCASYS PNN was tested on 2,700 fingerprint images and trained on 24,000 fingerprint images from the NIST special database 14. Each fingerprint image had a header that labels the image's classification. The total percentage of misclassification was 7.07 % [13].

MINDTCT produces a fingerprint template with an (x, y) coordinate and an angle of incidence with respect to the ridge [13]. BOZORTH3 accepts as input two MINDTCT templates and produces a correlation score. Several metrics defined below were used to measure the performance of the fingerprint matching system before and after the implementation of the hyperspace structure. The confusion matrix shown in Fig. 8.3. depicts the relationship of true positives, false positives, false negatives and true negatives relative to actual and predicted class. Further, confusion matrices are the basis of Receiver Operator Characteristics (ROC) curves that can be used to compare classification systems [11].

To measure the velocity of the true positives (*TPR*), the total number of true positives *TP* is divided by all predicted positive classes P:

$$TPR = TP/P \tag{8.14}$$

Similarly, the false positive rate is calculated as follows:

$$FPR = FP/N \tag{8.15}$$

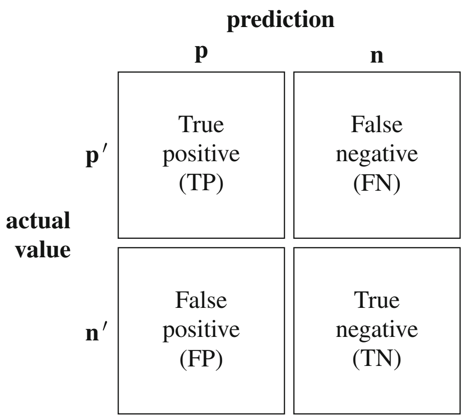


Fig. 8.3 Typical confusion matrix

The higher the TPR, the better classification velocity the system achieves. The accuracy metric measures the percentage that is correctly classified from both match and non-match sets:

$$\text{accuracy} = \frac{TP + TN}{\sum \text{match} + \sum \text{non-match}} \tag{8.16}$$

Equally important is the precision of the classification system, which calculates the percentage of true matches over all predicted and actual positives [11]:

$$\text{precision} = \frac{TP}{FP + TP} \tag{8.17}$$

Finally, the penetration rate, p_r, and penetration average, p_a, measure how well the SVM system reduces the search space:

$$p_r = \frac{\sum_k \|c(\text{probe}, \text{gallery}_k)\|}{\sum_i \|c(\text{probe}, \text{gallery}_i)\|} \tag{8.18}$$

$$p_a = \frac{1}{N} \sum_i^N \|c(\text{probe}, \text{gallery}_i)\| \tag{8.19}$$

where $c(\text{probe}, \text{gallery}_i)$ is the total number of fingerprint comparisons between the probe and gallery after search space reduction, $c(\text{probe}, \text{gallery}_k)$ is the count of fingerprint comparisons before search space reduction, and N is the total number of probes.

8.2.4 Experimental Results

The results of using support vector machines and hyperspace transformations to effectively search extremely large fingerprint databases to identify individuals are outlined below.

8.2.4.1 Bootstrapped C-Means Results

The accumulation of all 10 dimensions defined by fingerprint position had an average of 5.6 clusters with all six fingerprint patterns: arch, left loop, right loop, scar, tented arch, and whorl. Each kernel defined by fingerprint position and cluster space had an average of 122.1 fingerprint images. Of the kernel members, 101 fingerprints were not confidently classified. As a result, the fingerprints were inserted into each cluster within the fingerprint position dimension. In summation, 505 fingerprint samples were clones of poorly classified fingerprints by PCASYS. The additive property of kernel functions results in several C-Means cluster spaces that overlap to reduce false negatives.

8.2.4.2 SVM Results

As shown in Table 8.2, the SVM system reduced search space where the $N \times N$ search type had a total of 6,278 images in the gallery or search space. On the full identification search, each fingerprint searched 6,278 other samples. With the SVM, on average, each probe was correlated with 306.7 gallery samples. Only 4.89 % of the full gallery was searched with SVM searches.

The total gallery search contained 20 times more true non-match pairs than the SVM search despite the sample boosting by the c-means clustering algorithm. Both search types contained five match scores or false positives that were below the score threshold of 100. For data visualisation, if a comparison was missing from the SVM run, the pair was added as a non-match. Figures 8.4 and 8.5 depict that the SVM and $N \times N$ score distributions have the same number of samples and the score distribution

Table 8.2 Penetration rate comparison

	Avg. penetration (p_a)	Penetration ratio (p_r) (%)	Penetration mode
Full $N \times N$	6,278	100	6,278
SVM system	306.7	4.89	530

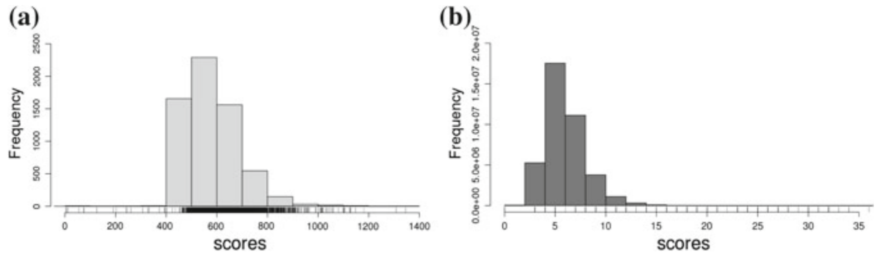


Fig. 8.4 Full $N \times N$ search match (*left*) and non-match (*right*) score distribution [6]

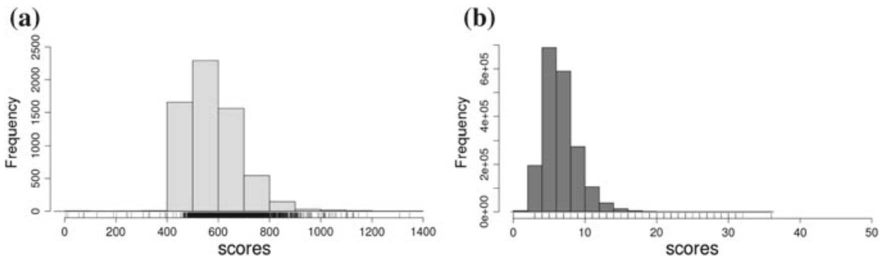


Fig. 8.5 SVM match (*left*) and non-match (*right*) score distributions [6]

Table 8.3 Full $N \times N$ run description [6]

Match set size	6,278
Non-match set size	39,400,726
Match scores < 100	4, 12, 55, 74 and 75

Table 8.4 SVM run description [6]

Match set size	6,278
Non-match set size	1,919,468
Match scores < 100	4, 12, 55, 74, and 75

is identical. As a result, the SVM does not change the normal score distribution of the $N \times N$ comparison.

The SVM and $N \times N$ runs maintained constant fingerprint data and implemented the same fingerprint matcher. Before the implementation of the SVM, the fingerprint matcher performed extremely well with a 0.99 TPR at 0.07 FPR. The high accuracy of 0.92 is offset with a low precision of 0.002 due to a high number of false positives. The precision begins to improve as the score threshold reduces the number of false positives. The SVM statistics empirically proves that the best TPR at a threshold of 10 has a lower FPR and higher precision than the same corresponding TPR in the $N \times N$ run. The tradeoffs between TPR, FPR, accuracy, precision and threshold are better with the SVM than the full $N \times N$ (Tables 8.3 and 8.4).

8.3 Facial Feature Selection Using Evolutionary Algorithms

To perform identification using facial features, all matching face profiles need to be retrieved as fast and reliably as possible from a database of facial features. Some subsets of features might be more important than others in retrieving the relevant individual's identity. Correctly identifying this subset of features has major implications on system performance. This section shows how an EA facial feature selector chooses a set of features from prior contextual and meta-face features that reduces the search space for large data sets [5].

8.3.1 Related Work

Modern biometric systems require increasingly sophisticated solutions to address the areas of data volume, velocity and variety through data dimension reduction and simplification. Often the resulting search space of possible data features, ways to create clusters and the sheer amount of data requires optimisation to find the best combination of operators. EA offers a powerful, resilient and flexible technique to address this challenge.

8.3.1.1 Evolutionary Algorithms

Evolutionary Algorithms (EA) and Genetic Algorithms (GA) describe a system that uses computational modelling of evolution, including processes such as natural selection, survival of the fittest and reproduction [20–24]. EA captures ideas from biological evolutionary theory by 'evolving' a solution instead of using algebraic derivation. Many types of EAs exist, including genetic algorithms, genetic programming, differential evolution, etc. GAs model genetic evolution by mimicking aspects such as survival of the fittest (selection) and reproduction (crossover and mutation) [21].

EAs can be used to find approximate solutions to NP hard problems, where several criteria for a good solution are known, but analytical techniques are not able to directly derive a solution. This is possible because an EA does not require the "genetics" of a problem to be known: the rules that govern whether solutions are valid or invalid, perform well or badly, or the exact transformation between states do not need to be known. Using stochastic operators, an EA is able to distinguish good genetic combinations from bad ones, and is able to modify and steadily improve the quality of solutions. An EA can also efficiently avoid getting stuck in local minima by adequately exploring the search space.

An EA is a "guided random walk". A set of initial random solutions (the parent generation) is modified by random operators (reproduction and mutation). The resulting solution set (the offspring generation) is evaluated by a fitness function. The offspring with the highest fitness become the next parent generation (selection), upon which the procedure is repeated. In a tuned EA, the population will converge towards solutions which are close to the optima of the fitness function [20, 22, 23, 25–27].

Learning the face space of a face database maintains or reduces the dimensionality of data. The reduction of data complexity reduces the amount of computational complexity. The evolutionary pursuit seeks to learn an optimal face space for the purpose of pattern classification and data compression [28]. Evolution is driven by a fitness function. An example fitness function combines performance accuracy, $\xi_a(F)$, with the predicted risk, $\xi_b(F)$, to evaluate a face space [28]. Chengjun Liu et al. examined the application of genetic algorithms to face recognition. The fitness function includes $\xi_a(F)$, which defines facial recognition accuracy and $\lambda\xi_b(F)$, which defines class scatter [29]:

$$\xi(F) = \xi_a(F) + \lambda\xi_b(F) \tag{8.20}$$

8.3.2 Facial Feature Selection Using EA

In a static system, the implementation of Principle Component Analysis (PCA) is straightforward to reduce the search space and only focus on the most important predictors. The challenge faced by identification systems is that, firstly, each data set is unique and, secondly, that many data sets are dynamic: new identities and facial features are constantly added or removed. Different or new information has the potential to completely change the importance of features. This in turn means the features selected for tuning the classifier will probably not be the most important features as more data is ingested. The implication is that the accuracy of the system will suffer over time. The optimal set of important features needs to be selected regularly as guided by the changing data space.

An Evolutionary algorithm approach was applied to facial feature selection to search over the space of all meta-face features to reduce the search space [5]. The result is a subset of those features that provides the highest identification accuracy together. The paper also presents the design, parameter selection and experimental results of an evolutionary facial feature selector algorithm. EA improves upon random search by exploiting structural knowledge of the search space. This structural knowledge is imposed by the features of the elements of the search space.

The creation of a search algorithm on the space of facial features has to address two problems: first, according to what criteria shall the search space be structured; and second, what is the best actual algorithm which should be chosen given that search space structure. The problem of imposing an appropriate structure on the search space is discussed in [5], i.e. which facial features need to be included in the search space

Algorithm 2 Evolutionary facial feature selection

```
A priori define $GA = \{p_c, p_m, G, l, \mu_\xi, xi_g, s_\xi, \sigma, t\}$ where $p_c$ is the probability of crossover, $p_m$ is the probability of mutation, $G$ is the number of generations, $l$ is the length of a chromosome, $\mu_\xi$ is the fitness function, $xi_g$ is the schema or set of chromosomes for generation $g$, $s_\xi$ is the selection method such as tournament selection, $t$ is the maximum clustering run time to avoid thrashing and $\sigma$ is an empirically determined threshold for k-means clustering [3].
Step 1: Initialise the genetic algorithm with $GA$.
Step 2: For all facial images, $\forall I_i \in I$, create chromosome feature selectors where $\xi_{gi} \in \xi_g$ and each chromosome has length $l$.
Step 3: Decode each $\xi_{gi}$ into feature vector $\boldsymbol{\xi_{gi}}$ with function $\boldsymbol{\xi_{gi}} = f(\xi_{gi})$.
Step 4: Run an agglomerative k-means clustering algorithm for all $\xi_{gi} \in \xi_g$ where $K = |\xi_g|$.
Step 5: Merge clusters together that are $\text{dist}(k)i, K_j) \leq \sigma$ to produce cluster space $cp_s$ where $s$ is the cluster step or iteration.
Step 5a: End clustering if $cp_{s-1} == cp_s$ or the total runtime, $t_t \geq t$ and return $cp_{\xi_i}$.
Step 6: For all $\forall CP_{\xi_i} \in CP$, apply a fitness function, $\mu_\xi$, to produce a ranked clusterspace set, $CP_r$.
Step 7: Select the next set of chromosomes with $\xi'_{g+1} = s_\xi(\xi_g)$.
Step 8: For each ranked $\forall \xi'_{g+1,i} \in \xi'_{g+1}$, apply $p_m(\xi'_{g+1,i})$ and $p_c(\xi'_{g+1,i}, \xi'_{g+1,j})$ to produce the next generation's set of chromosomes $\xi_{g+1}$
Step 9: Continue the genetic algorithm for $G$ generations or until $\xi_{g-1} == \xi_g$.
Step 10: Return the optimal chromosome $\xi_{g,i}$.
```

to maximise the system's classification abilities. The results in [5] show that the evolutionary algorithm (EA) approach is a convenient method to tackle such problems. First, advanced mathematical knowledge, the exact importance of each feature, and the efficiency of a particular search algorithm do not need to be known upfront to make use of EA to optimise the system. Secondly, implementation can be rapid, even when starting from scratch, and results can be generated fast. EA can easily be tailored to this algorithm selection problem to fine-tune performance. One drawback of EAs is their computational complexity, and a big part of good EA design goes into computational efficiency. The pseudo code for algorithm 2 follows Holland's notation and describes the Evolutionary Facial Feature Selection work [24, 26].

8.3.3 Experimental Procedure

An EA approach was used with agglomerative k-means cluster spaces as input parameters into a Linear Discriminant Analysis (LDA) evaluation function to select facial features from the Carnegie Mellon University Pose, Illumination and Expression database of human faces (PIE) [5, 30]. Example meta-features include blinking, has a hat, has glasses, pose, smiling, chubby cheeks, hair color, etc. (Fig. 8.6).

A basic principle of a search space structure is that similar items are "close together", meaning that if a search algorithm encounters one item, then items similar to that item are readily available. To this end, the approach chosen in [5]

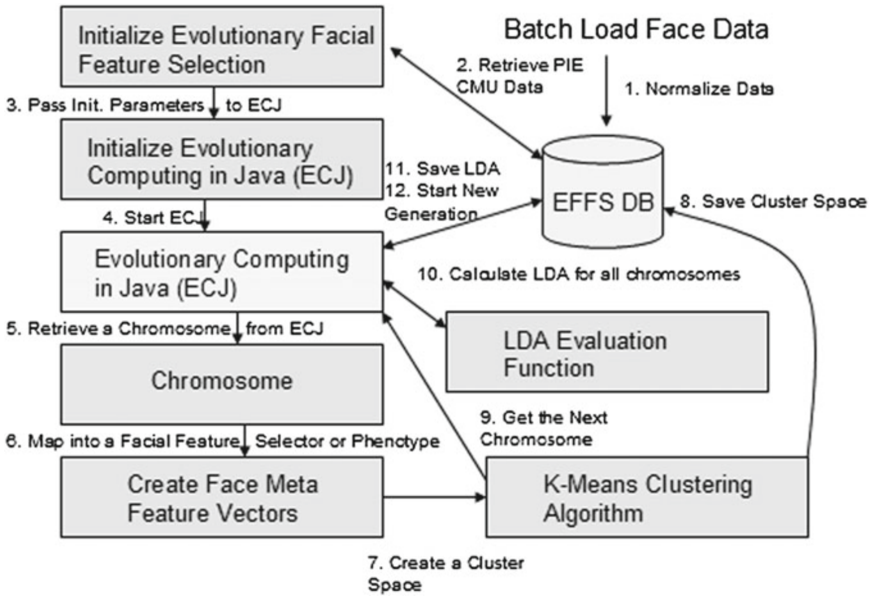

Initialize Evolutionary Facial Feature Selection
Batch Load Face Data
2. Retrieve PIE CMU Data
1. Normalize Data
3. Pass Init. Parameters to ECJ
Initialize Evolutionary Computing in Java (ECJ)
11. Save LDA
12. Start New Generation
EFFS DB
8. Save Cluster Space
4. Start ECJ
Evolutionary Computing in Java (ECJ)
10. Calculate LDA for all chromosomes
5. Retrieve a Chromosome from ECJ
LDA Evaluation Function
Chromosome
9. Get the Next Chromosome
6. Map into a Facial Feature Selector or Phenotype
Create Face Meta Feature Vectors
K-Means Clustering Algorithm
7. Create a Cluster Space

Fig. 8.6 Functional architecture of the evolutionary facial feature selection method [5]

combines agglomerative k-means clustering and LDA. Agglomerative k-means clustering establishes the clusters within which faces are similar to each other, and LDA measures the quality of the cluster space by relating the diameter of the clusters with their distance from each other. The LDA score is the between distance divided by the in-between distance, so the larger the score, the better the cluster space. A cluster space is considered better if the average diameter of clusters is small and the average in-between distance between clusters is large. Different cluster spaces are obtained by different projections of the feature space, essentially eliminating subsets of features.

An evolutionary algorithm approach was used to search for a good-performing cluster space. The EA operated on the population of binary vectors that indicate whether a feature was considered or not. A value of 0 means the feature was omitted while 1 indicates the feature is present. Each binary feature vector gives rise to an agglomerative k-means cluster space for which the LDA score was calculated. The LDA analytic was chosen since the goal of the clustering was to have clusters tightly centred around an epicentre with each cluster maximally spread apart [12]. In addition, LDA is a quick and simple analytic to calculate within the framework of a complex genetic algorithm framework. For each generation, the performing binary feature vectors were selected for crossover and mutation to form the next generation of binary feature vectors. The overall number of features for these experiments was 12, and the population size was 10. In each generation the best two binary feature vectors were subjected to either one point or two point crossover. The mutation rate was either 1 or 100 %, so there were 4 overall experiment configurations.

8.3.4 Experimental Results

Two sets of experiments were run on the full Pose, Illumination and Expression (PIE) database [30]. Each experiment consisted of two batches. Within each batch, one genetic operator was selected as an independent variable, while all other genetic algorithm parameters were held constant. Between batches, the affect of the chosen genetic algorithm parameters were examined with respect to LDA and the average number clusters for each generation. In total, four different sets of parameters were chosen for the evolutionary facial feature selection.

The experiment depicted in Fig. 8.7 was setup as:

- $|G| = 25$, where G is the generation.
- Independent Variable: mutate(x_j, $P(1)$) or mutate(x_j, $P(0.01)$) where x_j is a chromosome and $P(.)$ is the probability of a mutation.
- Constant: onePoint(x_j, x_i, $P(1)$) where x_j and x_i are two chromosomes from tournament selection and $P(.)$ is the probability of a one point crossover.
- Dependent Variables: LDA.

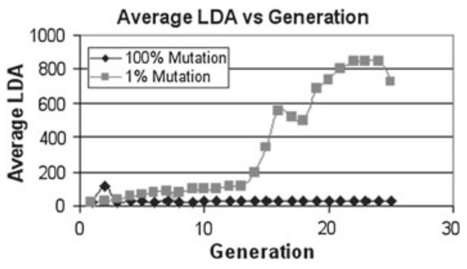


Fig. 8.7 Mutation and LDA results (25 generations) [5]

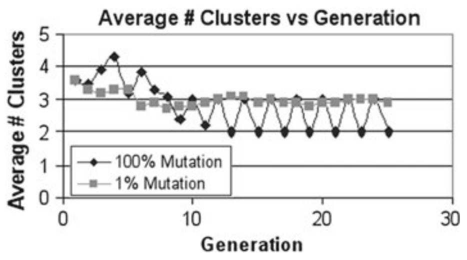


Fig. 8.8 Mutation and average number of clusters (25 generations) [5]

The experiment depicted in Fig. 8.8 was setup as:

- $|G| = 25$, where G is the generation.
- Independent Variable: mutate$(x_j, P(1))$ or mutate$(x_j, P(0.01))$.
- Constant: onePoint$(x_j, x_i, P(1))$.
- Dependent Variables: Average number of clusters.

The experiment depicted in Fig. 8.9 was setup as:

- $|G| = 25$, where G is the generation.
- Independent Variable: onePoint$(x_j, x_i, P(1))$ or twoPoint$(x_j, x_i, P(1))$
- Constant: mutate$(x_j, P(0.01))$
- Dependent Variables: LDA

The experiment depicted in Fig. 8.10 was setup as:

- $|G| = 25$, where G is the generation.
- Independent Variable: onePoint$(x_j, x_i, P(1))$ or twoPoint$(x_j, x_i, P(1))$
- Constant: mutate$(x_j, P(0.01))$
- Dependent Variables: Average number of clusters

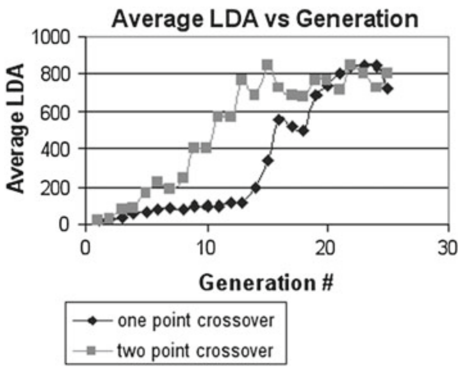


Fig. 8.9 Crossover type and LDA results (25 generations) [5]

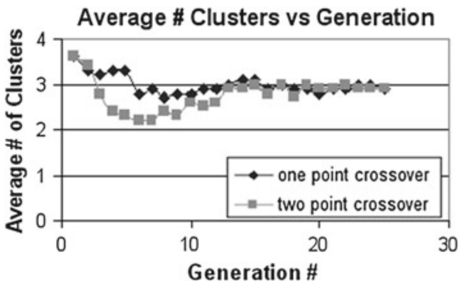


Fig. 8.10 Crossover type and average number of clusters (25 generations) [5]

With one point crossover, 0.01 mutation and 25 generations, the best converged solution included the chromosome with blinking, smiling and pose. By generation 14, the best fit chromosome was evaluated to 845.5 and appeared once. The entire population consisted of the best fit chromosome by generation 22. Two-point crossover yielded an identical best chromosome as one point crossover. By the fifth generation, the best fit chromosome appeared once. By generation 15, the best fit chromosome was the entire population. Two-point crossover oscillated between the best fit chromosome and a second member that was three times less fit.

Two-point crossover achieved the best fit chromosome faster than one point crossover. However, after achievement, the two point crossover operator diverged more than the single point crossover.

With a mutation of 1, one point crossover and 25 generations, the best chromosome oscillated between one that only contained blinking with a score of 25.42 and another that contained all traits except light with a score of 33.53. Neither score was close to a previous best of 845.5. When the mutation rate was changed to 0.01, the best converged solution contained blinking, smiling and pose with a score of 845.5. From Fig. 8.7, the LDA evaluation results with a mutation rate of 0.01 was over a magnitude better than with 1.

The LDA evaluation score was 25.42 as contrasted to a previous best of 845.5. The first ranked chromosome was present 5 times. The second ranking chromosome contained blinking and neutral weights. A mutation rate of 0.01 with 10 generations produced a chromosome of blinking and smiling of rank 1 twice. The second ranking chromosome was present twice and contained weights for neutral, pose, flash and needs glasses. Both mutation rates of 0.01 and 1 contained chromosome 101000000 within the top two.

This indicates that a lower mutation rate is better than an extremely high rate with achieving the best chromosome. In addition, Fig. 8.8 depicts the oscillation characteristic of mutation 1. Local optimum solutions were not annealed. The overall result was that the experiment with 1 % mutation rate and one point crossover produced the best binary vector by generation 14. By generation 22 the whole population had converged on this same solution. Two point crossover and 1 % mutation rate yielded the best vector to appear in generation 5, with the entire population converging to it by generation 15. A mutation rate of 100 % resulted in no selection whatsoever, although the best vector did emerge (Fig. 8.9 and Fig. 8.10).

8.4 Parallel Computing and Cloud Architecture for Biometrics

This section highlights some recent developments in multi-core programming and cloud computing, and how these technologies are enabling applications of biometrics on a scale to leverage Big Data. Specific architectural designs that foster greatly improved biometric application performance are also discussed.

8.4.1 Related Work

The first biometric system was built in 1967 by Cornell Aeronautical and North American Aviation, which was a precursor to the first US Federal Bureau of Investigation system that was deployed in 1972 [31]. The Finder system supported 10 matches per second and enrolled 1 fingerprint per second [31]. Starting in the 1990s, the National Institute of Standards and Technology (NIST) has pushed the progression of biometric algorithm development to improve overall biometric accuracy with programmes such as the Facial Recognition Vendor Tests (FRVT), Iris Competition and Evaluation (ICE), Fingerprint Vendor Technology Evaluation (FpVTE) and the Biometric Grand Challenge (BGC) [32–34].

The trade space on the Receiver Operator Characteristic Curve (ROC) and the Detection Error Trade (DET) curve has improved over several orders of magnitude

[32–35]. As a residual, time limitations were set on each of the evaluation experiments. For example:

- Within FRVT 2002, a high computational intensity test of 15 billion facial comparisons were evaluated with a limit of 264 h and minimum of 15,555 comparisons per second was designed to encourage extremely fast algorithms [32]. The medium computational intensive test was designed to evaluate slower and perhaps more accurate algorithms with a minimum of 66 matches per second [32].
- Within FpVTE 2003, the large-scale test set of 1.044 billion comparisons had a limit of 21 days or a minimum of 575.4 matches per second [34].
- The ICE 2006 competition limited algorithms that could run on a single Intel Pentium 4 3.6 GHz 660 processor in 3 weeks that matched 59,558 iris images or 0.03 comparisons per second [33]. The fluctuating matching velocity bounds between biometric evaluations was a compromise between speed and accuracy.

8.4.1.1 Multi-core Parallel Computing and Cloud

The usage of data acquisition devices is in high demand by application areas such as aware homes, surveillance, identity resolution, access control, medical prognosis and diagnosis, entertainment, transaction processing and national security. The key characteristics of Big Data are volume, variety and velocity of data. Multimedia data such as photos, videos and sound is accelerating in all three of these characteristics. A tremendous and ever-increasing amount of computing resources is required to process the biometric signatures to turn information into insight.

Cloud computing provides shared resources, software and information to computing devices through the Internet. Distributed computing platforms are connected by any number of networks to support high demand load. The density of Big Data can be defined by the mass of the data within a unit of bytes over a volume of independent cores. Cloud computing can help lessen the density of biometric Big Data. Large quantities of biometric data or processing cores can be distributed across any number of jobs.

Cloud computing technology is shifting towards continuously available cognitive systems that learn over time to be both proactive, reactive and some combination thereof [36, 37]. A shift from highly available to continuously available cloud computing began with a seminal paper by Scadden et al. [37]. Starting with the Nagano Olympics in 1998, a 3-active continuously available computing infrastructure spread over three geographically dispersed regions has supported major sporting and entertainment events such as the Australian Open, Tournoi de Roland-Garros, Wimbledon, the US Open, The Masters, United States Golf Association and the Tony Awards [37]. The team has never experienced a computing resource outage during a major event [37].

The current state of the art within cloud attempts to be autonomic and self configuring [38, 39]. CloudScale predicts computing resource requirements several minutes into the future using burst factors [40]. Another system such as PRESS uses action events to allocate computing resources ahead of the current time horizon [41]. An alternative method of prediction that uses cyclic information is exemplified by AGILE, which scales Infrastructure as a Service (IaaS) 10 min into the future [42]. Longer term prediction methods were proposed by Gmach et al. and Kalyvianaki et al. [43, 44]. The popular Amazon Elastic Compute Cloud (EC2)[3] is a reactive system that enables users to define how the system should respond to high demand loads. Scalr and Rightscale[4] are other reaction based cloud systems [44]. Galante and Bona provide a survey on elastic computing systems [45].

Continuously available cognitive cloud computing technology is rapidly becoming a prerequisite for large-scale biometric systems. As a thought exercise, if the entire United States population at roughly 300 million compared 10 fingerprints to a single probe of 10 additional fingerprints at a rate of 1k comparisons per second, the entire job would require 347 days of continuous computing. The population of China with 1.3 billion people would require a system that can stream many more fingerprint comparison in parallel.

8.4.2 Designs for Parallelism

Ideally, a dynamic biometric cloud would continuously provision and de-provision resources to optimise matching speed on available hardware. Figure 8.11 depicts a high level biometric architecture. Human signatures are acquired from any type of device such as a camera, retina scanner, thermogram measure, and etc. After the signal has been converted into a digital representation, features are extracted from the sample. Within fingerprints, minutiae points are discovered that indicate ridge endings, bifurcations or any ridge ending. An x-coordinate and y-coordinate provide the location within a Cartesian grid while an angle of incidence encodes the direction of the ridge flow. Within the NIST Biometric Image Software (NBIS), an algorithm called MINDTCT extracts minutiae points from an image by convolving minutia templates [46]. Each of the features becomes an element within a feature vector or within the case of fingerprints, a template. Matching technology that supports verification or identification attempts to assert a hypothesis about the feature vector. As an example, another NBIS algorithm called BOZORTH3 measures the similarity between fingerprint templates [46]. Statistical models that use pattern recognition technology such as the probability of an identity given evidence from a voice print and knowledge is another common method of implementing biometric systems. The output of matchers and pattern recognition technologies can be used by any type of application.

Within a cloud, biometric services and processes can be provisioned to reduce the density of the large pending job. The allocated computing resources can be called high performance computing (HPC) and includes parallel, distributed, and grid computing. Typical HPC computers build up from cores, processors, node boards,

[3] http://aws.amazon.com/ec2.

[4] http://www.rightscale.com.

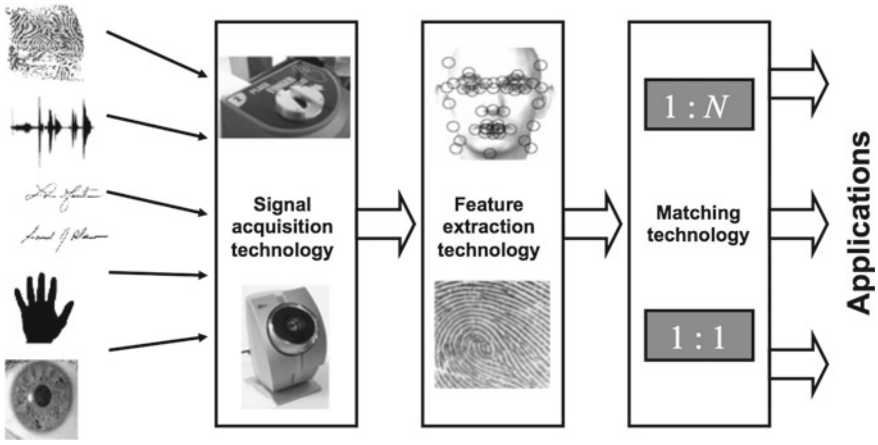


Fig. 8.11 High-level biometric architecture

cabinets, and finally to the entire system. A famous example of an HPC computer that successfully manages a single question-and-answer task consisting of hundreds of parallel processes that need to be distributed over thousands over cores is the IBM Watson computing system. Watson beat Brad Rutter and Ken Jennings on the Jeopardy! game show on 14 January 2011 and was scaled out on over 2,500 compute cores to answer a question in 2–3 s [47]. Before the parallelisation, Watson took 2 hours on a single processor to answer a single question. Over 400 parallel processes were deployed across 72 Power 750 machines [48]. To achieve performance similar to Watson within large-scale biometric applications, feature extraction, matching, and pattern recognition algorithms can be parallelised. As a result, the density of the biometric job decreases as the number of parallel operations increases.

Separation of heavy CPU algorithmic processes within fingerprint matching is depicted within Fig. 8.12. A gallery is the dataset that defines all enrolled fingerprints,

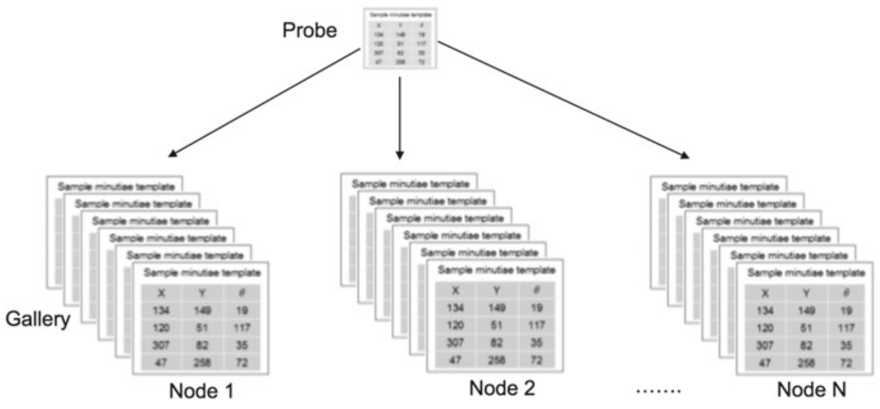


Fig. 8.12 Separating heavy CPU algorithmic process

and is spread out over any number of nodes. A probe is the search fingerprint that is compared against the gallery for similarity scores. The data on each node can utilise independent RAM to reduce I/O while also consuming cycles of independent cores. After the processing of all matchers is complete, the results are merged together into a candidate hit list. The process is similar to the Map Reduce paradigm of Hadoop. The feature extraction modules follow the Streams computing paradigm whereby each fingerprint is pushed through parallel analytic pipelines to build a feature vector for enrolment.

To further reduce the search space and to decrease the mass of data that is searched per probe, techniques such as binning, Support Vector Machines (SVM) as described above, or indexing on subsets of features can be used. Germain et al. depict a novel algorithm to construct constellations of meta-features that can further eliminate the gallery space [1]. Additionally, information about the subject such as gender, age, etc. can be used within evolutionary computing to select the best features to construct a database index [5].

High-Performance Computing (HPC) within a cloud creates a group of coupled computing machines that work together over a plurality of networks to solve deep arithmetic problems. If a problem can be solved by divide and conquer techniques, parallel operations can be performed on portions of the problem. Biometrics is a separable problem as depicted by fingerprint and facial recognition case studies. Figures 8.14 and 8.15 depict examples of distributing both biometric modalities within a cloud environment.

An initial fingerprint identification system that included matching, templetisation, pattern classification and image quality algorithms was deployed onto a single proprietary Solaris machine. The following NIST algorithms were used within an initial fingerprint identification system:

- *MINDTCT*: a template generating algorithm.
- *BOZORTH3*: a matching algorithm that correlates the output from the MINDTCT algorithm.

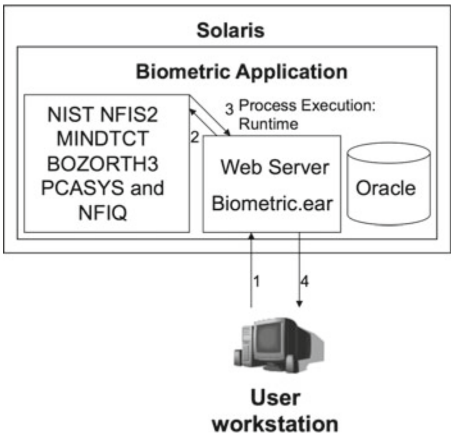


Fig. 8.13 Single System fingerprint identification system

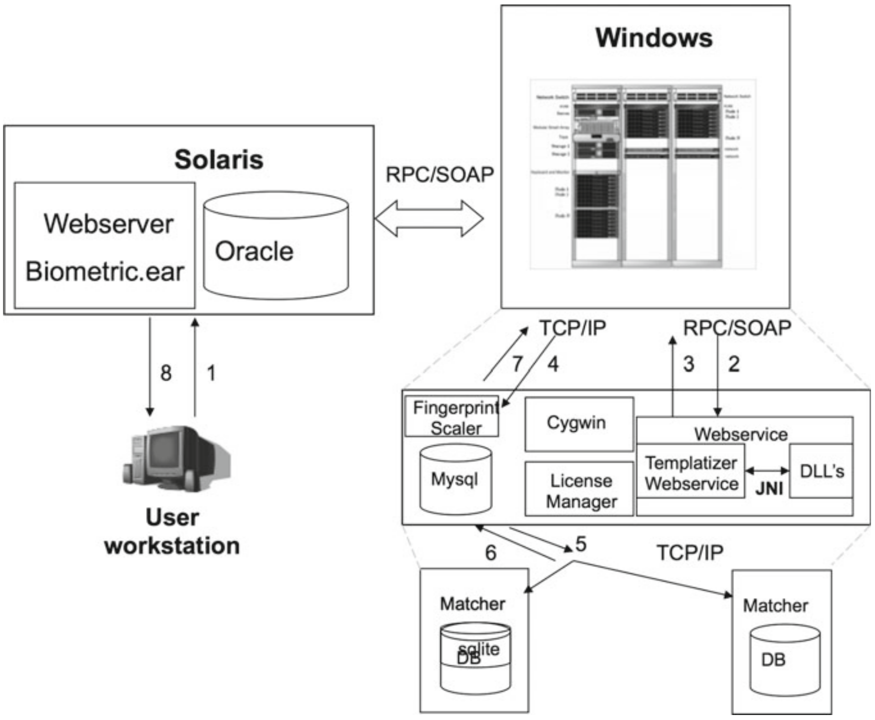


Fig. 8.14 Distributed fingerprint identification system

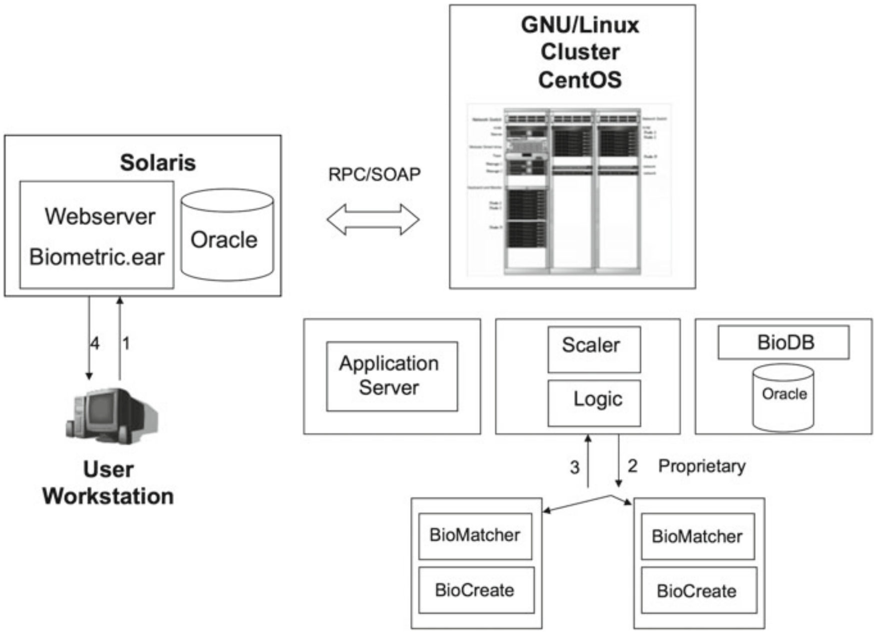


Fig. 8.15 Distributed facial biometric identification system

- *PCASYS*: classifies fingerprint ridge flows as an arch, left loop, right loop, scar, tented arch, whorl, or unknown.
- *NFIG*: ranks the quality of a fingerprint image

Figure 8.13 shows the process flow of a user submitting an Electronic Biometric Transmission Specification (EBTS) file that contains 10 fingerprints to a web server. The web server processes the request and spawns additional processes to create fingerprint patterns, templates, quality scores and matching. On a proprietary Solaris machine, the matching rate reached 20 comparisons per second.

A distributed architecture is shown in Fig. 8.14. A user submits a file to a biometric web server. The web server passes the EBTS to a clustered web service that parses the EBTS format. The EBTS parsing is encapsulated within several web containers to provide a failover in the event that a Java Virtual Machine that is linking to native code fails. Each parsed fingerprint within the Wavelet Scalar Quantisation (WSQ) format to minimise the image size and to prevent lossy based artefacts that would reduce the accuracy of a fingerprint identification system [19]. The WSQs are sent to a scalar machine that templatises the WSQ and pushes the templates to discrete matchers. Each of the matchers contains a shard of data that can be processed in parallel. The result set from each matcher is returned to the scaler for a merge operation. The candidate hit list is returned back to the original biometric web server to be rendered to the user's browser. With only two matchers and on proprietary hardware, the system reached 1,800 comparisons per second within a fault tolerant environment.

A custom built facial recognition system, depicted in Fig. 8.15, is very similar to the distributed fingerprint identification system. An Automatic Identification Biometric System that was developed by L1 Identification and acquired by Safran, was distributed and deployed on proprietary hardware [49]. A user submits a facial image to a web server that is in either bmp, gif, tif, raw or tiff format. The facial image is sent to the facial cluster to be distributed out to parallel services. The scaler receives the facial image and pushes the image to create a token image and template representation. Several thousand images can be enrolled at a much higher rate because several nodes support the ingestion process. Further, the matching nodes are parallelised to apply an assembly of facial matching algorithms in parallel on paired probe and gallery images.

Figures 8.14 and 8.15 depict high available and fault tolerant systems that are defined by Eq. 8.21. The serial availability (SA) is the product of all the component availability (CA) within a particular system:

$$\mathrm{SA} = \prod_{i=1}^{N} \mathrm{CA} \tag{8.21}$$

If a distributed BioMatcher, BioCreate or Matcher goes down, the other nodes will take over the functionality. The web server that provides the biometric Enterprise ARchive (EAR) web application has scripts and monitors to restart the application upon a failure. The scalers on both systems are also monitored throughout the computing life time. However, if the entire cluster, network, or other catastrophic event occurs, the biometric system will be offline. The parallel availability (PA) of the system is equal to 1 minus the product of the downtime for all of the components in a system [37]:

$$\mathrm{PA} = 1 - \left[\prod_{i=1}^{N} (1 - \mathrm{CA}_i) \right] \tag{8.22}$$

From Eq. 8.22, if high available system has a 99.5 % uptime potential, then a 2-active site will have 99.9975 % uptime potential [37]. By extension, a 3-active site will have 99.99999 % up time potential [37]. A 3-active site enables the biometric to not only attain high availability but also continuous availability.

8.5 Conclusion

Biometric data is growing at an accelerating rate, and new methods are needed to turn data at rest and in motion into insights. This chapter discussed some recent and novel developments in the approach taken to enable large-scale biometric identification. Novel work in the areas of fingerprint identification, facial feature selection and high-performance architectures for biometrics was empirically presented. Using kernel functions and support vector machines allows biometric data to be easily separated into clusters, which in turn reduces biometric probe penetration rate. Evolutionary algorithms offer effective ways to learn over time and to dynamically refine which biometric features are important. As a result, the overall dimensionality of the problem is reduced to avoid the curse of dimensionality. The maturity of cloud computing and mobile technologies gives biometric applications unprecedented reach to computation capability. The chapter illustrated how computing cores can be leveraged within several parallel biometric system architectures. Overall, the combination of the techniques described in this chapter enable the use of biometric analytics within large-scale data in a cloud computing environment.

Acknowledgments The authors would like to thank Nalini Ratha, IBM Corporation, for his suggestions and review of the chapter.

References

1. Germain RS, Califano A, Colville S (1997) Fingerprint matching using transformation parameter clustering. Comput Sci Eng 4(4):42–49
2. Bolle R, Connell J, Pankanti S, Ratha N, W A (2010) Guide to biometrics. Springer Professional Computing, Springer. http://books.google.co.za/books?id=wBBScgAACAAJ
3. Ratha NK, Bolle R et al (2004) Automatic fingerprint recognition systems. Springer, New York
4. Gelb A, Clark J (2013) Performance lessons from India's universal identification program. CGD Policy Paper 20
5. Baughman AK (2008) Evolutionary facial feature selection. In: Proceedings of the 2008 GECCO conference companion on genetic and evolutionary computation. ACM, pp 2099–2104

6. Baughman AK, Van Der Stockt S, Greenland A (2010) Large scale fingerprint mining. In: Proceedings of the tenth international workshop on multimedia data mining. ACM, p 1
7. Jain A, Pankanti S (2000) Fingerprint classification and matching. Handbook for image and video processing. Academic Press, London
8. Ross A, Nandakumar K, Jain A (2008) Introduction to multibiometrics. In: Jain A, Flynn P, Ross A (eds) Handbook of biometrics. Springer, New York, pp 271–292
9. Yao Y, Marcialis GL, Pontil M, Frasconi P, Roli F (2003) Combining flat and structured representations for fingerprint classification with recursive neural networks and support vector machines. Pattern Recognit 36(2):397–406
10. Blackburn D, Miles C, Wing B (2009) Biometrics foundation documents. Technical report, DTIC Document
11. Fawcett T (2006) An introduction to ROC analysis. Pattern Recognit Lett 27(8):861–874
12. Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall Inc, London
13. Watson CI, Garris MD, Tabassi E, Wilson CL, Mccabe RM, Janet S, Ko K (2007) User's guide to NIST biometric image software (NBIS). Citeseer
14. Hong JH, Min JK, Cho UK, Cho SB (2008) Fingerprint classification using one-vs-all support vector machines dynamically ordered with naïve bayes classifiers. Pattern Recognit 41(2): 662–671
15. Bishop CM et al (1995) Neural networks for pattern recognition. Clarendon Press, Oxford
16. Bishop CM et al (2006) Pattern recognition and machine learning. Springer, New York
17. Mohanta PP, Mukherjee DP, Acton ST (2002) Agglomerative clustering for image segmentation. In: Proceedings of the 2002 16th International Conference on Pattern Recognition. IEEE, vol 1. pp 664–667
18. Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach Learn 36(1–2):105–139
19. Bradley JN, Brislawn CM (1994) The wavelet/scalar quantization compression standard for digital fingerprint images. In: 1994 IEEE international symposium on circuits and systems, ISCAS'94. IEEE, vol 3, pp 205–208
20. Collins RJ (1992) Studies in artificial evolution. University of California at Los Angeles
21. Engelbrecht AP (2007) Computational intelligence: an introduction. Wiley, New York
22. Goldberg D (1989) Genetic algorithms in search, optimization, and machine learning., Artificial IntelligenceAddison-Wesley, Reading
23. Goldberg DE, Deb K, Clark JH (1991) Genetic algorithms, noise, and the sizing of populations. Complex Syst 6:333–362
24. Holland JH (2000) Building blocks, cohort genetic algorithms, and hyperplane-defined functions. Evolut Comput 8(4):373–391
25. Goldberg DE, Sastry K (2001) A practical schema theorem for genetic algorithm design and tuning. In: Proceedings of the genetic and evolutionary computation conference, pp 328–335
26. Goldberg DE, Sastry K, Latoza T (2001) On the supply of building blocks. In: Proceedings of the genetic and evolutionary computation conference, pp 336–342
27. Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. A Bradford Book
28. Wayman JL, Jain AK, Maltoni D, Maio D (2005) Biometric systems: technology, design and performance evaluation. Springer, New York
29. Liu C, Wechsler H (2000) Evolutionary pursuit and its application to face recognition. IEEE Trans Pattern Anal Mach Intell 22(6):570–582
30. Sim T, Baker S, Bsat M (2002) The CMU pose, illumination, and expression (PIE) database. In: Proceedings of the Fifth IEEE International Conference on automatic face and gesture recognition, pp 46–51
31. Banner CS (1972) The state of development of the FBI's automatic fingerprint identification system. Technical report, FBI
32. Phillips PJ, Grother P, Micheals R, Blackburn DM, Tabassi E, Bone M, (2003) Face recognition vendor test 2002. In: IEEE International Workshop on analysis and modeling of faces and gestures (AMFG), p 44

33. Phillips PJ, Scruggs WT, O'Toole AJ, Flynn PJ, Bowyer KW, Schott CL, Sharpe M (2007) FRVT 2006 and ice 2006 large-scale results. National Institute of Standards and Technology, NISTIR, Gaithersburg
34. Wilson CL, Hicklin RA, Korves H, Ulery B, Zoepfl M, Bone M, Grother PJ, Micheals RJ, Otto S, Watson CI et al (2004) Fingerprint vendor technology evaluation 2003: Summary of results and analysis report. US Department of Commerce, National Institute of Standards and Technology
35. Pelecanos J, Navrátil J, Ramaswamy G (2008) Conversational biometrics: a probabilistic view. In: Ratha N, Govindaraju V (eds) Advances in biometrics. Springer, London, pp 203–224
36. Kephart JO, Chess DM (2003) The vision of autonomic computing. Computer 36(1):41–50
37. Scadden RR, Bogdany RJ, Clifford JW, Pearthree HD, Locke RA (2008) Resilient hosting in a continuously available virtualized environment. IBM Syst. J. 47(4):535–548
38. Shivam P, Babu S, Chase JS (2006) Learning application models for utility resource planning. In: IEEE International conference on autonomic computing (ICAC'06), pp 255–264
39. Zhu X, Young D, Watson BJ, Wang Z, Rolia J, Singhal S, McKee B, Hyser C, Gmach D, Gardner R et al (2008) 1000 islands: integrated capacity and workload management for the next generation data center. In: International conference on autonomic computing (ICAC'08), pp 172–181
40. Shen Z, Subbiah S, Gu X, Wilkes J (2011) Cloudscale: elastic resource scaling for multi-tenant cloud systems. In: Proceedings of the 2nd ACM symposium on cloud computing. ACM, p 5
41. Gong Z, Gu X, Wilkes J (2010) Press: Predictive elastic resource scaling for cloud systems. In: 2010 international conference on network and service management (CNSM), IEEE, pp 9–16
42. Nguyen H, Shen Z, Gu X, Subbiah S, Wilkes J (2013) Agile: elastic distributed resource scaling for infrastructure-as-a-service. In: Proceedings of the USENIX international conference on automated computing (ICAC'13). San Jose
43. Gmach D, Rolia J, Cherkasova L, Kemper A (2007) Capacity management and demand prediction for next generation data centers. In: IEEE international conference on web services, ICWS 2007, IEEE, pp 43–50
44. Kalyvianaki E, Charalambous T, Hand S (2009) Self-adaptive and self-configured CPU resource provisioning for virtualized servers using Kalman filters. In: Proceedings of the 6th international conference on autonomic computing. ACM, pp 117–126
45. Galante G, Bona LCEd (2012) A survey on cloud computing elasticity. In: Proceedings of the 2012 IEEE/ACM fifth international conference on utility and cloud computing. IEEE Computer Society, pp 263–270
46. Watson CI, Garris MD, Tabassi E, Wilson C, McCabe R, Janet S, Ko K (2004) User's guide to export controlled distribution of NIST biometric image software. NIST report
47. Ferrucci D, Brown E, Chu-Carroll J, Fan J, Gondek D, Kalyanpur AA, Lally A, Murdock JW, Nyberg E, Prager J et al (2010) Building Watson: an overview of the DeepQA project. AI Mag 31(3):59–79
48. Epstein EA, Schor MI, Iyer B, Lally A, Brown EW, Cwiklik J (2012) Making Watson fast. IBM J Res Dev 56(3.4):1–15
49. Identification L (ed) (2005) ABIS system developer's guide. 050-138, Revision A, Software version 4.1

Chapter 9
Detection of Demographics and Identity in Spontaneous Speech and Writing

Aaron Lawson, Luciana Ferrer, Wen Wang and John Murray

Abstract This chapter focuses on the automatic identification of demographic traits and identity in both speech and writing. We address language use in the virtual world of online games and text entry on mobile devices in the form of chat, email and nicknames, and demonstrate text factors that correlate with demographics, such as age, gender, personality, and interaction style. Also presented here is work on speakers identification in spontaneous language use, where we describe the state of the art in verification, feature extraction, modeling and calibration across multiple environmental conditions. Finally, we bring speech and writing together to explore approaches to user authentication that span language in general. We discuss how speech-specific factors such as intonation, and writing-specific features such as spelling, punctuation, and typing correction correlate and predict one another as a function of users' sociolinguistic characteristics.

9.1 Introduction

This chapter investigates several facets of how identity and demographic categories are manifested in spoken and written language use, along with approaches to capturing this information for real world analysis, authentication, and talker and writer

A. Lawson (✉) · L. Ferrer · W. Wang
Speech Technology and Research Laboratory (STAR), SRI International,
333 Ravenswood Avenue, Menlo Park, CA 94025, USA
e-mail: Aaron.Lawson@sri.com

L. Ferrer
e-mail: Luciana.Ferrer@sri.com

W. Wang
e-mail: Wen.Wang@sri.com

J. Murray
Computer Science Laboratory, SRI International, 333 Ravenswood Avenue,
Menlo Park, CA 94025, USA
e-mail: John.Murray@sri.com

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_9

identification. The first section details work done by the VERUS (Virtual Environment Real User Study) team under the AFRL (Air Force Research Laboratory) VERUS program, which was tasked with identifying features in virtual world activity that contribute to predicting the real world demographics of the participants involved. In this chapter we specifically focus on virtual world language use, which generally came from two sources: online chat and avatar nicknames. This work was crucial to providing features to determine gender, age group, ethnicity, education level, and nativeness of the real-world participant based solely on activity in the virtual world.

The next section switches focus to spoken language, and the recent progress that has been made in the domain of speaker identification from their voice. We focus on the major problems inherent in speaker identification, both differences inherent to the talker (language, phonetic content, speaker state) and external factors (channel of collection and transmission, noise, reverberation). We examine the recent findings in terms of features (acoustic and prosodic), as well as modeling techniques that have provided breakthroughs in recent evaluations, such as low-dimensional i-vector representations of an utterance and probabilistic linear discriminant analysis (PLDA) for score generation. Further, we discuss the important area of calibration, in particular the issue of maintaining a coherent representation of the likelihood of a speaker given a specific utterance across a range of varying conditions.

The final section presents ongoing work that combines research from both written and spoken authentication and characterization approaches under the DARPA (Defense Advanced Research Projects Agency) Active Authentication program. The goal of this work is to provide continuous authentication of users on their mobile devices using spoken and written inputs on the device, such that if an unauthorized user accesses the device their behavior will quickly reveal them to be an unauthorized user. This continuous authentication will make use of the shared space of language, which covers speech and writing, and the sociolinguistic relationships that emerge from the intersection of language use and personality, background, gender, age, ethnicity, interaction style, etc. Our ultimate goal is to develop a framework for predictive models of users that is robust to incomplete enrollment samples, making use of natural feature correlations across speech and writing.

9.2 Demographics in Virtual World Environments

9.2.1 Background

This section focuses on research into the relationship between virtual world or online linguistic behavior and real-world demographic characteristics, with the goal of automatically predicting major real-world (RW) demographic attributes using only virtual world (VW) behavior. Much of this research came out of the VERUS study [1]. The RW attributes studied include age group, gender, ethnicity, income level, education level, leadership role and urban/rural background, among others. Volunteer partici-

Table 9.1 Data distribution of the VERUS corpus

Game	Turns	Talkers	Tokens
Guardian academy	914	57	2,688
Sherwood	13,149	271	57,843
SecondLife	79	4	392
WoW	2,337	117	56,036
Total	11,214	445	89,521

pants provided their RW demographic information and allowed their online behaviors to be recorded. Over one thousand participants generated data during online activities, including text chat and names chosen for online personae (aka their "avatars"). Hypotheses were gathered from the theoretical sociolinguistics literature, phonology and sound symbolism, semantics, and discourse analysis and from empirical observations of the data collected to generate features. These features were combined in a global model using statistical classifiers that enabled high-accuracy prediction of users RW attributes.

Participants ranged from minors in their early teens to retirees in their 70s from Canada, the United Kingdom, and the United States. Data from four virtual worlds was collected from existing online communities, namely SecondLife and World of Warcraft, and two VWs that were specifically developed for this study, Sherwood and Guardian Academy (see [30]) (Table 9.1).

9.2.2 Features

The focus of feature development was both to understand what factors and behaviors manifest in the text were associated with specific demographic categories and to identify textual elements that could be automatically extracted for use in effective machine learning. A substantial amount of feature research involved understanding the motivation behind a phenomenon in the text (e.g., use of ellipsis) and its association with a demographic category (older users). The primary sources for features were thus identified using both a top-down and bottom-up approach. The top-down features were motivated from findings in the sociolinguistic literature—claims about how males and females used language differently, or how adult language use would differ from teenaged language use, etc. Bottom-up features arose from an examination of the data itself. This is especially important since virtual world and online discourse represent emerging modes of communication and there is a reasonable expectation that theories of traditional spoken and written discourse may be inadequate in this context.

Sources for top-down features mainly came from studies of gender and discourse, beginning with Robin Lakoff's work in 1975, and including studies by [32, 37–39]. References [12, 13] has specifically focused on the interaction between language and

Table 9.2 Gender traits from the sociolinguistic literature

Trait	Gender	Source
Hedging, hesitation, uncertainty	F	Lakoff, Herring
Polite forms	F	Lakoff
Challenging or confrontational forms	M	Herring
Question forms and intonation	F	Lakoff, S&K, Herring
Frequent or gratuitous apologies	F	Lakoff, Herring
Modal verbs	F	Lakoff
Insults, cursing or put downs	M	Lakoff, Herring
Contentious assertions	M	Herring
Supportive and empathetic statements	F	S&K
Sarcasm, self promotion	M	Herring
Agreeing and thanking	F	Herring
Commands	M	S&K

gender in online communities yet many of her findings corroborate the results of the earliest studies by Lakoff. In general, the literature points out that linguistic features associated with females tend to be attenuative, indirect and cooperative, while male linguistic behavior tends to be more adversarial, direct and independent. Table 9.2 summarizes the major traits identified in the literature that were investigated in this study.

In addition, new phenomena that were observed to correlate highly with demographic classes were also added to our set of features (see Table 9.3). These include typographic variations, which are associated with age differences, as well as more subtle distinctions between slurs and direct and indirect apologies.

Additional features came from consultation with Subject Matter Experts (SME) on virtual worlds, freely available lexical class databases, and features developed from the structure of virtual world environments. We divide these features into two sets: lexical features and structural features.

Lexical features include unigram probabilities of words in the list of Internet slang and emoticons, unigram probabilities of other nonstandard words/shorthands (e.g., *ur* (*you are*), *thn* (*then*), *im* (*I'm*), *qust* (*quest*)), features related to disfluencies, features related to person addressing, and features related to occurrence of foreign characters.

We also studied features representing sentence complexity and structure, including vocabulary size of a participant, maximum length of 10 % most frequent words, fea-

Table 9.3 Sociolinguistic features from data observations

Trait	Demographic	Example
Use of slurs	Male	"You jerk!"
Direct apologies	Female	"I'm sorry"
Indirect apologies	Male	"Ooops", "my bad"
Standard emoticons	Female	:) : (
Use of all caps	Youth	"STOP BEING DUMB"
Frequent use of ellipsis	Adult	"if you bring up your questlog..."
Commas, apostrophes	Adult	"we're done. let's turn in"
Lowercase 'i' for 'I' and 'u' for 'you'	Youth	"u losted to 4 pokemno"
Single word texts	Youth	"come", "yo"

tures related to discourse markers, and features related to grammaticality (estimated based on the normalized likelihood of the sentence from a state-of-the-art statistical English parser adapted to game text chat).

We further used features from databases of lexical categories, including the "Dictionary of Affect in Language" (DAL) [42] and the Linguistic Inquiry Word Count (LIWC) [34]. The DAL is an instrument designed to measure the emotional meaning of words and texts. It does this by comparing individual words to a list of 8,742 words that have been rated by people for their activation, evaluation, and imagery. Each word in the lexicon also receives a score according to "pleasantness," "activity," and "imagery." Then we computed the average of these scores for the sentences contributed by a participant and average counts of words belonging to each of these three categories, and used them as DAL features. The goal of LIWC was to identify a group of words that tapped basic emotional and cognitive dimensions often studied in the social sciences, health and psychology and use them as features reflecting disposition, personality, etc.

We also developed a set of "structural" features, inspired by conversation analysis and based on observations from game subject matter experts on player behavior, to quantitatively represent the different participation patterns of participants in a text chat conversation. Structural features for a participant include the percentage of sentences and 'turns' (i.e., exchanges between speakers in a conversation) from that participant out of all sentences and turns in the session respectively, and the average number of words per sentence and per turn of that participant.

We also studied features related to "structure" of game chat. For example, "silence" (no chat) durations from a participant in a game session may be synchronized with other gaming activities (i.e., the participant was possibly busy with other in-game activities hence could not contribute much text chat). Other structures that were useful were the use of positive and negative extreme case formulations—for example "that was the worst game ever" and expressions of self-affirmation (e.g., *I'm the best*). We further extracted features based on social network analysis, by capturing who is talking to who and how frequently they do it in the text chat.

An additional set of features was also developed based on the names users chose for their avatar and VW character. Since the name chosen by a particular player generally reflected information about the player in terms of gender, age, personal interests, ethnicity, etc., many effective features were extracted from components of the avatar names. This included sound symbolism-based features [33] related to gender (e.g., female names having high, front vowels, sibilants and ending in ‘a’), typographic features related to age (use of capitalization, numbers and special characters), and features based on real-world cultural references in the name (e.g., youth names referencing elements from Harry Potter books). For a more detailed overview of the avatar naming phenomena see [21, 22].

9.2.3 Machine Learning and Findings

We further explored machine learning approaches to identify predictive features from a variety of features for detecting real-world (RW) target variables based on virtual world (VW) data. We applied these techniques to predicting the following RW target variables: gender, age group, community, ethnicity, English nativeness, RW and VW leadership and followership.

Two classification approaches were found to perform well on the features extracted from text chat from players: AdaBoost and linear kernel Support Vector Machines (SVMs), using RW target variable labels for supervised training. For feature normalization, we compared the effect of mean/variance normalization and rank normalization on RW target variable prediction accuracy. For feature selection, we compared forward–backward feature selection, the SVM-RFE (Support Vector Machine-Recursive Feature Elimination) algorithm, and logistic regression for fusion parameters.

We evaluated precision, recall, and overall classification accuracy for target variables, from tenfold cross-validation for model training/testing and feature selection, by polling text chat from all four VW together. We built multi-class classifiers for predicting RW target variables. For example, for predicting age group, the overall classification accuracy is 82.54 %; for predicting community, the overall classification accuracy is 83.51 %. The most predictive features for age and gender are presented in Tables 9.4 and 9.5.

Gender conclusion both tend to corroborate the findings of the sociolinguistic community, while adding new findings to our understanding of how gender traits manifest in language use.

9.2.4 Combined Results

Final results were obtained by combining the language-based features with features from VW economic activity, movement, dress, and game play activity (e.g.,

Table 9.4 The most predictive features for age group (Adult >24, YoungAdult (18–24), Youth (<18)

Rules for age group
• Adult has larger average number of words per turn than Youth
• Youth use all uppercase (shouting) much more than nonyouth
• Adult has larger average modal words per sentence than Youth
• Youth has larger average number of name addressing per sentence than YoungAdult
• Youth has larger average number of disfluencies per sentence than Adult
• YoungAdult uses more Internet slang per sentence than Youth
• Youth tend to use single word utterances at a rate almost double than adults and to use shorter phrases in general
• YoungAdult uses more extreme case formulations than Youth
• The use of proper contractions (e.g., "I'm" vs "im") increases with users' age
• The use of both periods '.' and commas ',' increases with users' age
• Adults' use of personal pronouns is double that of Youth
• Adults are twice as likely to use traditional emoticons as Youth and 3–4 times more likely to use ellipsis

Table 9.5 Most predictive features for gender

Rules for gender
• Females tend to use hedging forms more than men, including modal verbs, expressions of uncertainty and questions
• Males tend to use more offensive language and slurs than females, though females use more attenuated swears (e.g., 'darn', 'crud')
• Females tend to apologize more than males, though males use more indirect apologies ('Ooops')
• Females are more likely to agree and to express empathy than males
• Females are more likely to use traditional emotions than males, but males are more likely to use lewd emotions
• Both females and males choose avatar names or nicknames that tend to conform to the findings of the sound symbolism literature (Jespersen, Ohala, Gordon and Keith)
• Female avatar names tend to end in 'a', have sibilant consonants ('sh') or front vowels 'i, y, e'
• Male avatar names tend to end in back vowels ('u', 'o') or consonants, especially back or alveolar stops

dueling). Combination was done using the WEKA toolkit [11] and feature selection techniques based on work by [9]. The main goal of the program was arriving at human-understandable and human-usable rules—thus precision for each category and each feature was of paramount importance, with recall being of lesser importance. Results for the main categories are presented in Table 9.6.

Precision was prioritized over recall in this program, since the overall goal was to develop human-understandable collections of rules that could effectively pick out

Table 9.6 Final combined results for the major program demographic targets

Category	Precision (%)	Recall (%)
Gender	98	52
Approximate age group	88	13
Ethnicity	83	37
English as native language	77	22
Education	79	52
Socioeconomic status	83	67
Income level	85	35

demographic traits with high accuracy. Thus, even if a rule only applied to a minor subset of the total population, so long as it was precise, it was of high utility.

9.3 Detecting Identity in Large Collections of Spontaneous Speech

Automatic speaker recognition is the task of recognizing the person speaking in an audio recording. It can be classified into two main tasks: identification and verification. Speaker identification aims at identifying the speaker present in the recording among a set of known speakers. Speaker verification, sometimes also called speaker detection, on the other hand, aims at deciding whether the audio recording corresponds or not to a certain speaker of interest.

The task of speaker identification can be solved through a series of speaker verification queries against all target speakers, as explained for the related tasks of language detection and identification by [1]. Furthermore, given its binary nature, verification is easier to define and evaluate than identification. For these reasons most research in the area has been done for the speaker verification task.

Speaker verification is used in security applications to verify whether a speaker is who he claims he is. It is also used in applications that search for specific speakers within a large database of speech. In the last few years, speaker verification performance on clean telephone data has reached extremely good performance levels, with error rates below 1 % for recordings of around 2 min of duration making the technology adequate for use under these conditions.

On the other hand, performance on harder recording conditions involving noise, channel distortion, reverberation, and other nonideal conditions is severely affected and can reach unusable levels. Nevertheless, much progress has been made in these areas in recent years. This section covers some of the techniques that have lead to major improvements under these challenging scenarios.

9.3.1 Overview

The core speaker verification task is defined as determining whether a specified target speaker is speaking during a given segment of speech. More explicitly, one or more samples of speech data from a speaker (referred to as the "target" speaker) are provided to the speaker recognition system. These samples are the "training" or "enrollment" data. The system uses these data to create a "model" of the target speaker's speech. Then a sample of speech data is provided to the speaker recognition system. This sample is referred to as the "test" segment. Performance is judged according to how accurately the test segment is classified as containing (or not containing) speech from the target speaker.

Metrics that reflect accuracy are related to a typical hypothesis test (i.e., based on false positives (referred to as false alarms) and false negatives (misses)). In this work, we report equal error rates (EER), where false alarm and miss rates are equal, or the false alarm rate at a particular miss rate. The performance of a system over the range of possible operation points is generally represented in a Decision-Error Tradeoff (DET) curve, which plots the relationship between false alarms and misses over all points.

9.3.1.1 Challenges

As for any detection task, the main challenge of speaker recognition is extracting features that will represent a speaker independent of variations that can occur in the observations. Minimizing the intra-class variability while maximizing the interclass variability is our goal.

Speech is a complex signal, and many possible variations of that signal exist for the same individual. During the previous few years, the community has tackled the problem of extrinsic variability and how to factor out extrinsic variability from the speaker model (sometimes referred to as channel compensation in articles). This kind of variability is detrimental to high accuracy speaker recognition. Indeed, recorded speech varies as a function of many factors that are not a function of the speaker's identity, including: acoustic environment (e.g., background noise), channel (e.g., microphone, handset, recording equipment), high signal-to-noise ratio (SNR), audio degradation through compression, speaker's physical condition (emotion, intoxication, illness), what is said (text-independent versus text-dependent), and speaking context (level of formality, planning, language).

9.3.1.2 Approaches for Mitigation of Undesired Variability

Mitigation of the undesired variability can be performed at different levels in the system from the feature extraction to the last stage of system fusion. This section summarizes several different approaches, focusing on recent methods implemented in state of the art systems.

(1) *Feature Diversity*
A successful approach to speaker verification is to combine different knowledge sources by separately modeling them and by fusing them at the score level to produce the final score that is later thresholded to obtain a decision. Combinations of systems are most successful when the individual systems being combined are significantly different from each other.

Prosody—the intonation, rhythm, and stress patterns in speech—is not directly reflected in the spectral features. As a consequence, these features show great effect in combination with traditional features [18]. The state-of-the-art approach to extracting prosodic features is to compute the pitch and energy contour in the signal using Legendre polynomial coefficients.

Standard spectral-based features include perceptual linear prediction (PLP) features and mel-frequency cepstrum coefficients (MFCC). In addition, many spectral-based features were developed specifically for noise robustness under the DARPA RATS (Robust Automatic Transcription of Speech) program. Medium duration modulation cepstrum (MDMC) features [29] extract modulation cepstrum-based information by estimating the amplitude of the modulation. Power-normalized cepstral coefficient (PNCC) [17] features use a power law to design the filter bank as well as a power-based normalization instead of a logarithmic one. Mean Hilbert envelope coefficient (MHEC) features [36] use a gammatone filter bank instead of the Mel filter bank, and the filter bank energy is computed from the temporal envelope of the squared magnitude of the analytical signal obtained using the Hilbert transform. Subband autocorrelation classification (SACC) [23] provides a pitch estimate from an estimator that is trained using a multilayer perceptron, allowing for a robust prosodic system implementation, which we call PROSACC (Prosodic Subband AutoCorrelation Classification) in this article.

(2) *Advanced Modeling*
Recently, the speaker verification community has enjoyed a significant increase in accuracy from the successful application of the factor analysis framework. In this framework, the i-vector extractor paradigm [5] along with a Bayesian backend is now the state of the art in speaker verification systems. An i-vector extractor is generally defined as a transformation where one speech utterance with variable duration is projected into a single low-dimensional vector, typically of a few hundred components.

The low rank of the i-vector itself opened up new possibilities for the application of advanced machine-learning paradigms that would have been otherwise too costly with the very high dimensionality used by most earlier systems. Probabilistic linear discriminant analysis (PLDA) [14, 35] has proved to be one of the most powerful techniques for producing a verification score. In this model, each i-vector is separated into a speaker and a channel part, analogous to the formulation in the Joint Factor Analysis framework [15].

A simple and quite effective approach for robustness against undesired variability is to include data with the corresponding variability during training of the PLDA model [24], so that the model can learn the appropriate intraspeaker variability under the conditions of interest. On the other hand, the components of the i-vector extractor seem much less sensitive to exposure to a new type of variability and can be kept untouched without much, if any, effect in performance.

(3) *Metadata Extraction*
Metadata information about the audio recording can be used to affect the parameters of the models, allowing adaption to the specific conditions of the recording. Rather than relying on either annotated data, or developing specific systems for each type of variability, a universal audio characterization system can be used to extract metadata information based on the i-vector [7]. This enables the system to detect if an audio recording contains certain kinds of noise, channels, or the speaker's gender or language.

(4) *System Fusion and Calibration*
Fusion of systems is usually performed either at the score level or at the i-vector level. At the score level, system fusion is generally performed using logistic regression with a cross entropy objective [2], the standard fusion approach in speaker recognition. This approach offers the benefit of producing calibrated scores, treatable as log-likelihood ratios, which are ideal for forensic comparisons and decisions.

As mentioned in [7], the metadata extracted from the universal audio characterization system can be used during fusion to adapt the output score to the signal's conditions. A modified version of the logistic regression fusion algorithm is used so that log-likelihood ratios are still produced but are biased depending on the metadata between the enrollment and test utterances.

9.3.2 Robustness to Undesired Variability

In this section, we highlight the impact of the approaches described above for different types of degraded audio conditions and other extrinsic variations.

9.3.2.1 Channel, Noise, Reverb, Vocal Effort and Language Variation

To evaluate speaker recognition accuracy on multiple types of variability, SRI created the PRISM (Promoting Robustness for Speaker Modeling) dataset [8], building on data previously collected by the Linguistic Data Consortium and creating trials from waveforms degraded by adding noise or reverberation. The PRISM data set is available online at https://code.google.com/p/prism-set/.

In Fig. 9.1, we show the benefit of different mitigation approaches by showing the increase in speaker recognition accuracy for every step of the pipeline. Note that results for different conditions are not comparable since they involve different speakers and other factors that affect the absolute performance. Comparisons should be made within condition and across systems.

The conditions defined in the PRISM set and represented in the horizontal axis of the figure are:

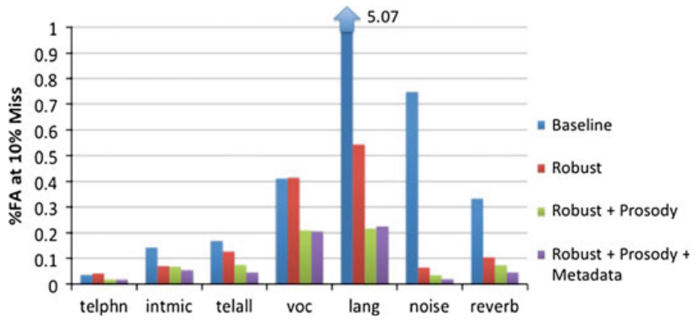


Fig. 9.1 SRI's speaker verification results on the PRISM set

- *telphn*: Telephone calls over telephone channels.
- *intmic*: Microphone recordings in an interview setting.
- *telall*: Telephone calls over telephone channels and other microphones.
- *voc*: Vocal effort: low and high.
- *lang*: Trials made of languages other than English.
- *noise*: Clean signals degraded with real noise samples at different SNR levels ranging from 20 to 6 dB.
- *reverb*: Clean signals degraded with artificial reverb at reverb times (RT) of 0.3, 0.5, and 0.7 s.

The baseline system is a standard i-vector/PLDA recognition pipeline on MFCC features, without the mitigation mechanism for the variations of interest.

The robust system uses an enhanced PLDA model designed to be robust to the variations of interest by adding data with these types of variation during training. This system also includes other techniques that add robustness to the system, namely i-vector length normalization [10], an LDA step for dimensionality reduction, and i-vector adaptation, where the mean i-vector over each condition is subtracted from the corresponding i-vectors before PLDA modeling. Improvements are highly significant, reducing error by a factor of ten times on the noise condition while also improving results for "cleaner" conditions like telephone calls.

The robust + prosody system is a fusion of the robust MFCC system and a robust prosodic system. We see that an additional improvement can be observed in most conditions. Finally, we enable metadata extraction and handling in the robust + prosody + metadata system to obtain additional improvements for most conditions except language and vocal effort. These conditions were not represented as classes for the metadata extractor due to lack of training data for them and, hence, could not be appropriately predicted.

The figure shows how the different approaches for mitigation reduce the effect of the undesired variations, in some cases reducing the false alarm rates at 10 % miss rate by an order of magnitude.

9.3.2.2 Highly Degraded Channels

The DARPA RATS program aims at developing robust processing methods for speech acquired from highly degraded transmission channels. The audio recordings [40] used in the RATS program are severely degraded with additive noise, channel-convolved noise, bandwidth limitations, and frequency shifting. Telephone conversations are retransmitted over eight different military transmitter/receiver combinations. All the data was retransmitted across all the channels and re-recorded, resulting in more than 100,000 files. The core languages from which speakers are selected are Levantine Arabic, Farsi, Dari, Pashto, and Urdu. In the speaker verification task each speaker model was trained using six different sessions from different channels. A trial was designed using one speaker model and one test session. The duration of each training and testing session was 3, 10, 30 or 120 s depending on the condition.

SRI's system was composed of five different features: PLP, MDMC, MHEC, PNCC, PROSACC. For the i-vector framework used by all feature streams, we used universal background models (UBMs) with 2,048 diagonal covariance Gaussian components trained in a gender-independent fashion. The PROSACC systems used 1,024-component UBMs. The i-vector dimensions of 400 were further reduced to 200 dimensions by LDA (in the case of PROSACC, 200D i-vectors were reduced to 100D), followed by length normalization and PLDA.

The systems are fused at the i-vector level by concatenating each i-vector from each stream into a single vector before employing the PLDA backend. The i-vector dimensions are first reduced using LDA, and only after concatenation does a second dimensionality reduction shrink the total dimension to 200. Fusion of systems at the score level was performed using logistic regression.

Results from four core conditions are provided in Fig. 9.2, showing the relative performance of the five acoustic features with both HMM (Hidden Markov Model)

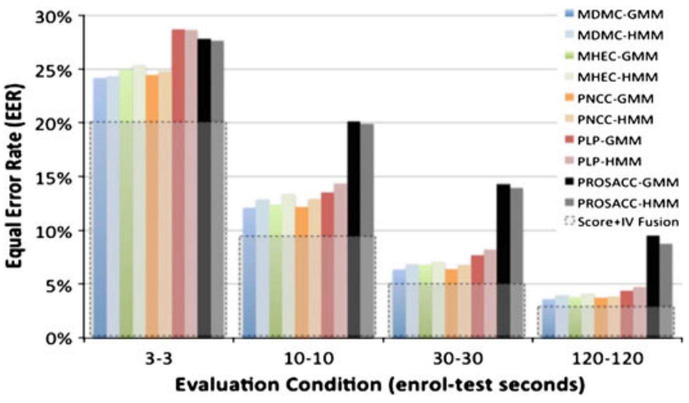


Fig. 9.2 SRI speaker recognition system results on the DARPA RATS development set for different combinations of train and test durations

and GMM (Gaussian Mixture Model) SAD (Speech Activity Detection), as well as the gain from the final score plus i-vector fusion system (in dashed lines). For more details on the results presented in this figure, see [27, 28]. For all durations, the MDMC and PNCC features with GMM SAD had the least errors. The fusion system was always significantly better than any single system, benefiting in particular from the PNCC features and substantially from the inclusion of PROSACC, despite the system's low accuracy on its own.

9.4 Work in Progress: Identification of Identity Factors from Both Speech and Text in the Active Authentication Program

This section describes an ongoing project, LinguaKey, to combine both speech and text data for continuous authentication of mobile device users through their language usage. The goal of this work system is to provide continuous authentication of users actively performing routine tasks on a mobile device. These include telephone conversation, spoken device interaction (e.g., "SIRI"), text chat, instant messaging and email. This research builds a profile of a user's distinctive linguistic usage and generates models to provide an ongoing means of ensuring that only authorized individuals have access to their mobile device.

Our approach is to identify discriminative features based on ***behavioral*** and ***cognitive*** dimensions of individual linguistic variability. These factors represent deeply ingrained parts of a user's way of thinking and behaving, of which the user may not even be conscious. The notion is that since these factors are difficult to control they will be resistant to spoofing. Behavioral dimensions include features from speech and text production, i.e., the idiosyncrasies of how users actually produce spoken or written language. In speech this includes factors such as *intonation*, *speech rate*, and *speech energy/frequency*. In text, behavioral features will target *spelling errors*, *punctuation style*, use of *abbreviations*, *sentence* and *word length,* among other factors. Cognitive features are focus on sociolinguistic factors derived from linguistic activity that provides clues about the individual user's background, personality, and approach to interpersonal relations. An individual's language use is highly colored by sociolinguistic background, with influences from gender, age, ethnicity, native language, region, level of education, social class, dialect, and others. We further leverage the rich information in word sense information and linguistic context to provide parameters for identifying users based on personality traits and their typical approaches to interacting with others.

9.4.1 Approach

The LinguaKey process begins with the capture of speech and text on the mobile device (see Fig. 9.3). This will be accomplished through tools that provide hooks into

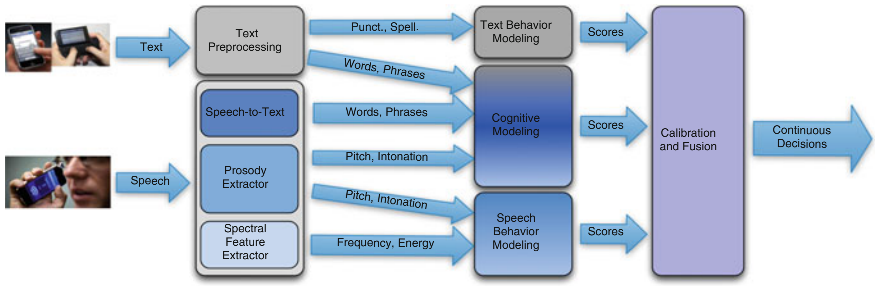


Fig. 9.3 LinguaKey feature extraction and modeling process

the Android operating system to record keystrokes and the microphone on the mobile device. Three initial features are extracted from speech: (1) the words being spoken are identified using a speech-to-text tool, (2) the intonation contours and prosody are identified and (3) the spectral features are extracted from every frame. For text, the input to the keystroke logger is collected, both in its raw form and the final sent message. This text then feeds a module that identifies behavioral features from the text dealing with spelling and typography. In an analogous speech behavior module, the spectral and prosodic features are modeled as a short duration biometric. The cognitive modeling component receives words from both the spoken and text inputs, as well as intonation, pitch, etc., and identifies linguistic features that encompass spoken and written language use. The three modeling components output scores as log-likelihood ratios to the fusion and calibration engine which is continuously using current and recent past information to determine the probability that the current user is the authorized user.

The system architecture to support this processing relies on a client/server configuration, in which the algorithms to process the data, extract features and score reside on a remote server that communicates via wireless or cellular connection. In Fig. 9.4, the basic flow of Linguakey is depicted, with a *database* serving to contain audio and speech organized by enrolled users. A *modality* server contains models developed from different modalities of data (spoken, written, etc.). The mobile device collects data in real time from users, which is passed to the core server, which processes the data, extracts features, scores the data against the proper model and stores the audio in the database. The client allows access to the system offline for research purposes such as extracting and studying the data and running authentication experiments.

9.4.2 Data Collection

Volunteer participant data collection is a crucial component of this effort, since data is required to understand the relationship between how language behavior and cognition manifests itself in both spoken and written modalities. Four types of data

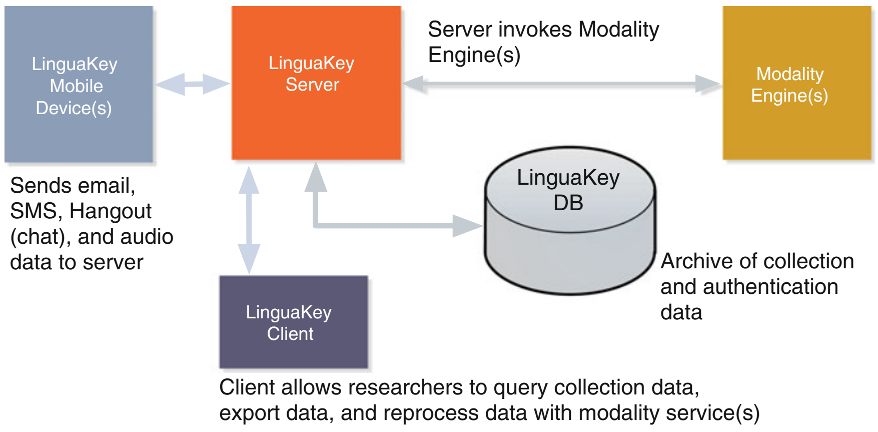

LinguaKey Mobile Device(s)
LinguaKey Server
Server invokes Modality Engine(s)
Modality Engine(s)
Sends email, SMS, Hangout (chat), and audio data to server
LinguaKey DB
Archive of collection and authentication data
LinguaKey Client
Client allows researchers to query collection data, export data, and reprocess data with modality service(s)

Fig. 9.4 The LinguaKey client/server architecture

were collected in this effort over 90 sessions: (1) phone calls, (2) personal digital assistant-type queries, (3) text chat and (4) emails.

The final tally of collected data was 1,300 audio files collected totaling 1,200 min of speech, which amounted to about 15 min per user. On the text side, 1,800 lines of text and 21,000 words were collected and 18,000 corrections were recorded. We are calling this collection the SpOntaneous Mobile Language Use Corpus (SOMLUC). The eventual goal is to release this data and make this widely available to the research community (Fig. 9.5).

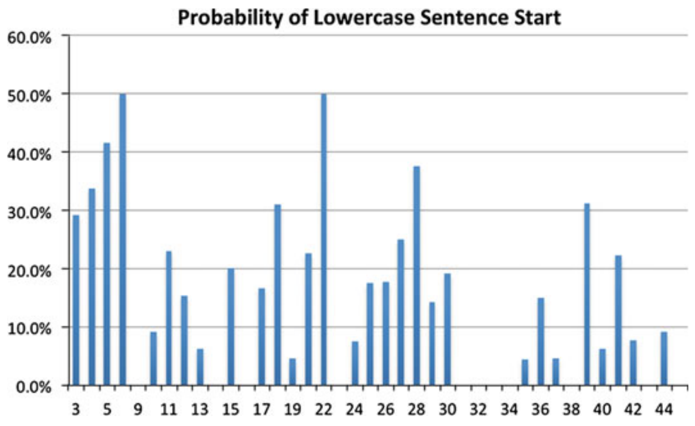


Fig. 9.5 Probability of starting a sentence with a lowercase letter in the SOMLUC corpus

9.4.3 Research and Results

We have implemented a large set of 102 features covering everything from basic behavioral traits (punctuation, capitalization, word length, etc.) to higher level features focused on word semantics, emotional content, and language frames. Speech-specific features include pitch, intonation, speech rate, and cepstrum; text specific features focus on typographic and orthographic information and corrections. We have also implemented a set of language general features that are the core of this effort, including words, speech act frames, semantics, and sociolinguistic trends.

We are currently working to understand and take advantage of entailment—the fact that features are correlated across individuals because they are associated with the same set of personality and background traits. Part of our research program is to look at how features interact to predict higher level aspects of a person's cognitive and behavioral traits, traits that effectively characterize users but about which they have little control.

Since our goal is to combine our 100 or so features into groups that correlate with higher level personality, cognition or demographic factors, understanding entailment within and across speech and writing is crucial.

In Tables 9.5 and 9.6 we present data from SOMLUC showing the variability between participants in terms beginning a sentence with a capital letter and probability that they will make a written correction. On their own, these two pieces of information are useful in terms of both characterizing individuals, and certainly correlate with aspects of an individual's personality in that careful users will tend to make corrections and use standard written forms, as do older users and more educated users based on evidence from the VERUS program.

A major focus on the LinguaKey program is to understand not just which features are useful, but to understand the relationship between features in both spontaneous writing and speech. This is important in its own right in terms of basic research as a means of clarifying the relationship between demographics and personality and types of linguistic phenomena. It is also important for the goal of authentication of users, since one of the main problems encountered in supervised system is availability of sufficient training data both in terms of raw amount and in terms of coverage of phenomena important for characterizing the user in the feature space (Fig. 9.6).

Thus being able to predict absent features from the presence of other features is of high value in modeling an individual when limitless enrollment data is not available. In support of this approach Fig. 9.7 shows the extent to which features interact and bundle, revealing higher level correlations between the phenomena and allowing us to group features into higher level classes that predict other features. For example, users who are careful to use the proper forms of "I" and "you" (rather than "i" and "u") are naturally going to have a higher rate of corrections, since the form of the text is important to them. Likewise, users who fail to capitalize sentences are inversely correlated with correction rate, with the same rationale.

Next steps for LinguaKey will focus on the relationship between features in the spoken language, such as uptalk or filled pauses, and sociolinguistic factors in the

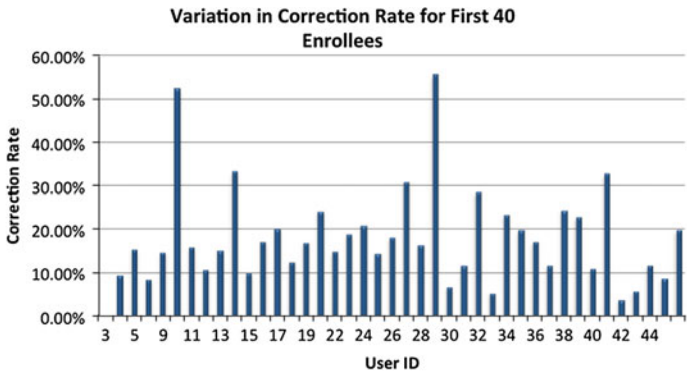


Fig. 9.6 Differences in correction rate among users in the SOMLUC corpus

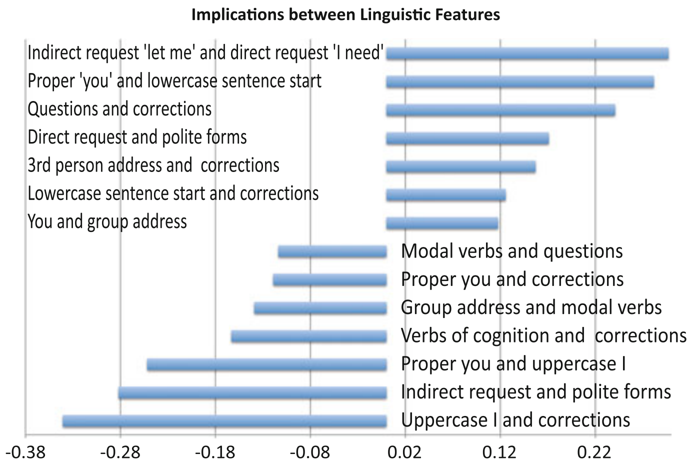


Fig. 9.7 Positive and inverse correlations between features in the SOMLUC corpus

written spontaneous language use, such as use of emoticons, questions and other phenomena associated with hedging. For example, in the VERUS data it was found that females tended to use emoticons much more than males, and that emoticons played the role of "softening" or hedging the utterance they were associated with. Likewise, "uptalk" (the persistent rising of pitch at the end of statements—Ref. [4]) is considered to be a kind of hedging phenomenon. Our hypothesis is that individuals who have pervasive uptalk in their speech will also tend to use written hedging phenomena (such as emoticons and modal verbs) more frequently. These findings will allow us to move from a set of isolated and uncorrelated features to the capability to represent the feature space as a natural gestalt, where users can be characterized effectively in a predictive space based on naturally clustering of sociolinguistically related features.

9.5 Conclusion

The future of identity detection lies in the intersection of higher level features and machine learning—both in speech, where prosody and phonetic content are beginning to play a crucial role and writing where sociolinguistic factors are having a significant impact. In speaker identification the use of highly accurate neural network-based phone identification systems at the senone level [25] are being combined with i-vector modeling approaches to help eliminate the influence of phonetic content on speaker traits. In text processing the VERUS research demonstrated how the findings of sociolinguistics could be applied to a completely new domain and form the basis for an effective system to extract demographic information from spontaneous text chat in the virtual world environment. Ongoing work targets the intersection of text, speech, and identity to identify the commonalities and correlations between features across language use, and factors that reflect the background, behavior, cognition, and physical makeup of the individual.

Acknowledgments This material is based on work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract D10PC20024. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA or its contracting agent, the U.S. Department of the Interior, National Business Center, Acquisition & Property Management Division, Southwest Branch. This material is based on work supported by United States Air Force and DARPA under Contract No. FA8750-13-C-0280. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force and DARPA. Approved for Public Release, Distribution Unlimited.

References

1. Brümmer N (2006) FoCal II: toolkit for calibration of multi-class recognition scores. Software available at http://www.dsp.sun.ac.za/~nbrummer/focal/index.htm. August 2006
2. Brümmer N et al (2007) Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006. IEEE Trans Audio Speech Lang Process 15(7):2072–2084
3. Brümmer N, van Leeuwen D (2006) On calibration of language recognition scores. In: Proceedings of the speaker and language recognition workshop, Puerto Rico, Odyssey
4. Ching M (1982) The question intonation in assertions. Am Speech 57:95–107
5. Dehak N et al (2010) Frontend factor analysis for speaker verification. IEEE Trans ASLP 19(4):788–798
6. Dieterle E, Murray J (2011) Virtual environment real user study: design and methodological considerations and implications. J Appl Learn Technol 1(1):19–25
7. Ferrer L et al (2010) A unified approach for audio characterization and its application to speaker recognition. In: Proceedings of the speaker and language recognition workshop, Odyssey 2010, Brno
8. Ferrer L et al (2011) Promoting robustness for speaker modeling in the community: the PRISM evaluation set. In: Proceedings of SRE11 analysis workshop, December 2011
9. Flach P, Lachiche N (2001) Confirmation-guided discovery of first-order rules with tertius. Mach Learn 42(1–2):61–95

10. Garcia-Romero D, Espy-Wilson C (2011) Analysis of i-vector length normalization in speaker recognition systems. In: Proceedings interspeech, Florence
11. Hall M et al (2009) The WEKA data mining software: an update. SIGKDD Explor 11(1):10–18
12. Herring S (1994) Gender differences in computer-mediated communication: bringing familiar baggage to the new frontier. In: American library association annual convention, Miami
13. Herring S, Paolillo J (2006) Gender and genre variation in weblogs. J Sociolinguist 10(4): 439–459
14. Kenny P (2010) Bayesian speaker verification with heavy-tailed priors. In: IEEE Odyssey 2010—the speaker and language recognition workshop, 29 June 2010
15. Kenny P et al (2008) A study of inter-speaker variability in speaker verification. IEEE Trans ASLP 16(5):980–988
16. Kim C, Stern R (2010) Feature extraction for robust speech recognition based on maximizing the sharpness of the power distribution and on power flooring. In: Proceedings of IEEE international conference on acoustics speech and signal processing (ICASSP), pp 4574–4577
17. Kim C, Stern R (2012) Power-normalized Cepstral coefficients (PNCC) for robust speech recognition. In: Proceedings of IEEE international conference acoustics, speech and signal processing (ICASSP), Kyoto, 25–30 March 2012
18. Kockmann M et al (2011) i-Vector fusion of Prosodic and Cepstral features for speaker verification. In: Proceedings of interspeech, Florence
19. Lakoff R (1975) Language and woman's place. Harper & Row, New York
20. Lawson A et al (2012) Sociolinguistic factors and gender mapping across real and virtual world cultures. In: 2nd international conference on cross-cultural decision making, San Francisco, July 2012
21. Lawson A, Murray J (2014) Identifying user demographic traits through virtual-world language use. In: Ahmad MA, Shen C, Srivastava J, Contractor N (eds) Predicting real world behaviors from virtual world data. Springer, London
22. Lawson A, Taylor N (2012) The names people play: exploring MMOG players' Avatar naming conventions. In: Canadian games studies association symposium, May 2012
23. Lee B, Ellis D (2012) Noise robust pitch tracking by subband autocorrelation classification. In: Proceedings of interspeech, Portland
24. Lei Y et al (2012) Towards noise-robust speaker recognition using probabilistic linear discriminant analysis. In: Proceedings of IEEE international conference acoustics, speech and signal processing (ICASSP), Kyoto, 25–30 March 2012
25. Lei Y et al (2014) A novel scheme for speaker recognition using a phonetically-aware deep neural network. In: ICASSP 2014, Florence
26. Martin A et al (1997) The DET curve in assessment of detection task performance. In: Proceedings Eurospeech, pp 1899–1903
27. McLaren M et al (2013a) Improving speaker identification robustness to highly channel-degraded speech through multiple system fusion. In: Proceedings of ICASSP, Vancouver
28. McLaren M et al (2013b) Improving robustness to compressed speech in speaker recognition. In: Proceedings of interspeech, pp 3698–3702
29. Mitra V et al (2012) Normalized amplitude modulation features for large vocabulary noise-robust speech recognition. In: Proceedings of IEEE international conference on acoustics, speech and signal processing (ICASSP), Kyoto, 25–30 March 2012
30. Murray J et al (2012) Virtual environment real user study (verus): final project report. AFRL-RY-WP-TR-2012-0286, Air Force Research Laboratory
31. NIST SRE12 Evaluation Plan (2012) http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12 evalplan-v17--r1.pdf
32. O'Barr WM, Atkins BK (1980) Women's language or powerless language? In: McConnell-Ginet S, Borker, N, Thurman R (eds) Women and Language in Literature and Society. Praeger, New York, pp 93–110
33. Ohala J, Hinton L, Nichols J (1994) Sound symbolism. Cambridge University Press, New York
34. Pennebaker J, Booth R, Francis M (2007) Linguistic inquiry and word count: LIWC2007—operator's manual. LIWC.net, Austin

35. Prince S (2007) Probabilistic linear discriminant analysis for inferences about identity. In: IEEE 11th international conference on computer vision (ICCV), pp 1–8
36. Sadjadi S, Hansen J (2011) Hilbert envelope-based features for robust speaker identification under reverberant mismatched conditions. In: Proceedings of IEEE international conference acoustics, speech and signal processing (ICASSP), pp 5448–5451
37. Shuttleworth J, and Keith G (2000) Living Language. Hodder Education
38. Tannen D (1984) Conversational style: analyzing talk among friends. Ablex, Norwood
39. Tannen D (1994) Gender and discourse. Oxford University Press, Oxford
40. Walker K, Strassel S (2012) The RATS radio traffic collection system. In: Odyssey 2012—the speaker and language recognition workshop, 25–28 June 2012
41. Wang W (2011) Automatic detection of speaker attributes based on utterance text. In: Interspeech, Florence, 27–31 August 2011
42. Whissell C (2009) Using the revised dictionary of affect in language to quantify the emotional undertones of samples of natural language. Psychol Rep 105(1):509–521

Part IV
Multimedia Data Modeling, Search and Evaluation

Chapter 10
Evaluating Web Image Context Extraction

Sadet Alcic and Stefan Conrad

Abstract Images on the Web appear with other textual contents—referred to as *Web Image Context*—providing valuable information to the image semantics. Unfortunately, HTML documents are usually cluttered with multiple different contents to different topics and therefore the right image context needs to be precisely determined in order to deliver high quality descriptions. Several methods that automatically determine and extract the Web image context from Web documents have been applied in different applications over the years. However, in these applications context extraction is only a preprocessing step and therefore the quality of the extraction task has rather been evaluated on its own. To sum up, there is hardly information about which extraction method to choose in order to get best results. Keeping this necessity in mind, an evaluation framework that objectively measures and compares the quality of different Web Image Context Extraction (WICE) algorithms will be the main subject in this book chapter. The main parts of the framework are a large ground truth dataset consisting of diverse Web documents from real Web servers and objective quality measures tailored to fit the special characteristics of the image context extraction task. In order to demonstrate the capabilities of the framework, common extraction methods from the literature are implemented and integrated into the framework. Finally, the evaluation results are summarized and discussed.

S. Alcic (✉) · S. Conrad
Department of Databases and Information Systems, Institute for Computer Science,
Heinrich-Heine-University of Duesseldorf,
Universitaetsstr. 1, 40225 Duesseldorf, Germany
e-mail: alcic@cs.uni-duesseldorf.de

S. Conrad
e-mail: conrad@cs.uni-duesseldorf.de

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_10

10.1 Introduction

In recent years, the World Wide Web has become an integral part in our lives with millions of documents for almost every conceivable topic. This tremendous amount of data is only useful when its information is explicitly accessible. While the textual contents of Web documents are retrievable using proven indexing techniques from Information Retrieval (IR), the automatic indexing of images by means of their semantics is still an open challenge. The difficulty is known as the Semantic Gap [1], which describes the lack of a reliable mapping from the raw representation of images to the semantic meaning of the depicted objects.

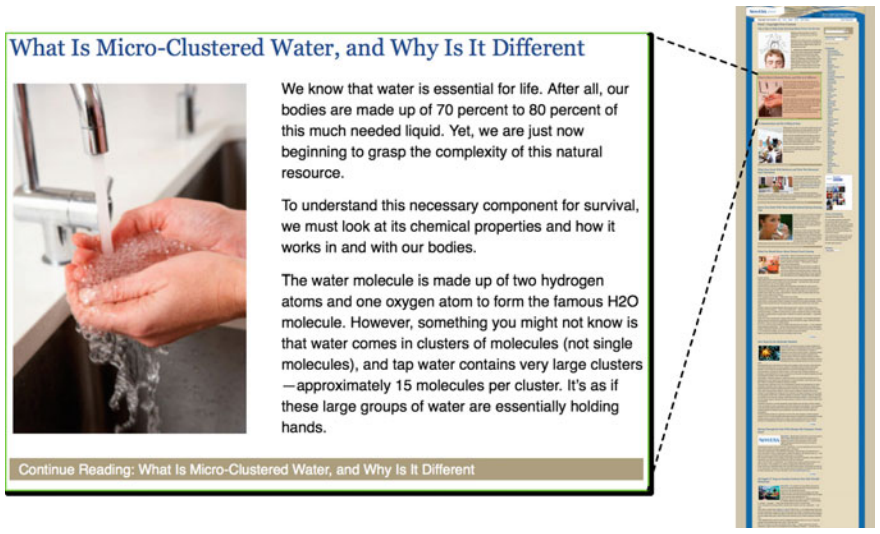
Fortunately, images on the Web appear in a context within textual articles or explicit image captions that provide meaningful information to the semantics of the depicted objects. This additional information can be exploited to provide useful image annotations and thus to avoid the direct confrontation with the Semantic Gap. By *Web Image Context* we are not referring to some meta attributes of an image found in a Web document like filename, alternative text and others—these can also be useful but are out of the scope. For now, we are only talking about visible text that appears nearby an image on a Web page.

Figure 10.1 shows a typical Web image and its context as placed on a news overview page. Although the image context does not exactly describe the depicted scene, we can extract keywords like "water" and "holdings hands" from the context. Furthermore, we get a background story about micro-clustered water and its properties which is beyond the semantics of the image alone. Whether this additional information is valuable or the opposite depends on the application and is not the subject of this work.

However, Fig. 10.1 further illustrates how the article is embedded in a Web document. We can observe that this Web document is built up from contents of multiple topics, advertisements, social network add-ons and other structural data. In order to gather the best possible descriptions, we have to apply an extraction algorithm that will separate the text content belonging to the image from the complete text of the document.

The idea of using Web image context as description source is not new and has been addressed by several researchers in different applications using different extraction methods. Each applied extraction method seems eligible, but to our knowledge there is no objective investigation and comparison of extraction methods. Most of the evaluations done so far (see Related Work) examined the main application and in this way are implicit, reflecting only an impact of the extraction method on the main application.

We fill this gap by presenting an evaluation framework that uses a huge dataset of Web documents that were automatically collected from real Web servers and custom designed evaluation metrics that fit the characteristics of the problem and ensure the objectivity that we demand. Common extraction methods are implemented within the framework and are used to demonstrate the functionality of the framework.



What Is Micro-Clustered Water, and Why Is It Different

We know that water is essential for life. After all, our bodies are made up of 70 percent to 80 percent of this much needed liquid. Yet, we are just now beginning to grasp the complexity of this natural resource.

To understand this necessary component for survival, we must look at its chemical properties and how it works in and with our bodies.

The water molecule is made up of two hydrogen atoms and one oxygen atom to form the famous H2O molecule. However, something you might not know is that water comes in clusters of molecules (not single molecules), and tap water contains very large clusters —approximately 15 molecules per cluster. It's as if these large groups of water are essentially holding hands.

Continue Reading: What Is Micro-Clustered Water, and Why Is It Different

Fig. 10.1 Web image context example on a typical News Web page (extracted from NewsUSA, http://www.copyrightfreecontent.com, on September 6th, 2014)

The proposed framework was first introduced as a research paper [2] at the Multimedia Data Mining Workshop of the KDD 2010 conference. Later on, it was successfully applied to measure the performance of newly introduced extraction methods in [3, 4].

This work is structured as follows: first, we are going to present and discuss related work in the next section. Following that, in Sect. 10.3 the problem of Web Image Context Extraction will be formalized. The evaluation framework and its modules are the subject of Sect. 10.4. There we first give general overview to the framework followed by a deeper description of the particular components and their concrete implementation. Finally, the chapter is concluded with a reflection on the evaluation results.

10.2 Related Work

Image context has been used as description source in several applications over the years. As a result, different context extraction techniques have been proposed in the literature. However, since WICE was only a preprocessing step in these applications, there is hardly direct evaluation of this task. At this point, we give an overview to the few evaluation scenarios that deal with image context to a certain degree.

Souza-Coelho et al. [5] list four sources for image descriptions within a Web document, namely *description text* (image filename or alternative text), *meta tags* (located in the HTML head of the document), *full text*, and *surrounding text* *passages* (see window of surrounding terms in Sect. 10.4.4). Within an evaluation task they analyzed the image retrieval performance based on the different description sources applied for image indexing. They have found that the surrounding text passage consisting of 20 terms before and after the image (passages of 10, 20 and 40 terms were separately inspected) performed best in their evaluation task. Although other methods of WICE have not been evaluated, this result supports our assumption that beside all structured data that can be directly extracted from Web documents as image description, Web image context is most valuable and this encourages us to search for reliable extraction methods.

Another system, where the surrounding text passage plays a great role is Image Rover [6]. The image indexing in this system is based on textual and visual descriptions of images, while the textual descriptions are obtained from plain text of web documents. Different document parts are weighted depending on their parent tag properties, where the surrounding text weight is among the highest. The system performance has been evaluated by applying the *target test paradigm* [7], which tests how efficiently a system performs in finding a target image in the data collection. As in [5], the presented evaluation shows only the impact of one context extraction method to the image retrieval task. However, there is no comparison, since no other extraction method has been tested.

Tian et al. [8] extract visual, relational, and textual image descriptions for Web image classification. The textual information are gathered from the sibling nodes of Web images in the DOM representation. Based on the associated descriptions, the images are classified and the *classification performance* is analyzed. This is another example where only the impact of one WICE method on another application (namely image classification) was evaluated. As an alternative textual context extraction method, VIPS [9] is mentioned but not applied in the scenario due to its high complexity. VIPS (Vision Based Page Segmentation) is a page partitioning algorithm, that will be described in Sect. 10.4.3 briefly.

The effectiveness of VIPS was only judged indirectly by its inventors. In [9], Cai et al. proposed and tested VIPS's segmentation quality by employing 5 human experts who classified the segmentation results to perfect, satisfactory, fair and bad. He et al. [10] applied VIPS for context extraction and have evaluated Web image retrieval and clustering efficiency, without focusing on the context extraction, nor comparing with other extraction methods.

To our knowledge, the only direct evaluation on WICE has been done in [11]. In order to evaluate their extraction method (see Monash Extractor in Sect. 10.4.3), Fauzi et al. have built the following testing environment:

Document Collection. As data set, 100 Web documents were randomly selected across various categories in Alexa Web Directory [12] and manually labelled by 30 volunteers. Each volunteer processed 10 Web pages, which means that every page was labelled 3 times. This resulted in 3 labelled sets of data, and as the final set, they took the broadest context from the 3 available. Banners and layout graphics were filtered by checking image dimensions: only images with a width and height greater than 45 pixels and a width-height ratio between $\frac{1}{2}$ and 2 were processed. The outcome is a set of 1,019 image-and-context pairs.

Evaluation Measures. The evaluation of the proposed extraction algorithm is done within a system-based framework where the *Precision* and *Recall* measures are applied. In this context, Precision is the percentage of correctly (test on exact matching) extracted image descriptions over the total extracted image descriptions and recall is the percentage of correctly extracted image description over the total actual number of image descriptions in the dataset. For both measures, the average value over all extracted image-context pairs is computed. Furthermore, the *average processing time* needed to extract all images and the corresponding contexts per Web page has been estimated.

Evaluated Methods. Within this study, two methods were evaluated: the Monash Extractor and a VIPS-based extractor. VIPS has one central parameter called *Permitted Degree of Coherence* (PDoC), which regulates the granularity of the segmentation. In this task the PDoC value has been varied from 5 to 7, similar as in [13], so that three different extractions were computed using VIPS. This resulted in four precision and respectively recall values in total. However, the average processing time has only been estimated for the Monash Extractor.

As mentioned above, this work is the only direct evaluation task of image context extraction. However, the evaluation task has some weaknesses. First, the document collection consists of only 100 documents which is very small in order to represent the diversity of a repository that consists of billions of documents. Secondly, the ground truth data has been combined by taking the broadest text segment from three human labelled extractions, but taking the broadest text means usually accepting errors in all of the manual labels. Here possibly an intersection of the three would fit better. Finally, the evaluation metric is not focused on one mapping (image-context-pair) but on the complete Web document. The particular mappings are compared on exact matches, meaning that if the ground truth data and the extracted data match 99 %, the metric treats them as they do not match at all. We think that this criterion is very strict and does not reflect the performance of extraction task adequately.

10.3 WICE Problem Formulation

Web Image Context Extraction (WICE) deals with finding the image context of an image within a Web document. To be more precisely, it describes the process of, given a Web image, estimating those textual parts from the hosting Web page that are semantically related to the image; we refer to these textual parts as the *Web image context*.

A common representation for Web documents is the node-based Document Object Model (DOM) tree, where each content is wrapped within so-called *tag-nodes*. The inner content units (*text*-, *image*- or *object-nodes*) are represented as leaf nodes.

Based on this representation, we can emphasize three main parts within the WICE task:

- the *Web image*, represented by an image node,
- the *hosting Web page*, which contains the image node, and of course
- the *image context*, a subset of the documents text contents, which has to be estimated.

Having outlined these concepts, a more formal definition of WICE can be given.

Definition 10.1 Let I be a Web image and D be the Web document that contains I. The WICE task can be denoted as a two dimensional function

$$f(I, D) = C_I,$$

where

$$C_I = \{t_i \mid t_i \text{ is the content of a text node element in } D\}$$

is the set of text node contents representing the image context.

From this general formalization of the WICE task, we can derive that the Web image and the Web document serve both as input, while the returned image context represents the output.

10.4 Evaluation Framework Design

Figure 10.2 presents a flow diagram showing the basic parts the WICE evaluation environment is composed of.

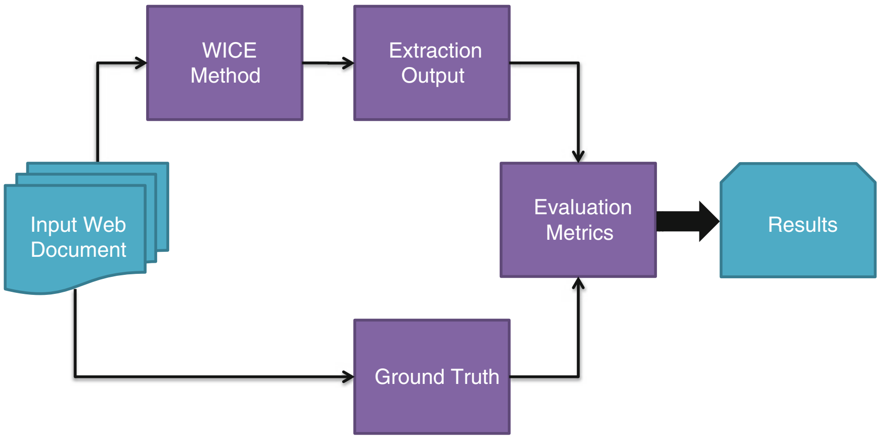


Fig. 10.2 Overview of the different modules of the evaluation framework

The *input data* in this evaluation process are Web documents that are collected from different Web servers. On one side, these documents are processed by a human expert who labels the documents and in this way creates the *ground truth data* for the evaluation. On the other side, there are concrete *WICE methods* that analyze the input Web documents and automatically compute image-context pairs that are assumed to belong together. Both outputs, the ground truth as well as the extraction results are of the same *output format*. In the next step, these outputs are compared to each other by applying suitable *evaluation metrics*. The result of the evaluation framework is the outcome gained during that final comparison step which can be visualized and presented to the user in different graphs.

In the following, the particular modules of the framework will be presented in more detail.

10.4.1 Input Data

The evaluation framework estimates the performance of WICE methods by applying them on real Web documents. These documents are served as input data for the WICE methods and are therefore an essential part of the framework.

Web documents are mostly written in HTML by different authors around the world with different programming experience and designing skills. This variety is further encouraged by the loosely restricted standard of HTML which allows the author to produce a needed output in different ways, which can be misleading for the later analysis, e.g., a bold tag with an increased font size can produce the same output as an header tag. Even more challenging are deficient HTML documents that are affected with missing (unclosed) tags but are accepted and correctly presented in browsers, which apply different techniques to repair the ill-formed documents. In this framework, the input documents are therefore passed through the JTidy parser [14], which is able to generate valid HTML code as the parsers in most browsers do. This preprocessing method is imperative at this step since without a well-defined HTML structure, it is not possible to build the DOM tree which is the core presentation for further analysis.

Another difficulty for the extraction process becomes apparent when we look at the desired contents in an input document. As described in the introduction, Web documents are cluttered with different kinds of contents which belong to functional, structural and the main information content. This implies that there are also images that belong to these different parts. It is therefore necessary to find out which images belong to the real content of a Web page and which are only structural or navigational, because it does not make sense to search for the image context of non-content images. In our implementations, we therefore apply a rule-based image filter, which detects the unwanted images and excludes them from being analyzed. Structural and functional images serve as background or click-areas in different parts of a Web page. We empirically found out that these images mostly are characterized by certain image dimensional properties which allow us to define filtering rules based on these values. We have applied the following filtering rules which are very similar to those suggested in [5, 11, 15]:

1. Filter images with a width or height smaller than 60px.
2. Filter images with a width-height ratio greater than 2.5 or smaller than 0.4.
3. Filter images of Graphic Interchange Format (GIF).

The first rule excludes too small images, because they are assumed to be either decoration images or button areas. The second rule filters images whose width-height ratio exceeds a predefined range. The affected images are mostly background graphics, website logos or banners which do not belong to the main content of the Web page. Finally, the third rule filters all GIF images which mostly represent advertisement images and rotating banners.

Web documents are hypermedia documents which contain links to other documents and also include many objects of other media types by pointing to the URLs of these data objects. For example, images are included in a Web document using the source attribute of the image tag that gets an URL of the image as input. For our evaluation framework it was necessary to collect a set of Web documents and to store this set locally in order to ensure a reliable and fast access on the documents. However, URLs in Web documents are mostly encoded relatively in respect to the actual location of the document on the Web server. If we download the Web document source to store it locally, the relative links and paths become invalid and have to be adapted. It is possible either to store all of the needed resources in the same directory structure as they are presented on the Web server, or—the way we have chosen for this evaluation framework—to convert all relative links and URLs to absolute ones and thus provide the access on the desired resources (scripts, images) without the need to store them locally.

To sum up, an input Web document is preprocessed in three steps in our evaluation framework as depicted in Fig. 10.3. First, the document is checked for validity and possible errors are corrected using the JTidy library. In the second step, all URLs are converted to absolute paths. And finally, some images that are not of the main part of the Web document are filtered. *Filtered* does not mean that they are removed from the original Web document but just that they are not passed to the next analysis steps.

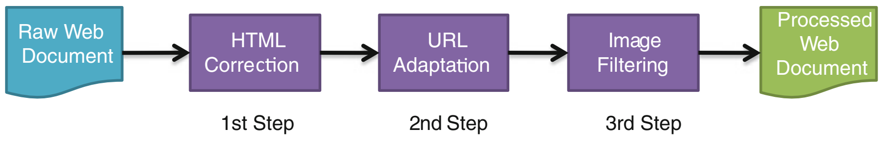


Fig. 10.3 Web document preprocessing

10.4.2 Ground Truth Dataset

The ground truth dataset plays a very important role during the evaluation process, since it defines the desired extraction result that a WICE method should output. However, the creation of a ground truth dataset that is representative for all the documents on the Web comes with several problems that will be described next.

The convenient way to create the ground truth data is to let a human expert do it manually. Although the determination of the context of an image can be subjective and therefore not always deterministic—there are parts of a Web page that in some situations belong to the context, in others not (e.g., the complete article text can be omitted when a detailed image caption is present)—we assume that the human expert at least has a good idea of what the image context should be. On the other hand, the effort needed to determine the image context manually is tremendous. The average time needed to manually process a Web page containing 20 image-context pairs using a special labelling tool [16] that alleviates the process took 183 s. As a consequence, the processing of 1,000 documents would take approximately 51 h of work. This brought us to the decision to think about an automation of the process.

Finding a general extraction method that is able to create our ground truth dataset seems to be impossible since if such a method would exist, the problem of WICE would be solved. On the other hand, for one specific Web document, it is possible to write a tailored extraction application (supported by a human expert) that exactly determines the image context for the images within this document. This extractor can be based on rules determining the desired DOM structure.

Using textnodes as smallest image context units in the ground truth, seems to be inflexible and to privilege DOM-based extraction methods. However, a broad analysis of several documents used in this evaluation task showed that a further partitioning of textnodes into, e.g., the particular words, would not be more useful, since all image context parts are complete textnode texts and never portions of them.

Example 10.1 An example extraction rule for collecting the image context in a Web document:

```
// 1st rule
IF ImgTag has ParentNode with Attribute a = "b"
  THEN
    extractAllTextNodes under ParentNode as ImageContext;
    exit;
// 2nd rule
...
```

The order of the rules is very important, since the images covered by different rules may overlap. For some documents, rules can be much more complex, e.g., they can exclude particular textnodes under a parent node.

Example 10.1 contains a typical extraction rule that is based on particular HTML tag properties. The example rule is performed for a given ImgTag which contains the image for which the context is sought. The rule condition dictates if the given tag contains a parent node with an attribute “a” having a value “b” then collects all text nodes under the estimated parent node as the image context.

Instead of creating a tailored image context extractor for each Web document—this would be even more time consuming than directly estimating the image context—we observed that there are collections of Web documents for which one specific image context extractor would fit. These are Web documents that are part of a common Web site (e.g., Wikipedia) that is maintained using a *Content Management System* (CMS). Usually these documents are based on a common structural *template*. There are two different methods to gather documents that share a common template, either by crawling a specific Web site or by recalling a Web page that changes very often such as the start page of many news portals.

We decided to apply both methods to collect the data collections, the first for static documents like the pages collected from Wikipedia, and the latter for dynamically changing documents like the news pages. To be more detailed, while the first method could be implemented straight forward by crawling through several documents on one Web page, the algorithm for the latter is more complex:

```
LOOP until NrOfCollectedDocuments = M
  current := load source from www.example.com

  IF (difference(current, last) > threshold)
   THEN
    store current to disk
    last := current
  ENDIF
  wait N minutes
```

The algorithm consists of one loop that is repeated until the desired number of documents *M* has been collected. Inside the loop, first the source code of the Web document (here *www.example.com*) is downloaded as `current`. Then the difference of the current document to the last stored is estimated. We have tried different algorithms for computing the difference of two Web page sources and we decided to compare the sources line-by-line. If the difference is greater than a predefined threshold, we store the current document to disk since it seems to differ significantly from the last stored. After that the “last” content is overwritten by the current and the algorithm waits a small period before repeating the whole process.

Finally, the result is a collection of Web documents that have different contents that are based on one template. For this template, a human expert can now define rules that extract the images and corresponding context. To do this, the expert only needs to analyze a small portion of the Web documents, which is an enormous reduction of the needed effort compared to manual image context extraction. Figure 10.4 depicts different pages from NewsUSA.com domain that are based on the same template. As we can see, there are many elements as the header, the footer and the navigation

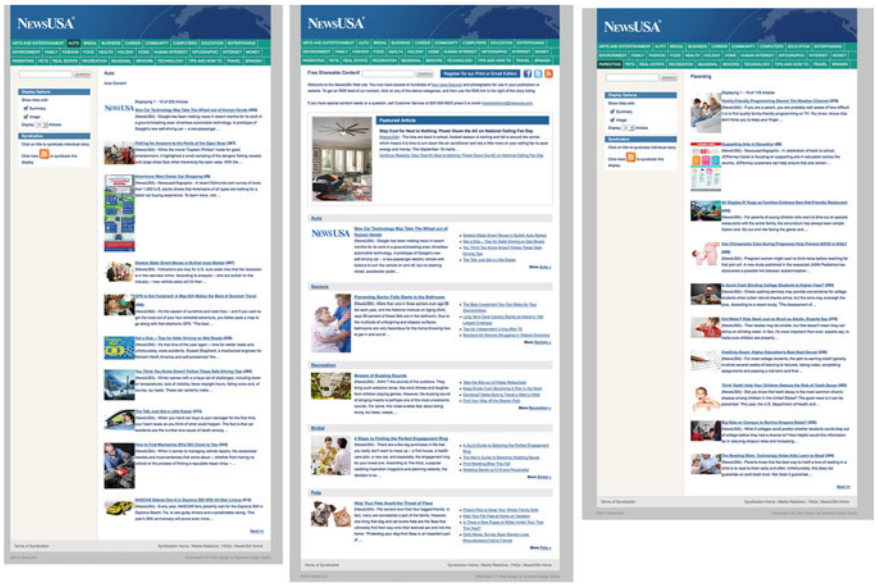


Fig. 10.4 Three different pages based on the same template (extracted from http://newsusa.com)

Table 10.1 Test collections with total number of documents and images

Collection	#Documents	#Images
BBC	1,077	7,878
CNN	874	11,612
Golem	789	3,061
Heise	79	1,403
MSN	375	9,264
New-York Times	556	10,927
Spiegel	1,076	36,310
Telegraph	530	10,503
The Globe and Mail	735	15,808
Wikipedia	3,000	6,728
Yahoo! (english)	3,737	41,170
Diverse (manual)	79	901
Total	12,907	155,565

bar that hardly differ. Also there are high structural similarities between the content articles.

The properties of the resulting test collections are summarized in Table 10.1. According to these values, to our knowledge this is the greatest existing test collection of extracted image-context pairs.

10.4.3 Applied WICE Methods

In order to test the framework capabilities, we have implemented some of the commonly used extraction methods that will be described next.

10.4.4 Window of Surrounding Terms

A very fast and therefore commonly used method to extract image descriptions is the window-based method. The *N-Terms Window* algorithm has been applied in [5, 15, 17, 18].

In most Web pages, the image context is placed next to the corresponding image (as depicted in Fig. 10.1). Following this assumption, the idea behind this method is to extract text that surrounds the image in the HTML source code as image context.

```
Data: Web document $d$, image $I$, window size $N$
Result: Set $T$ of terms surrounding the image
$S \leftarrow getImagesAndTerms(d)$;
$i \leftarrow indexOf(I, S)$;
for $k \leftarrow (i - \frac{N}{2})$ to $(i + \frac{N}{2})$ do
    if $S[k]$ instanceOf $term$ then
        $T.add(S[k])$;
    end
end
```

Algorithm 1: N-Terms Window

The extraction method is described in Algorithm 1 in pseudocode notation. As input, it needs a Web document, an image of this document and a window size N. In the initialization step (line 1), the document is transformed into a sequence of terms and images S. The terms are the particular words of the textual content of the given Web page and the images correspond to the images in the document embedded by <img> tags. The elements in sequence S are ordered by their position in the original HTML code of the document. Further, we denote the ith element of S as $S[i]$. The transformation can easily be accomplished by a linear scan of the HTML code.

In the next step (line 2), the position of the image in the sequence S is determined and stored as index i. This step is necessary in order to know where to position the window frame in the next step.

The main part of the algorithm (lines 3–7) is a loop which iterates over S from $S[i - \frac{n}{2}]$ to $S[i + \frac{n}{2}]$ and collects the visited terms. Additionally, if the window exceeds the borders of S, the iteration index has to be adapted (not contained in the pseudocode algorithm). Provided that the element $S[i]$ is an image, the Web image context is estimated as in Fig. 10.5.

The parameter n determines the size of the frame of terms surrounding the image and has to be estimated. Souza-Coelho et al. [5] have used different frame sizes in

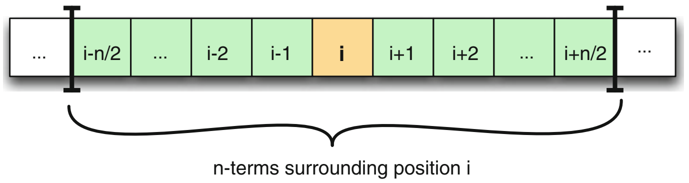


Fig. 10.5 Frame of n terms surrounding an image in a list of terms and images

their evaluation studies while $n = 20$ has performed best (experiments have been carried out with $n = 10$, $n = 20$ and $n = 40$). Sclaroff et al. [17] have applied a frame of 30 terms, while the number of terms before the image was set to 10 and respectively after the image to 20 terms, which can be well justified by an observation of Feng et al. [15] stating that for 73 % of the examined images the context appears after the image and for 27 % of the images context appears before the image.

The *time complexity* of the described method is *linear*, since the Web page transformation, the image index estimation, and the window of terms computation are sequentially executed and each of them is linear in time depending on the length of the document.

10.4.5 Paragraph Extractor

As the name implies, this extraction method (applied in [19, 20]) aims to find the nearest paragraph of an image and considers this paragraph to be the image caption.

This is a DOM-based approach that uses the parent-child relation between DOM elements to determine the context paragraph by estimating the parent tag element of the given image element, which includes text elements in its subtree. All text elements under the estimated parent tag are considered as parts of the image context.

```
Data: Web document $d$, image $I$
Result: Set $T$ of text nodes representing the image caption
$D \leftarrow createDOM(d)$;
$i \leftarrow find(I, D)$;
while $\neg$ $containsTextnodes(i)$ do
    $i \leftarrow i.getParent$;
end
$T \leftarrow$ getTextnodes($i$);
```

Algorithm 2: Paragraph extraction

The algorithm proceeds as follows: in the initialization step (line 1–2), the DOM tree D of the input document and the image element I corresponding to the input image are initialized. The main part is the while-loop, which starting at the image element, walks the tree upwards, until a parent element is reached that includes text elements in its sub-tree. In the final step (line 6), the text elements under the determined parent are collected as the image context.

Another, possibly more expressive way to describe this algorithm is to point out that the extracted context is always the text contained in all sibling nodes of an image node in DOM tree. Thus this method is also known as the siblings extractor.

Computation complexity of the algorithm depends linearly on the length of the document in regard to building the DOM tree and further the while-loop is at maximum in $O(d \cdot t)$, with d as the depth of the DOM tree and t as the total number of nodes of the tree.

10.4.6 Monash Extractor

The monash extractor [11] can be viewed as an extension to the paragraph extractor and has been introduced by Fauzi to handle the different template types in which an image can be embedded in. The basic idea relies on the concept of *list pages* and *detail pages* [21] in which generally extractable data records can be placed. List pages usually contain a list of many records with similar structure (e.g., a listing of products of a catalog), while detail pages contain detailed information on one particular record.

Depending on how listed and unlisted data records can principally be modeled in HTML, Fauzi distinguishes three classes of Web images: listed, unlisted and semi-listed images. *Listed images* are two or more images that are ordered within a regular pattern of HTML elements (see Fig. 10.6a, segment 3–7). In the DOM tree each image is placed under one sub-root node (see Fig. 10.6c). *Unlisted images* are standalone images that can be placed at any position in the Web page (see Fig. 10.6a, segment 1 or segment 2 and Fig. 10.6b). *Semi-listed images* own the same visual properties as listed images. However, in the DOM tree the segments of particular images are not each placed under one root node, but they are all together under a root node while the visual separations are made by special HTML elements (see Fig. 10.6d). Although semi-listed image is a justifyable layout style for images, we could not found any sample of image-context where it was applied. One reason is surely the fact that, in modern well structured documents the layout style has changed towards grouping cohesive contents into separate DOM blocks.

The algorithm proceeds as follows. The input is a DOM tree and the image node whose context has to be identified. There are three state variables that maintain the current state of the algorithm: `stateText` and `state` keep the current number of text nodes under the actual node and respectively its previous value. Both are set to 0 at beginning. The variable `stateChangeTwice` is true, if the state variable has changed twice during the actual run.

Starting at the image node, the algorithm walks upward the DOM tree, until the number of text nodes under the tree has changed (identified by `stateText` $\neq$ `state`). The algorithm now checks if the number of text contents has changed

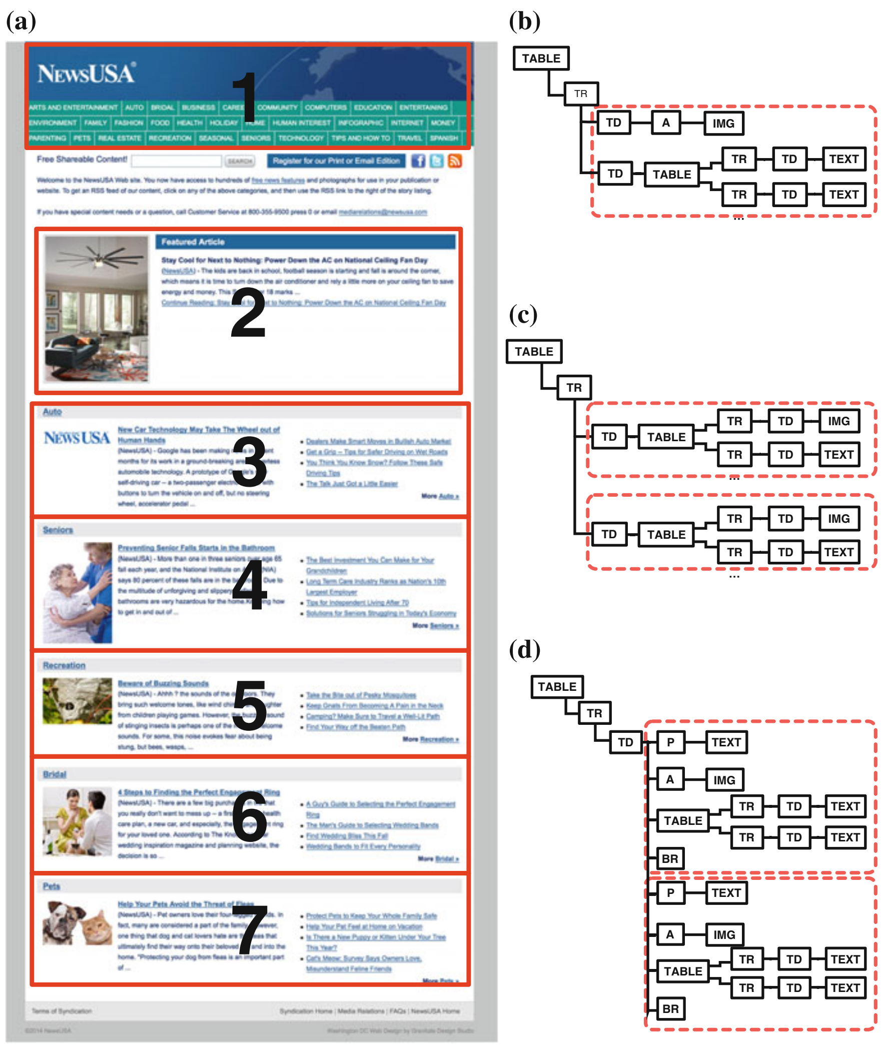


Fig. 10.6 Different classes of images in browser and DOM representation (Webpage extracted from http://newsusa.com). **a** Image segments 1–7. **b** DOM tree for segment 1(unlisted image). **c** DOM tree for segments 3–7 (listed image). **d** DOM tree for semi-listed images

twice. Since in the first run this is not the case, the current node is checked for typical semi-listed structure with repeating patterns of HTML tags. If such patterns are found, the image is a semi-listed image and the region of the image is extracted and returned as image context. If no semi-listed structure can be found, the algorithm continues to traverse the tree upwards until a parent node is found at which the number of nodes in the sub-tree changes again. In this case, the number of nodes has changed twice and the sub-tree is checked for sibling nodes with similar tree structure. If such

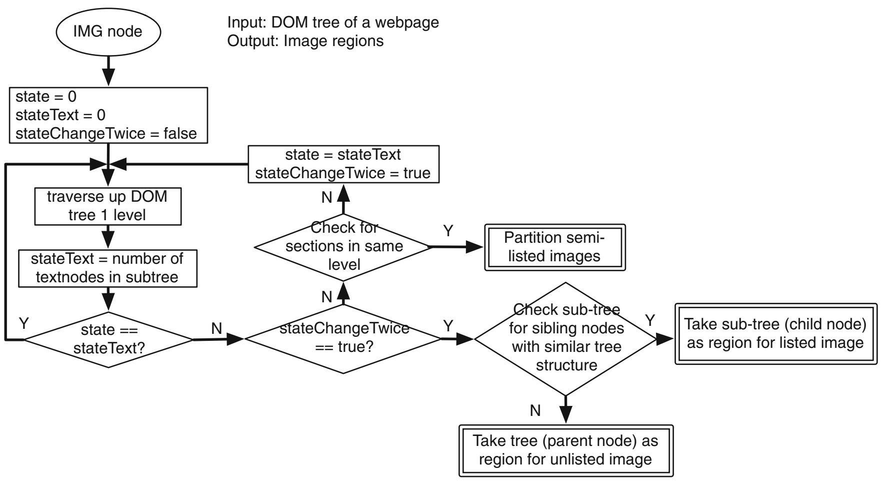

IMG node
Input: DOM tree of a webpage
Output: Image regions
state = 0
stateText = 0
stateChangeTwice = false
state = stateText
stateChangeTwice = true
traverse up DOM tree 1 level
stateText = number of textnodes in subtree
state == stateText?
Y
N
Check for sections in same level
Partition semi-listed images
stateChangeTwice == true?
Check sub-tree for sibling nodes with similar tree structure
Take sub-tree (child node) as region for listed image
Take tree (parent node) as region for unlisted image

Fig. 10.7 Flowchart diagram to Monash extractor (based on [11])

sibling nodes are found, the current image is a listed image and the sub-tree which belongs to the image is returned as image region. Otherwise, the image is an unlisted image, and the complete tree under the actual node is returned as image region. The algorithm is specified in the following flow-chart (see Fig. 10.7) adapted from [11].

The time complexity is the same as that for the siblings extractor, since DOM traversion and text node collection are both contained in the Monash Extractor.

10.4.7 Page-Segmentation-Based Context Extraction

The nature of Web documents being overloaded with multiple contents demands for methods to partition such documents into particular segments which would allow processing each segment more precisely. There are different approaches to page segmentation in the literature, but it is out of scope to present them here. However, one of these algorithms, namely the Vision-based Page Segmentation (VIPS) [22], has been applied as a preprocessing step for WICE and thus will be described here in detail.

The VIPS algorithm [22] is an hierarchical top-down approach, which starts with the whole page as the initial block. For each block, a Degree of Coherence (DoC) is computed using heuristic rules based on the DOM Tree structure and visual cues obtained from the browser representation. To get an idea of what the rules look like, we present one of the thirteen rules applied in [22]:

```
IF (currentNode.bgcolor != childNode.bgcolor)
  THEN
    divide currentNode;
    childNode.setDoC(childNode.TAG,  childNode.size);
END iF;
```

This rule checks whether the current node has the same background color as one of its child node. If it is not the case, the current node is divided and at the same time, the child node with different background color will not be divided in this round. Finally, the DoC value for the child node is set to a value between 6 and 8 based on the HTML tag and the size of the child node.

The algorithm applies different rules on determined HTML tags. For example, the presented rule is only applied on TABLE-tag and TR tag. However, there are some general rules that are applied on every tag.

The assigned DoC value determines how much the contents within a block correlate to each other. It ranges from 1 to 10 while 10 represents the highest correlation. At the beginning a Permitted Degree of Coherence (PDoC) value is specified, which controls the segmentation granularity. If a particular block has a DoC value smaller than PDoC, this block has to be subdivided recursively until all blocks on the bottom fulfill the condition.

Although the VIPS algorithm was not primarily designed to extract the context of a Web image, we can use the segmented block structure to assign the text of a block to an image within this block, as used by Cai et al. in [13]. He et al. [10] estimated a PDoC of 5 as a best suited value. We have extracted the image context for a PDoC value of 5, 6 and 7, since in our runs, the PDoC value of 5 produced too coarse segments.

Since the VIPS library exists only for the Windows operating system, we run a batch job over our Web site collections and stored the results in XML files for different PDoC values. In a further processing step, the image context was extracted from the XML files.

10.4.8 Full Plain-Text

One of the simplest methods to context detection is to associate the complete text of a Web document with the images within this document. At first glance, this approach seems to be defective since in multi-topic documents every image will get the same context resulting in reduction of the precision. Nevertheless, this method has been applied to image indexing for Web image retrieval [23]. We have implemented this baseline in our system to show what accuracy is reachable without any selection and also to compare and highlight the benefits of particular WICE methods.

10.4.9 Performance Measures

In order to compare the extracted image context with the ground truth objectively, appropriate performance measures are needed that will reward any congruence and penalize any divergence between the two sets.

Even among human experts there might be slightly different opinions on where exactly the borders of image context are, therefore testing on exact matches between extracted and ground truth context poses a criterion that is too strong. Instead, a partial accordance between the expert judgments and the output of a context extracting algorithm should be considered.

Another research field where the partial accordance is considered in the evaluation task is Information Retrieval (IR). The scenario in IR is usually as follows: the user has a collection of documents and he or she wants to know which documents in the collection are related to a predefined query; the task is processed by a human expert who defines the ground truth, and by an automatic search method which is to be tested. Finally, the accordance of the two sets is measured by *Precision* and *Recall*. In IR, the concept of precision P is defined as the ratio of the relevant retrieved document to all retrieved documents, and the concept of recall R is the ratio of the relevant retrieved documents to all relevant documents. Since these measures complement each other, the harmonic mean of both comprised in the F-score provides a suitable performance measure to compare the retrieved objects to the relevant objects. It is defined as follows:

$$F\text{score} = 2 \cdot \frac{P \cdot R}{P + R}$$

The usage of the mentioned IR measures in the context extraction scenario requires a specification of what the retrieved and relevant objects are. As the context is written in natural language, we can simply use the individual words of the document as these objects. The context can then be represented as a sequence of words. The computation of the intersection for two word sequences, as required to compute precision and recall, is accomplished by the *longest common subsequence* (LCS) of the two sequences.

In order to analyze the stability of the WICE methods within a specific document collection C, we compute the *standard deviation* of the F-score within a collection C, which is defined as

$$\sigma(C) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(F_i - \mu)^2}$$

where F_i is the F-score for a document $D_i \in C$ and μ is the average F-score over the complete collection C. Further N is the number of documents in C.

10.4.10 Framework Output

The final output of the evaluation framework are the different evaluation results that were computed for different combinations of WICE methods, evaluation metrics and ground truth collections. For example, one data block in the output could be the recall value of the extraction done with the N-terms extraction method, where the CNN ground truth collection was used.

Here, the output is generated as a csv-file which allows us to use the data as input in a visualizing application that creates appropriate diagrams in order to better present the results to a user. Furthermore, the output data can be passed to other applications where other useful computations can be performed (e.g., the computation of the standard deviation).

10.5 Evaluation Results

The results of the performed evaluation are summarized into the diagrams in Figs. 10.8 and 10.9. The diagrams show the precision, recall, F-score and the standard deviation of the F-score for every included WICE method. The precision and the recall values were admitted in the diagrams to reason the F-score. The title of each diagram refers to one of our test collections.

Figure 10.8 shows the results for the test collections created with simpler extraction rules. By simple extraction rule, we mean a rule of the format:

```
IF ImgTag has ParentNode with Attribute a ="b"
  THEN
    extractAllTextNodes under ParentNode as ImageContext;
    exit;
```

On the other hand, a complex rule is more tailored and looks like:

```
IF ImgTag has ParentNode with Attribute a = "b"
  THEN
    IF ParentNode has ChildNode with Attribute c = "d"
      THEN
        addAllTextNodes under ChildNode to ImageContext;
    IF ParentNode has ChildNode with Attribute e = "f"
      THEN
        addAllTextNodes under ChildNode to ImageContext;
    exit;
```

The Monash and the Siblings method perform best with an F-score value in the range of 85–100 %. The standard deviation of F-score for the DOM-based methods

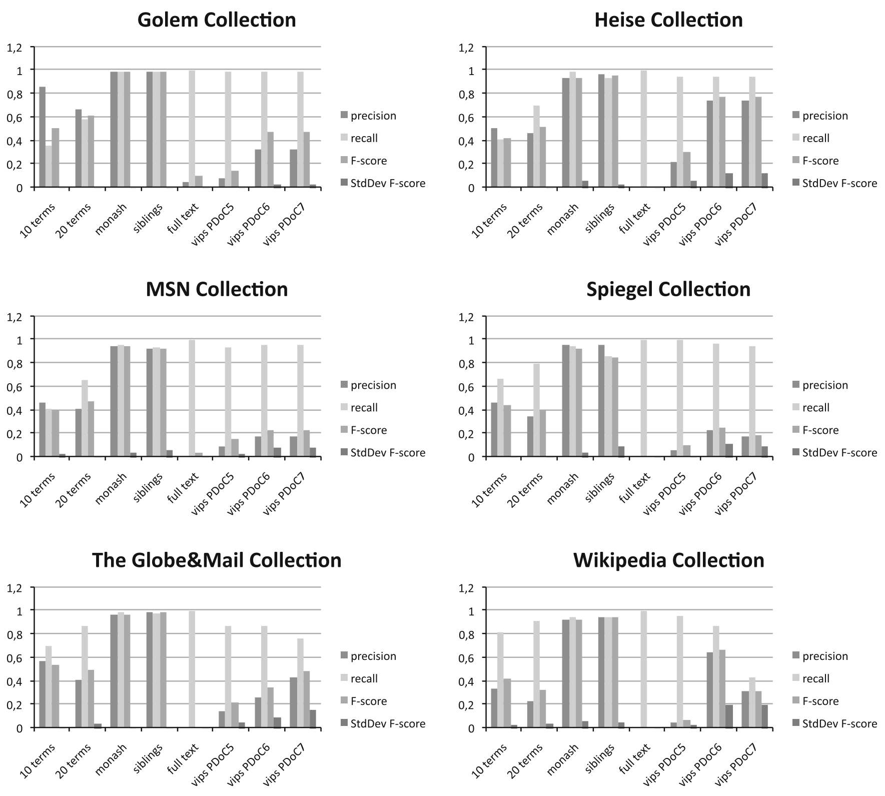


Fig. 10.8 Evaluation results for test collections with simpler extraction rules

is relatively small and hence we can argue that the reliability of these algorithm is high.

The N-Term Extractor has a lower accuracy, since its F-score ranges within the middle third. The precision value indicates that about half of the extracted text does not belong to the image context. This can be explained by analyzing a small portion of the test documents, considering that the image context is mostly placed either before the image or after it. The N-Term methods choose the N words before and after the image, therefore half of them is falsely selected. The recall values vary from 30 to almost 86 % depending on the collection. Of course, the recall for 20-term method is always higher than for the 10-term method. On the other hand, it is obvious that the 10-term method has a higher precision. A significant advantage of the 20-term method over the 10-term method could not be detected, as there is no winner according to the F-scores. It is interesting, that the standard deviation of F-score for the N-Term methods is very small indicating the similar structure for all the documents within a test collection.

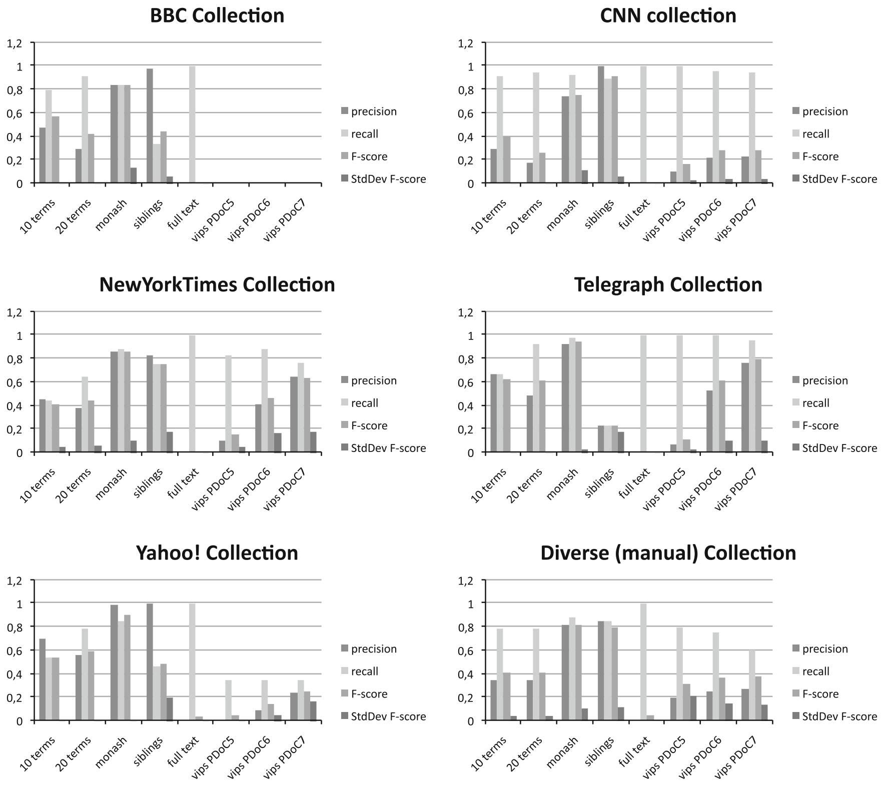


Fig. 10.9 Evaluation results for test collections with more complex extraction rules

The VIPS algorithm almost always has an excellent recall value but its precision is relatively small which results in small F-score values. This indicates, that the visual segments obtained by the VIPS algorithm envelop too broad content. We expected that greater PDoC value would yield a higher precision which, however, surprisingly was not the case. A PDoC value of 6 seems to perform best for almost all collections. The best results for VIPS were evaluated on the *Heise* and the *Wikipedia* collection. But the standard deviation of the F-score in these collections has the highest value, meaning that the extraction accuracy has a high variance and thus the reliability of the method is low. Compared to the N-Term method, the VIPS algorithm is much more complex but does not achieve a significant improvement.

The baseline algorithm extracting the full text of the document achieved always a recall of 100 % (which was expected), but since the precision value is close to zero the overall F-score is also near zero. This fact implies the necessity for applicable WICE methods.

In Fig. 10.9, the diagrams with the evaluation results for the collections which need more complex extraction rules are shown. Also, the results for the *diverse* collection of manually extracted context pairs is included in this figure.

To start with, for the VIPS based extractors, we remember these are based on the browser Internet Explorer. Because the collected BBC pages could not be displayed by this browser due to java scripting failures, VIPS was not able to deliver any context and thus the values are missing. This is a very important disadvantage of VIPS that relies on a specific browser application. On the other collections, VIPS performed best with high PDoC values. A higher PDoC value than 7 was not suitable, since then the visual block solely contained the image. The F-score ranges from fair for the *telegraph* collection to poor for the *Yahoo!* collection. As already discussed for the first group of evaluation results, VIPS has a high F-score standard deviation which signals that its performance highly varies from image to image.

The N-Term extractors yield very similar performance values as in the first evaluation group: the performance does not depend much on N, the F-score ranges in the middle third, and the standard deviation of F-score is low, indicating that the extraction performance is highly reliable and repeatable.

The Siblings extractor could not solidly achieve high performance values as in the first evaluation group, due to the more complex template structures of the *telegraph* and the *Yahoo!* website. For these collections, the other algorithms performed better. But surprisingly, the Siblings extractor produces a very high F-score in the *diverse* collection, which argues that most of the Web pages available in the internet contain a simple-structured HTML code.

The Monash extractor performed best for almost all collections. Nevertheless, when the template structure gets more complex, its reliability and repeatability gets lower, as indicated in higher F-score standard deviation.

10.6 Conclusion and Future Work

In this work, an evaluation framework for image context extraction was proposed and tested with different common extraction methods.

A main result from the evaluation experiments is that the WICE methods based on DOM achieved the best performance (almost always an F-score over 90), indicating that many HTML documents gathered from popular domains are well structured (with regard to their DOMs). As a second result—to our surprise—VIPS, one of the mostly applied methods for image context extraction in the literature, performed poorly on the test data, since the page blocks computed by VIPS were too broad.

The obtained results can help scientists on deciding which method to use in their applications when image context is needed.

Further, the framework is easily extendable with other metrics and other extraction methods. Therefore, this system can also be used to evaluate and compare the performance of new extracting methods, which will be the main scope of our future work.

References

1. Gudivada VN, Raghavan VV (1995) Content-based image retrieval systems. Computer 28: 18–22
2. Alcic S, Conrad S (2010) Measuring performance of web image context extraction. In: Proceedings of the 10th international workshop on multimedia data mining, MDMKDD'10, ACM, New York, pp 8:1–8:8
3. Alcic S, (2011) Web image context extraction: methods and evaluation. PhD thesis, Heinrich-Heine-University of Duesseldorf
4. Alcic S, Conrad S (2011) Page segmentation by web content clustering. In: International conference on web intelligence, mining and semantics (WIMS11), May 2011
5. Coelho TAS, Calado PP, Souza LV, Ribeiro-Neto B, Muntz R (2004) Image retrieval using multiple evidence ranking. IEEE Trans Knowl Data Eng 16(4):408–417
6. Sclaroff S, Taycher L, Cascia ML (1997) Imagerover: a content-based image browser for the world wide web. In: Proceedings of the 1997 workshop on content-based access of image and video libraries (CBAIVL'97), CAIVL'97, IEEE Computer Society, Washington
7. Vasconcelos N, Lippman A (2000) Bayesian relevance feedback for content-based image retrieval. In: Proceedings of the IEEE workshop on content-based access of image and video libraries (CBAIVL'00), IEEE Computer Society, Washington, p 63
8. Yong-hong T, Tie-jun H, Wen G (2005) Exploiting multi-context analysis in semantic image classification. J Zhejiang Univ Sci, 1268–1283
9. Cai D, Yu S, Wen J-R, Ma W-Y (2003) VIPS: a vision-based page segmentation algorithm. Technical report, Microsoft Research (MSR-TR-2003-79)
10. He X, Cai D, Wen J-R, Ma W-Y, Zhang H-J (2007) Clustering and searching WWW images using link and page layout analysis. ACM Trans Multimed Comput Commun Appl 3(2):10
11. Fauzi F, Hong J-L, Belkhatir M (2009) Webpage segmentation for extracting images and their surrounding contextual information. In: ACM multimedia, pp 649–652
12. Alexa (2011) The web information company. http://www.alexa.com
13. Cai D, He X, Li Z, Ma W-Y, Wen J-R (2004) Hierarchical clustering of WWW image search results using visual, textual and link information. In: Proceedings of the 12th annual ACM international conference on multimedia, MULTIMEDIA'04, New York, pp 952–959
14. Sandor A, Tripp A, Giustina F, Peskin GL, Lempinen S, Gold R, Sanders J,Yount S (2011) The homepage of the JTidy java API. http://jtidy.sourceforge.net/
15. Feng H, Shi R, Chua T-S (2004) A bootstrapping framework for annotating and retrieving WWW images. In: Proceedings of the 12th annual ACM international conference on multimedia, MULTIMEDIA'04, ACM, New York, pp 960–967
16. Trifonova G (2010) Implementation of a tool for manual web image to context mapping. Bachelor Thesis, September 2010
17. Sclaroff S, Cascia ML, Sethi S (1999) Unifying textual and visual cues for content-based image retrieval on the World Wide Web. Comput Vis Image Underst 75(1–2):86–98
18. Zhigang H, Xiang-Jun W, Qingshan L, Hanqing L (2005) Semantic knowledge extraction and annotation for web images. In: Proceedings of the 13th annual ACM international conference on multimedia, MULTIMEDIA'05, ACM, New York, pp 467–470
19. Frankel C, Swain MJ, Athitsos V (1996) WebSeer: an image search engine for the world wide web. Technical Report, University of Chicago, Chicago
20. Shen HT, Ooi BC, Tan K-L (2000) Giving meanings to WWW images. In: Proceedings of the eighth ACM international conference on multimedia, MULTIMEDIA'00, ACM, New York, pp 39–47
21. Liu B (2007) Web data mining: exploring hyperlinks, contents, and usage data. Data-centric systems and applications. Springer, Berlin

22. Cai D (2011) Download site of the DEMO of VIPS algorithm. http://www.zjucadcg.cn/dengcai/VIPS/VIPS.html
23. Ortega-Binderberger M, Mehrotra S, Chakrabarti K, Porkaew K (2000) WebMARS: a multimedia search engine for the world wide web. In: Proceedings of the SPIE electronic imaging 2000: internet imaging, San Jose

Chapter 11
Content Based Image Search for Clothing Recommendations in E-Commerce

Haoran Wang, Zhengzhong Zhou, Changcheng Xiao and Liqing Zhang

Abstract A number of algorithms exist in measuring clothing similarity for clothing recommendations in E-commerce. The clothing similarity mostly depends on its shape, texture and style. In this paper we introduce three models of defining feature space for clothing recommendations. The sketch-based image search mainly focuses on defining similarity of clothing in contour dimension. The spatial bag-of-feature approach is employed to measure the clothing similarity of local image patterns. Finally, we introduce a query adaptive shape model which combines shape characteristics and labels of clothing, in order to take the semantic information of clothing. A large number of simulations are given to show the feasibility and performance of the clothing recommendations by using content-based image search.

11.1 Introduction

Content Based Image Retrieval (CBIR) is to retrieve images from image database, by using features derived from images. The term “content” of an image refers to colors, shapes, textures or any other features that can be derived from the image itself. CBIR has a large number of potential applications, such as clothing recommendations in internet, picture drawing learning and image copyright verification. Almost every big E-commerce company like eBay and Amazon supports the keywords based recommendations, such as by typing “red, round collar t-shirt”. Taobao, a Chinese

H. Wang · Z. Zhou · C. Xiao · L. Zhang (✉)
Brain-Like Computing and Machine Intelligence Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800, Dongchuan Road, Shanghai 200240, China
e-mail: zhang-lq@cs.sjtu.edu.cn

H. Wang
e-mail: hare_ran@sjtu.edu.cn

Z. Zhou
e-mail: tczhouzz@sjtu.edu.cn

C. Xiao
e-mail: david@sjtu.edu.cn

© Springer International Publishing Switzerland 2015
A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_11

E-commerce platform provider, has recently released a new service of bags/shoes recommendations based on shape similarities. In this paper we investigate clothing similarity problem and develop a number of rules to measure the similarities for clothing recommendations.

There're quite lot of studies on clothing similarities. Liu et al. [11] proposed a system for automatic occasion-oriented clothing recommendations. Occasion-oriented means given a user-input occasion (e.g., wedding, shopping or dating), the system suggests the most suitable clothing from the user's own clothing photo album. Liu utilized middle-level clothing attributes (e.g., clothing category, pattern) as latent variables in latent Support Vector Machine (SVM) for recommendations. Wang et al. [19] proposed a re-ranking system for apparel recommendations. First, the system extracted color features and retrieved by bag-of-visual features (BOW). Then re-ranking approach is fulfilled by exploiting clothes attributes. Di et al. [8] also used attribute learning to recommend [2]. Chen et al. [5] proposed a system that is capable of generating a list of nameable attributes for clothes on human body in unconstrained images. Chen extracted low-level features in a pose-adaptive manner, and combine complementary features for learning attribute classifiers.

On cross-scenario clothing recommendation, Liu et al. [12] used transfer learning method to deal with the big discrepancy between snapshot in the street and clothing image in internet shops. The extracted features for learning include HOG, LBP, Color moment, Color histogram and skin descriptor. Besides, Fu et al. [9] utilized semantic-preserving visual phrases formed by vocabulary tree [15].

Other works including Yang et al. [20] which leveraged both face detection, tracking and clothing segmentation for clothing recognition in surveillance videos. Wang et al. [18] studied a multi-person clothing segmentation algorithm blocking models for highly occluded images. Two clothing recommendation systems have been proposed by Liu et al. and Shen et al. [11, 16].

Image similarity depends on both similarities in visual feature and image semantics. As shown in Fig. 11.1, even the same-class clothing varies significantly in contour and visual features, which is called intra-class variation. Besides, there also

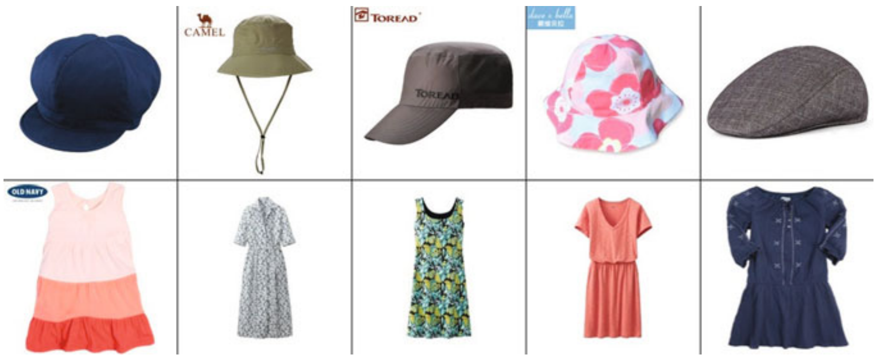


Fig. 11.1 Examples of intra-class variation. Same objects in a line present quite different contour

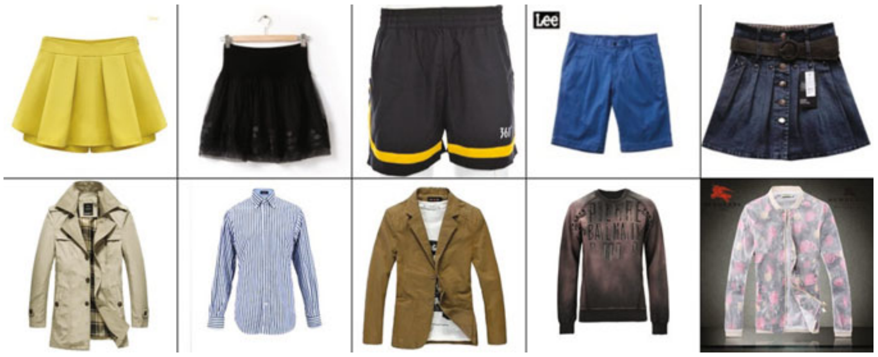


Fig. 11.2 Examples of inter-class ambiguity. The shapes of different objects in a line might have similar shapes

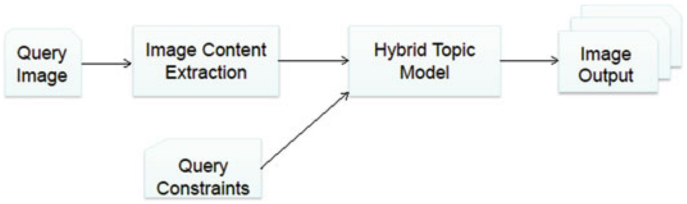


Fig. 11.3 Framework of image content based clothing recommendations. We use contour, local feature for image content extraction and textual feature for query constraint respectively

exists many different types of clothing with equivocal contours, called inter-class ambiguity, as shown in Fig. 11.2.

To address the intra-class variation and inter-class ambiguity, we introduce three different methods of image similarities for clothing recommendations shown in Fig. 11.3. First, a Histograms of Oriented Gradients (HOG), proposed in Dalal and Triggs [7], is developed to measure the similarity in image contour. Then, a Bag of Features (BOF) approach [4] is proposed to measure the image similarity in sophisticated local features, and a hybrid topic model [17] is achieved by both visual features and textual information. In this article, contour feature presents the contour of clothing, precision local feature accounts the local detail of clothing (e.g. position of button, shape of neckline). Textual information means the words that describes the clothing (e.g. color, pattern, material).

Zhou et al. [21] proposed a shift-invariance and size-invariance Hierarchical Orientation Feature (HOF) for large scale image retrieval, see Sect. 11.2. The main idea for dealing with object size in the image is to adopt the selective attention model. The human visual system will pay attention to the most informative region in the scene, instead of the whole image. This mechanism in the visual system is known as the selective attention. The selective attention model comes from the fact that when human beings see an image, they usually look through the whole image for a short

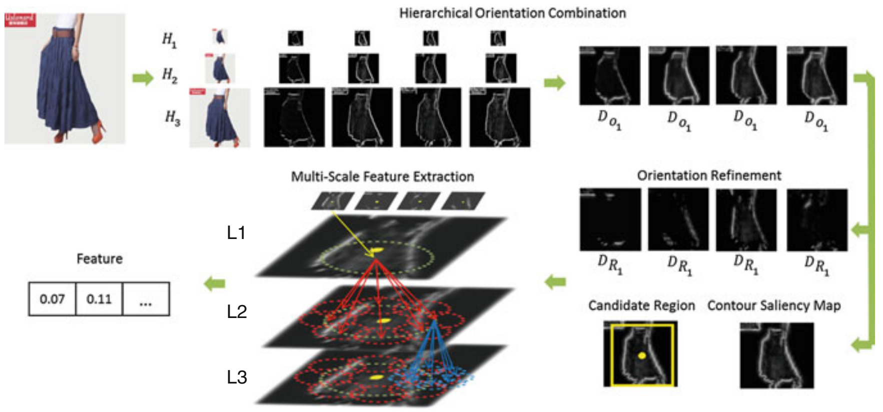


Fig. 11.4 Feature extraction consists of these steps: (I) Hierarchical orientation combination. (II) Orientation refinement. (III) Locate candidate region. (IV) Multi-scale feature extraction

while and then focus their eyes on the salient place of the image. To adopt such a mechanism for image retrieval, we don't extract features on the the whole image. Instead, firstly we find the region of interest from the saliency map which is usually a part of the image with an interesting object, and then extract features on that region for finding similar objects from image dataset.

To speed up image search, indexing the feature vectors of images is necessary. We introduce a hierarchical inverted index algorithm and split the next process into coarse-to-fine similarity measure. The whole process will filter out a large number of irrelevant images as early as possible, and make it possible to perform the real-time retrieval. The whole retrieval procedure is illustrated in Fig. 11.4.

Besides contour characteristic, clothing similarity in many cases depends on sophisticate local features. To explore such local features, Cao et al. [4] proposed a new class of bag-of-features [10] to encode local geometric features of objects, see Sect. 11.3. Beyond existing orderless bag-of-features, local features of an image are first projected to different directions or points to generate a series of ordered bag-of-features, based on which different families of spatial bag-of-features are designed to capture the invariance of object translation, rotation, and scaling. Then the most representative features are selected based on a boosting-like method to generate a new bag-of-features-like vector representation of an image. The retrieval framework works well in image retrieval task owing to the following three properties: (1) the encoding of geometric information of objects for capturing objects with spatial transformation, (2) the supervised feature selection and combination strategy for enhancing the discriminative power, and (3) the representation of bag-of-features for effective image matching and indexing for large scale image retrieval.

To further explore semantical features and combine them with contour features, Sun et al. [17] developed a hybrid topic model to retrieve images with both contour similarity and semantic similarity, see Sect. 11.4. Shape Topic Model provided a general sketch recognition system to recognize any semantically meaningful objects. Sketch means a hand-draw draft. To increase the recognition coverage, Sun et al. [17] utilized a web-scale clip art image collection as the knowledge base of the recognition system.

To alleviate the problems of intra-class shape variation and inter-class shape ambiguity in this unconstrained situation, a query-adaptive shape topic model is proposed to mine object topics and shape topics related to the sketch, in which, multiple layers of information such as sketch, object, shape, image, and semantic labels are modeled in a generative process [6]. Besides sketch recognition, the proposed topic model can also be used for related applications like sketch tagging, image tagging and sketch-based image search.

11.2 Hierarchical Orientation Feature

Different from existing algorithms, the features used in the retrieval system contain not only local information, but also global information of the object. By taking advantage of this characteristic, we could build a hierarchical index structure to accelerate retrieving images from the large scale database.

11.2.1 Feature Extraction

The framework of feature extraction is showed in Fig. 11.4. By leveraging hierarchical information with multi-scale feature extraction, it could obtain both position and global-to-local feature of the salient object. Global-to-local means feature contains both global and local information of object, the usage of global-to-local feature will be discussed in Sect. 11.2.2.

We calculate the hierarchical difference image D_{O_j} and the contour saliency map S from an image from:

$$S = \sum_{j=1}^{N} D_{O_j} = \sum_{j=1}^{N} \left(\sum_{i=1}^{M} [\max_k \{D_{H_i O_j}\}]_{m \times n} \right) \tag{11.1}$$

where $D_{H_i O_j}$ is the difference image of the ith level and the jth orientation as shown in Fig. 11.4. The size of $D_{H_{i-1} O_j}$ is lower than the size of $D_{H_i O_j}$ by two times. k is the color index, taking values of red, green and blue. $\max_k\{\cdot\}$ is the maximum of difference image among the three channels. $[\cdot]$m $\times$ n means scaling the difference image proportionably to the maximum size m $\times$ n. S is the contour saliency map of an image and is normalized to between 0 and 1. In Fig. 11.4, M = 3, N = 4, O_j denotes 0, $\pi/4$, $\pi/2$ and $3\pi/4$ orientation respectively. Because the minimum resolution of images for human beings is 32 $\times$ 32, we set the maximum size of $D_{H_1 O_j}$ is 32 $\times$ 32, and then m $\times$ n is 128 $\times$ 128.

$$S_x = \arg\max_x\{\sum(\lfloor S\rfloor_{T_s})_x \star g_x\},\ S_y = \arg\max_y\{\sum(\lfloor S\rfloor_{T_s})_y \star g_y\} \tag{11.2}$$

where $\lfloor\cdot\rfloor T_s$ denotes the value greater than T_s, in our experiment, $T_s = 0.25$, which is determined by experiment. $\sum(\cdot)_x$ and $\sum(\cdot)_y$ are the sum along the axis x and the axis y respectively, g is the Gaussian kernel, and $\star$ denotes convolution. (S_x, S_y) denote the centroid of the object in the image. From Fig. 11.4 we can see, D_{Oj} cannot represent contour orientation information of the object clearly. Since 0 and $\pi/4$, $\pi/2$ and $3\pi/4$ are orthogonal respectively, we make the following operation:

$$\begin{aligned} C_{O_1} &= \lfloor D_{O_1} - D_{O_3}\rfloor_0, C_{O_3} = \lfloor D_{O_3} - D_{O_1}\rfloor_0 \\ C_{O_2} &= \lfloor D_{O_2} - D_{O_4}\rfloor_0, C_{O_4} = \lfloor D_{O_4} - D_{O_2}\rfloor_0 \end{aligned} \tag{11.3}$$

where C_{O_j} denotes clear orientation map. The final feature of an image is:

$$F_{L_pO_jt} = \sum C_{O_j}(x_{L_pt}, y_{L_pt}, r_{L_p}) \cdot G(r_{L_p}) \tag{11.4}$$

where $F_{L_pO_jt}$ denotes tth feature of the L_pth level and the O_jth orientation, and $G(r_{L_p})$) is the Gaussian kernel which radius is r_{L_p}, and C_{O_j}(x, y, r) is the region of the clear orientation map which centroid is (x, y) and the radius is r, and $r_{L_p} = 2r_{L_{p+1}}$, $r_{L_3} = 32$. When $p = 1, t \in \{1\}$, and when $p = 2, t \in \{1, 2, \ldots, 8\}$, and when $p = 3$, $t \in \{1, 2, \ldots, 64\}$. So the feature vector $F = F_{L_pO_jt}$ has $1\times4+8\times4+8\times8\times4 = 4 + 32 + 256 = 292$ dimensions. Finally, values of the feature are normalized to between 0 and 1. So the similarity measure of two images with feature vector F and F' is given by:

$$\text{Dist}(F, F^{'}) = \text{sim}\left(\{F_{L_pO_jt}\}, \{F^{'}_{L_pO_jt}\}\right) \tag{11.5}$$

where sim$(\cdot)$ could be any similarity measure, for example, Euclidean distance or cosine similarity.

Figure 11.5 is the histogram of average value of feature vector F from 100 thousand images. As shown in Fig. 11.4, region L_1 denotes the first 4 values of F, which are the most discriminative features in image similarity L_2 denotes the following 32 values of F, and region L_3 denotes the rest 256 values of F. From Fig. 11.5, we can observe that the component value of the features decreases as the component index increases. This observation indicates the feature vector F contains the global-to-local feature. Values from L_1 to L_3 denote the information which is from global to local respectively. Therefore, the component values in L_1 will dominate distance computing of Eq. (11.5). If objects in two images are very different in contour, the difference of the feature vectors must be large, and resulting in small similarity score in Eq. (11.5). But if two objects are only a little different, they should have almost the similar global information and are just different in local parts, resulting in high similarity score in Eq. (11.5).

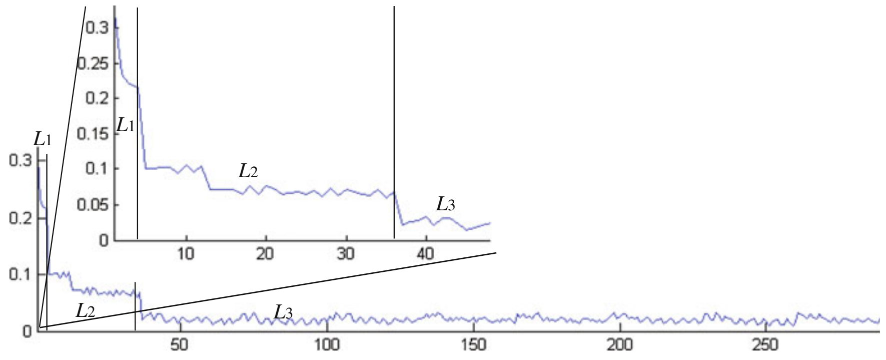


Fig. 11.5 Histogram of average feature values from 100,000 images. The first 4 values of F belong to region L_1, the components from 5 to 36 of F belong to region L_2, and the rest 256 values of F belong to region L_3

11.2.2 Index Structure

The feature vector contains an objects global-to-local information. For speeding up, at first we select only global values of F which is from both L_1 and L_2, which means, in this step, we ignore local L_3 values (see L_1, L_2, L_3 in Fig. 11.4). Specifically, global value takes exactly the first 36 values of F. And for each value, we separate it into some parts, and for each part, there is a corresponding inverted list of images, as shown in Fig. 11.6. With the index structure, we could select top $N_1 (\leq T)$ candidate results from the database quickly, and then, we select top N_2 results from L_1 candidate results with similarity measure of first 36 values of F. Finally, we rank the L_2 results

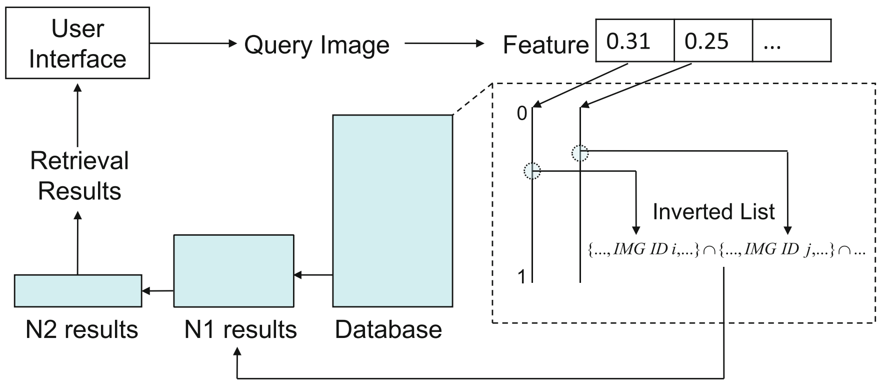


Fig. 11.6 Index process of database. For each query, N_1 results are first selected from 100 thousand images by inverted files, and then top N_2 results are selected from N_1 results with similarity measure of first 36 values of F, and final results are from N_2 results with similarity measure of all 292 values of F

with similarity measure of all values of F (global-to-local) and take them as the finally retrieval results. Thus we could build a hierarchical top-down retrieval structure. In our experiment, $T = 50{,}000$ $(N_1 \leq T)$, $N_2 = 2{,}000$.

11.3 Spatial-Bag-of-Features

Local features such as SIFT [13] and SURF [1] are widely used in image retrieval, which detect interest points from the images and use descriptors to describe them. To be convenient for retrieval, we use BOF technology to translate the local features extracted from images into one-dimensional vectors. The procedure is described as follows: First, we extract descriptors from all the images in database. Then k-means or other clustering methods is performed over these descriptors. Finally, we produce a “dictionary” of local features, in which each “word” is defined as the centers of the clusters and the size of “dictionary” is the number of clusters. For a given image, we map each descriptor detected from it to a “word” through the clustering process and the image can be represented by the histogram of the “word”s. We index these histogram features by inverted files [10] so as to support large-scale online image retrieval.

However, the accuracy of such an approach is not satisfactory. The main reason is that it ignores the spatial information of interest points in process procedure. To solve this problem, we propose a new class of features for large-scale image retrieval. It encodes spatial information as well as visual information of images. Besides, it has the same format as current bag-of-features, which guarantee that traditional index technologies can be applied to it.

Generally, scene images such as buildings or sea, have horizontal and vertical relationships among local features, while objects like sun and flowers have circle-like relationships. We aim to project local descriptors onto certain lines or circles in order to capture basic geometric information in images. We call the features generated by this way as “ordered bag-of-features”. Two families of ordered bag-of-features are designed in Cao et al. [4]: Linear ordered bag-of-features and Circular ordered bag-of-features.

Linear ordered bag-of-features projects the descriptors in the image plane onto a line with an arbitrary angle θ. The locality of each feature is transformed to a one-dimensional coordinate along the line. We divide this line into L equal segments and treat them as bins. The descriptors inside each bin are represented by a histogram statistics. Therefore the feature of an image can be described as L connected histograms. We enumerate the values of θ and L, and determine the best ones by machine learning technology.

Circular ordered bag-of-features divides the image plane into L sectors with the same radian after locating the center (x, y). This time we treat each sector as a bin and use a histogram to represent the descriptors in it. Similar to Linear ordered bag-of-features, we can use L connected histograms to describe the feature of an image.

We use Θ to represent the parameters of linear projection $\{L, \theta\}$ and circular projection $\{L, (x, y)\}$. The feature generated by either a linear or a circular projection is denoted by:

$$H^{\Theta} = [h^{1,\Theta}, h^{2,\Theta}, \ldots h^{L,\Theta}]. \tag{11.6}$$

where $h^{i,\Theta}$ is the histogram in the ith bin of the projection parameterized by Θ. The similarity between image P and image Q is defined as:

$$\langle H_P^{\Theta}, H_Q^{\Theta} \rangle = \sum\nolimits_{i=1}^{L} \text{Sim}(h_P^{i,\Theta}, h_Q^{i,\Theta}) \tag{11.7}$$

where $\text{Sim}(\cdot, \cdot)$ could be any histogram similarity measure.

In addition, we propose three variant features of ordered bag-of-features which could handle typical transformations of objects such as translation, rotation, and scaling. We call them as “spatial bag-of-features”. Technologies like histogram calibration and histogram equalization are applied in the extraction procedure, which is further discussed in his paper.

A series of spatial bag-of-features could be obtained by enumerating parameter θ, (x, y) and L. We treat each histogram $h^{i,\Phi}$ in them as a “spatial bag-of-word”. $\Phi = \{\Theta, k(k \leq M)\}$, in which M is the number of bins of the projection (with or without encoding invariance) parameterized by Θ, and k is the index of the bin. The final feature we applied is defined as the linear combination of these “word”s. The similarity between image P and Q is given by:

$$\langle H_P^{\Phi}, H_Q^{\Phi} \rangle = \sum\nolimits_{i \in S} a^{\Phi} \text{Sim}(h_P^{i,\Phi}, h_Q^{i,\Phi}) \tag{11.8}$$

where S represents a set of best “spatial bag-of-words”. We can use RankBoost algorithm to learn S and coefficient a^{Φ}s.

To verify the effectiveness of “spatial bag-of-features”, we conduct a series of experiments on the collected database in Experiment section. The results show that the proposed approach does capture the local precision features of the clothing.

11.4 Query-Adaptive Shape Topic Model

Most existing Content-based Image Retrieval systems find similar images in low level features, while Text-based Image Retrieval finds images with relevant tags regardless of content in the images. Generally, people are more interested in finding similar images both in contours and high-level concepts. Same type of objects usually have different shapes as shown in Fig. 11.1, while the same shape may belong to different objects shown in Fig. 11.2. This results in large variance in intra-class shape, and also uncertainty in inter-class shapes. Therefore, using sketch as query image is not sufficient to express user’s expectation for retrieved images. We need to explore both the shapes and descriptive features in retrieving images via sketch.

To explore both semantics and shape information in images, we introduce a hybrid topic model to discover semantic topics possibly representing the sketch. In this section, we introduce a probabilistic topic model for hand-drawn sketch recognition, i.e. Query-adaptive Shape Topic (QST) model [17], to simulate the generative process of an image and its textual information.

We first analyze and discover the generative process for the observed information and latent topics. Since there are only simple clip art images in the collection, it is natural to assume there is only one object topic in each image. To overcome the challenge of shape ambiguity, besides the visual feature, the textual information of images is also leveraged in the model. Therefore, for each image, its object topic further generates both visual information like shapes and semantic information like the surrounding text of the image. Since one object might correspond to different shapes, it is natural to add another layer of topics, i.e. shape topics, to represent variant types of shapes related to the sketch. This can further alleviate the problems of shape variation and shape ambiguity. The shape feature is then generated according to the shape topic. Instead of a purely unsupervised topic mining problem, in this work it is necessary to discover object topics that are more relevant to the sketch query. Different from sLDA [14], in which the supervised information (a response variable) is generated by the latent topics, in QST we use the sketch query to influence the possibility of generating a shape feature from a shape topic. This guarantees that the discovered shape topics are more relevant to the sketch query, and so are the object topics.

It should be noted that, to make the model more general, we also enable users to optionally add keywords when they draw sketches. Thus, two factors, i.e. the sketch and the keywords, will supervise the generative process, in which the keywords could be an empty set Ø.

We first introduce some notations and definitions. Assume there are N images $\{I_1, I_2, \dots, I_N\}$ in the collection, and the words in the dictionary are $\{w_1, w_2, \dots, w_M\}$. The shape feature of I_n is represented by r_n and the noisy labels are represented by $\{w_1, w_2, \dots, w_T\}$. $\delta(w_m, I_n) = 1$ if $w_m \in \{w_1, w_2, \dots, w_T\}$; and 0, otherwise. Let z denote a latent variable to represent an object topic, with discrete values $z = 1, \dots, K$, and s denote a latent variable to represent a shape topic, with discrete values $s = 1, \dots, N_s$. We abbreviate "sketch" and "keywords" to "ske" and "key" for long equations. Given N, M, K and N_s, the generative process of QST model is given as follow:

1. For each image I_n, sample the object topic: $z \sim p(z|I_n)$
2. For each object topic z:

   (a) sample T words $\{w_1, w_2, \dots, w_T\}$, in which for each word w_m we have

$$w_m \sim p(w_m|z, \beta, \text{keywords}) = \beta_{z,w_m}^{\delta(w_m,\text{keywords})} \tag{11.9}$$

   (b) Sample the shape topic

$$s \sim p(s, z, \theta) = \theta_{z,s}$$

(c) For each shape topic s sample the shape feature:

$$\begin{aligned} r_n &\sim p(r_n|s, \mu, \sigma, \beta, \text{sketch}) \\ &= \frac{1}{\sqrt{2\pi}\sigma_s} \exp\left\{-\frac{\text{dist}(r_n, \mu_s)^2}{2\sigma_s^2} \text{dist}(r_n|\text{sketch})\right\} \end{aligned} \tag{11.10}$$

in which $\text{dist}(r_n, \mu_s)$ is defined as the distance between r_n and μ_s, and $\text{dist}(r_n, \text{sketch})$ is the distance between r_n and the sketch. In particular we define $\delta(w, \emptyset) = 1$.

Given the parameters $\theta, \mu, \sigma, \beta$, we have the following joint probability of a set of N objects topics z, a set of N shape class s and a set of N shape feature r and words w:

$$\begin{aligned} &p(I, w, r, z, s|\theta, \mu, \sigma, \beta, \text{sketch}, \text{keywords}) \\ &= \prod_{n=1}^{N} \{p(I_n)p(z, I_n)p(r_n|s, \mu, \sigma, \beta, \text{sketch})p(s|z, \theta) \\ &\prod_{m=1}^{M} (p(w_m|z, \beta, \text{keywords}))^{\delta(I_n, w_m)}\Bigg\} \end{aligned} \tag{11.11}$$

QST contains different levels of parameters, such as the parameters of corpus and parameters of shape features. These parameters can be learned from hybrid training image dataset. The detailed training procedure is referred to Zhou et al. [21].

11.5 Experiment

In this section, we introduce image datasets for evaluating the performance of three different image retrieval methods, i.e. HOF in Sect. 11.2, spatial bag-of-features (SBOF) in Sect. 11.3 and QST in Sect. 11.4.

11.5.1 Image Dataset

There are several proposed datasets, however, none of them is suitable for training the model of QST. Yang et al. [20] collected the dataset with 200×300 pixels ,which is not appropriate for local feature detection. The dataset proposed by Bourdev et al. [3] contains attributes but no textual information. Hence we collect a new dataset with 100 thousand clothing images mainly from Taobao, an E-commerce platform provider in China. For obtaining textual information of each image in the dataset, we collect the surrounding texts of image. For each clothing image, we select the top

20 most frequent words (e.g., color, style, material) to describe the clothing image which appear in the same web page.

There are over 30 categories (e.g., skirt, longuette) of clothing in our dataset. These categories are defined in Liu et al. [12]. For each category, we collect the images and relevant textual information from top results from online shopping website by search engines. We didn't collect the nonstandard images with messy backgrounds because QST requires images with single piece object and clean background.

We run experiments on the server with 2 Intel Xeon 2.66 GHz Six-Core processors and 64 GB memory. The average retrieval time is between 1∼2 s.

11.5.2 Experimental Results and Analysis

We measure the precision-recall rate on category (class). For each class, we sample 100 images of same category, measure the average rate and compare the performance of HOF, SBOF and QST. The precision of top k retrieved clothings with respect to a query clothing q is calculated by:

$$\text{Precision@}k = \frac{\sum_1^k \text{Rel}(i)}{k} \tag{11.12}$$

where $\text{Rel}(i) = 1$ if ith result belongs to the same category as q, $\text{Rel}(i) = 0$ otherwise.

As shown in Fig. 11.7, QST outperforms the low-level feature methods, both HOF and SBOF, in average rate among all categories. Besides, for classes such

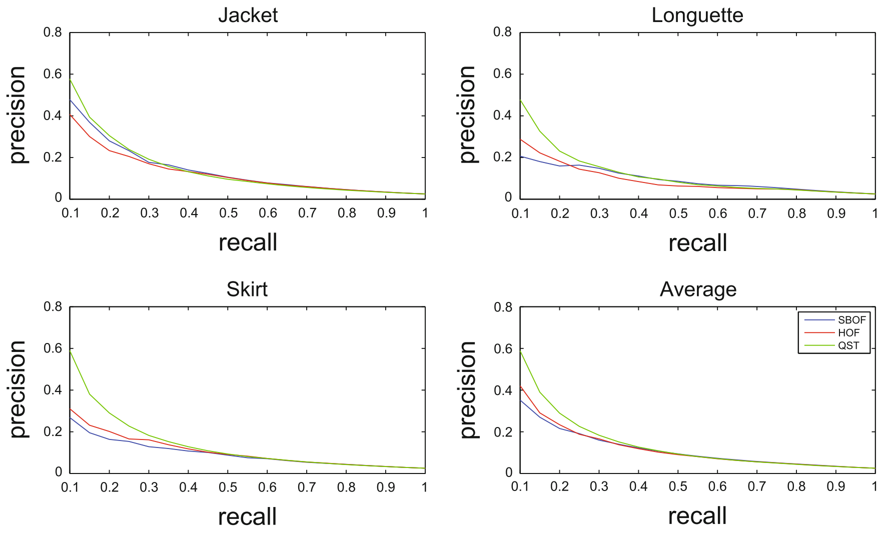


Fig. 11.7 The precision-recall rate over three example category and average rate among all categories

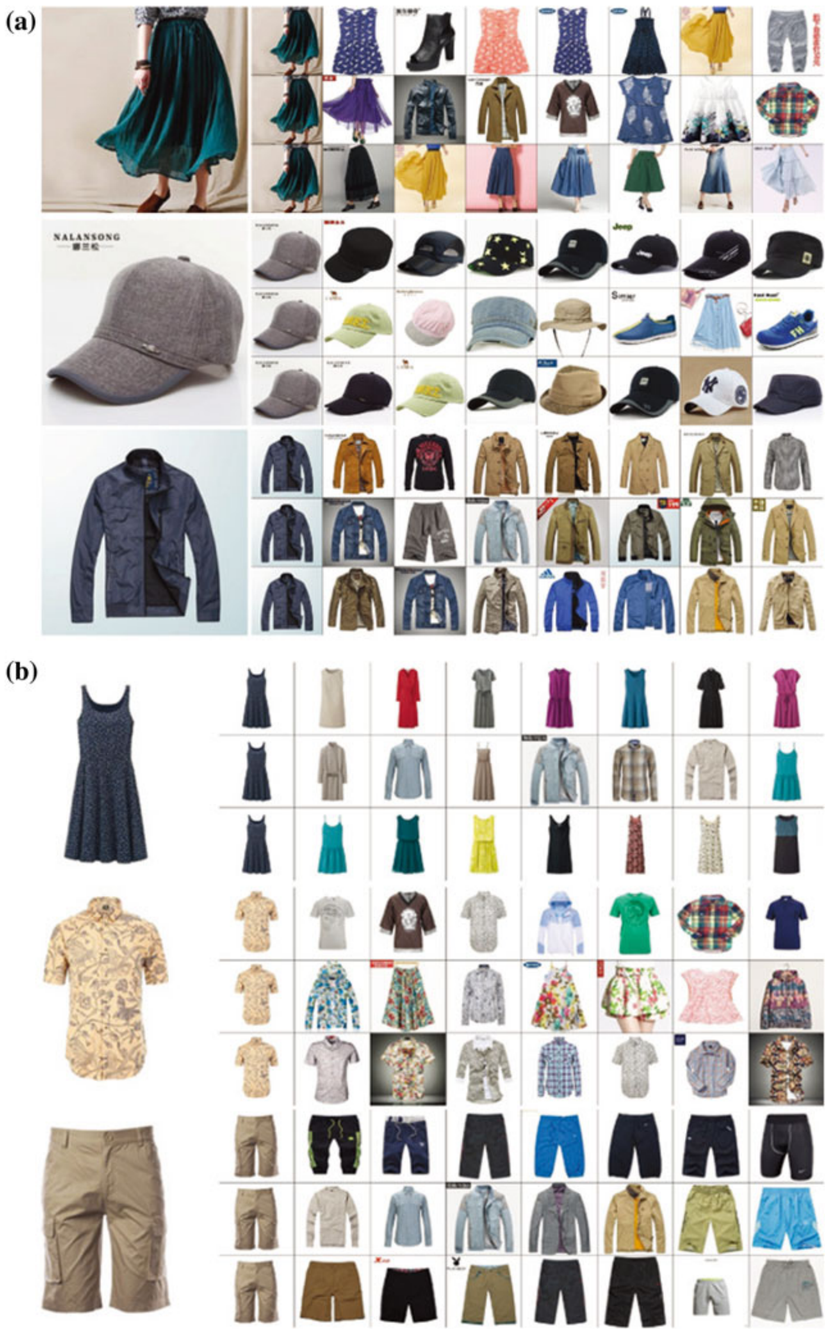


Fig. 11.8 Example of retrieval results. For each query image, the top row is result of HOF, the middle row is SBOF and bottom is QST. **a** longuette, hat, jacket, **b** slip dress, fancy short shirt, casual shorts

as “longuette” and “skirt”, the improvement of effect is significant. There are two main reasons. First, large intra-class shape variation exists in such classes of images. Second, different sellers take photos with various angles in these classes. Therefore, the low-level visual feature does not reflect the contour variation information in this situation.

As shown in Fig. 11.8, we list the recommendations results on several examples with comparison of other methods.

From “longuette” and “hat” in Fig. 11.8a, QST is more robust to the post variation of fashion models compare with SBOF and HOF. Because QST not only utilizes low-level visual features, but also text words. SBOF captures local-precision feature, which is quite powerful in modeling texture as “Fancy short shirt” in Fig. 11.8b, but not enough for style recommendations. HOF is more effective as shown in “hat” in Fig. 11.8a, for the common shape of “hat”. But still HOF is not enough due to Intra-class variation and Inter-class ambiguity.

To show the power of QST, from “Casual shorts” in Fig. 11.8b, QST captures the textual information “casual”, while the results from HOF contains many shape analogous“sports shorts”. As result in “slip dress” Fig. 11.8b, QST captures text like “slip” and “sling” which is hard for visual features to express.

11.6 Conclusion

In this work, we studied the problem of apparel recommendations, and compared the performances of three image retrieval algorithms, HOF, SBOF and QST. The experimental results show QST outperforms the rest two visual feature methods. The main reason is huge variation in intra-class variance and different poses in taking photos, which lead to significant changes in the visual features. And QST takes both text and visual contents into consideration, which generates more robust features than HOF and SBOF in such large variation.

There still exist a number of grand technical challenges, such as clothing style, age suitability, in clothing recommendations for online service. It is still difficult to represent clothing style by using visual features of images. Some new models exploring clothing style are needed to be further developed in order to make image search algorithms suitable for clothing recommendations.

Acknowledgments The work was supported by the national natural science foundation of China (Grant Nos. 91120305, 61272251).

References

1. Bay H, Tuytelaars T, Van Gool L (2006) Surf: speeded up robust features. In: Computer vision-ECCV 2006. Springer, Heidelberg, pp 404–417
2. Berg TL, Berg AC, Shih J (2010) Automatic attribute discovery and characterization from noisy web data. In: Computer vision-ECCV 2010. Springer, Heidelberg, pp 663–676

3. Bourdev L, Maji S, Malik J (2011) Describing people: a poselet-based approach to attribute classification. In: Computer vision (ICCV), 2011 IEEE international conference on IEEE, pp 1543–1550
4. Cao Y, Wang C, Li Z, Zhang L, Zhang L (2010) Spatial-bag-of-features. In: Computer vision and pattern recognition (CVPR), 2010 IEEE conference on IEEE, pp 3352–3359
5. Chen H, Gallagher A, Girod B (2012) Describing clothing by semantic attributes. In: Computer vision-ECCV 2012. Springer, Heidelberg, pp 609–623
6. Czarnecki K, Helsen S (2003) Classification of model transformation approaches. In: Proceedings of the 2nd OOPSLA workshop on generative techniques in the context of the model driven architecture, vol 45, pp 1–17
7. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Computer vision and pattern recognition. CVPR 2005. IEEE computer society conference on IEEE, vol 1, pp 886–893
8. Di W, Wah C, Bhardwaj A, Piramuthu R, Sundaresan N (2013) Style finder: fine-grained clothing style detection and retrieval. In: Computer vision and pattern recognition workshops (CVPRW), 2013 IEEE Conference on IEEE, pp 8–13
9. Fu J, Wang J, Li Z, Xu M, Lu H (2013) Efficient clothing retrieval with semantic-preserving visual phrases. In: Computer vision-ACCV 2012. Springer, Heidelberg, pp 420–431
10. Jégou H, Douze M, Schmid C (2009) Packing bag-of-features. In: Computer vision IEEE 12th international conference on IEEE, pp 2357–2364
11. Liu S, Feng J, Song Z, Zhang T, Lu H, Xu C, Yan S (2012) Hi, magic closet, tell me what to wear! In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 619–628
12. Liu S, Song Z, Liu G, Xu C, Lu H, Yan S (2012) Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In: Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on IEEE, pp 3330–3337
13. Lowe DG (1999) Object recognition from local scale-invariant features. In: The proceedings of the seventh IEEE international conference on computer vision. IEEE, vol 2, pp 1150–1157
14. Mcauliffe JD, Blei DM (2008) Supervised topic models. In: Advances in neural information processing systems, pp 121–128
15. Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. In: Computer vision and pattern recognition, 2006 IEEE computer society conference on IEEE, vol 2, pp 2161–2168
16. Shen E, Lieberman H, Lam F (2007) What am i gonna wear?: scenario-oriented recommendation. In: Proceedings of the 12th international conference on intelligent user interfaces. ACM pp 365–368
17. Sun Z, Wang C, Zhang L, Zhang L (2012) Query-adaptive shape topic mining for hand-drawn sketch recognition. In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 519–528
18. Wang N, Ai H (2011) Who blocks who: simultaneous clothing segmentation for grouping images. In: Computer vision (ICCV), 2011 IEEE international conference on IEEE. pp 1535–1542
19. Wang X, Zhang T (2011) Clothes search in consumer photos via color matching and attribute learning. In: Proceedings of the 19th ACM international conference on multimedia. ACM, pp 1353–1356
20. Yang M, Yu K (2011) Real-time clothing recognition in surveillance videos. In: Image Processing (ICIP), 2011 18th IEEE international conference on IEEE, pp 2937–2940
21. Zhou R, Chen L, Zhang L (2012) Sketch-based image retrieval on a large scale database. In: Proceedings of the 20th ACM international conference on multimedia. ACM, pp 973–976

Chapter 12
Video Retrieval Based on Uncertain Concept Detection Using Dempster–Shafer Theory

Kimiaki Shirahama, Kenji Kumabuchi, Marcin Grzegorzek and Kuniaki Uehara

Abstract For a long time, it was difficult to automatically extract meanings from video shots, because, even for a particular meaning, shots are characterized by signifincantly different visual appearances, depending on camera techniques and shooting environments. One promising approach for this has been recently devised where a large amount of shots are statistically analyzed to cover diverse visual appearances for a meaning. Inspired by the significant performance improvement, *concept-based video retrieval* receives much research attention. Here, concepts are abstracted names of meanings that humans can perceive from shots, like objects, actions, events, and scenes. For each concept, a detector is built in advance by analyzing a large amount of shots. Then, given a query, shots are retrieved based on concept detection results. Since each detector can detect a concept robustly to diverse visual appearances, effective retrieval can be achieved using concept detection results as "intermediate" features. However, despite the recent improvement, it is still difficult to accurately detect any kind of concept. In addition, shots can be taken by arbitrary camera techniques and in arbitrary shooting environments, which unboundedly increases the diversity of visual appearances. Thus, it cannot be expected to detect concepts with an accuracy of 100 %. This chapter explores how to utilize such *uncertain* detection results to improve concept-based video retrieval.

K. Shirahama (✉) · M. Grzegorzek
Pattern Recognition Group, University of Siegen, Hoelderlinstr. 3,
57076 Siegen, Germany
e-mail: kimiaki.shirahama@uni-siegen.de

M. Grzegorzek
e-mail: marcin.grzegorzek@uni-siegen.de

K. Kumabuchi · K. Uehara
Graduate School of System Informatics, Kobe University, 1-1, Rokkodai,
Nada Kobe 657-8501, Japan
e-mail: kumabuchi@ai.cs.kobe-u.ac.jp

K. Uehara
e-mail: uehara@kobe-u.ac.jp

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_12

12.1 Introduction

With the recent progress in multimedia technology, the volume of video data that one can access is explosively increasing on the Web. Since videos are essentially temporal media, even browsing one video often takes a long time. It is necessary to develop video retrieval methods, which efficiently analyze a large number of videos to find shots (video segments) of interest. This has been traditionally investigated as *content-based video retrieval* which retrieves visually similar shots to example shots provided by a user [1]. The term "content-based" in this context means that "low-level" features like colors, edges, and motions are used to represent and describe shots. These features can be directly extracted from shots via some physical metrics or mathematical transformations.

Figure 12.1 illustrates an overview of content-based video retrieval. First, a feature is extracted from each example shot and is generally organized into a vector. In other words, the example shot is represented as a point in a multi-dimensional space. Similarly, shots in the database are located in the multi-dimensional space based on their features. These feature representations of shots are stored as an index to facilitate the retrieval process. Based on this, shots located near to the example shot are retrieved as depicted by the dashed circle in Fig. 12.1.

However, since a feature just represents the physical characteristic of a shot, content-based video retrieval often fails to retrieve shots that are semantically the same or similar to example shots. For example, in Fig. 12.1, the example shot is provided to represent the query "a person appears with a computer." But, as shown on the right side, the irrelevant shot is falsely retrieved together with the relevant one. The reason is that the computer monitor in the example shot and the car window in the irrelevant shot have similar rectangular shapes, and these shots have similar spatial layouts formed by a person and a rectangle. This problem is called the *semantic gap* which is the lack of coincidence between automatically extractable low-level features and user-perceived high-level semantics [2]. In other words, shots containing similar features often display different semantic meanings, and shots displaying similar meanings do not necessarily contain similar features. The semantic gap is the central problem in many multimedia applications like image/video categorization and object recognition, and has received much research attention [3, 4].

The current research progress reveals that one key for bridging the semantic gap is to statistically analyze a large amount of shots. Let us consider the detection of

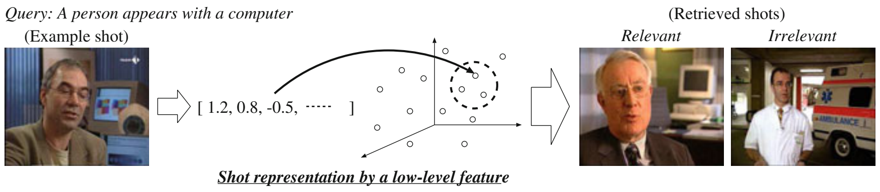

Query: A person appears with a computer
(Example shot)
[1.2, 0.8, -0.5, ·····]
Shot representation by a low-level feature
(Retrieved shots)
Relevant
Irrelevant

Fig. 12.1 An overview of content-based video retrieval

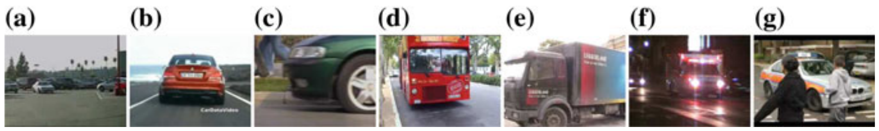


Fig. 12.2 An illustration of diverse visual appearances of cars **a**: Shot taken by a camera at a very long distance. **b**: Shot taken by a camera at a long distance. **c**: Shot taken by a camera at a short distance. **d**: Shot displaying a bus. **e**: Shot displaying a truck. **f**: Shot taken in a dark situation. **g**: Shot where a car is occluded by persons

an object like person, car, or building. The semantic gap is attributed to the diversity of visual appearances of the object, depending on camera techniques and shooting environments. Figure 12.2 illustrates this regarding cars. Figure 12.2a–c present that the visual appearance of a car varies depending on its distance to the camera. Figure 12.2d and e exemplify that different types of cars are shaped differently. Moreover, Fig. 12.2e and f show that lighting conditions and occlusions (a car is masked by other objects) cause different visual appearances. Thus, features of the above shots form significantly different vectors.

One approach for accurately covering such vectors is to compare many shots where an object is present to the ones where it is absent. As a result, it is possible to identify what kind of vectors characterize the presence or absence of the object. Thereby, the object can be accurately detected regardless of camera techniques and shooting environments. In general, the detection performance is proportional to the logarithm of the number of shots to be analyzed, although each object has its own complexity of detection [5]. Along with objects, the approach is applicable to detect various semantic meanings, such as actions, events, and scenes.

The above progress inspires researchers to devise *concept-based video retrieval* which retrieves relevant shots to a query based on detection results of concepts [6]. Here, concepts are textual descriptions of various semantic meanings that can be observed from shots, such as objects like *Person* and *Car*, actions like *Walking* and *Airplane_Flying*, events like *Car_Crash* and *Explosion_Fire*, and scenes like *Beach* and *Desert*. In the following discussion, concept names are written in italics to distinguish them from the other terms. A concept is regarded as present in a shot as long as humans can recognize it from a small region on video frames or short sequence of video frames. This accounts for neither the location of the concept on a video frame nor the time period where it is present. Hence, concept-based video retrieval can be considered as an analogy with the vector space model in text retrieval. This model represents a document as a vector, where each dimension represents the frequency of a word without considering the order of words. Similarly, each concept defines a human-perceivable dimension like a word. Compared to this, dimensions of a low-level feature used in content-based video retrieval just represent physical metrics. In addition, as text retrieval matches words in documents with keywords specified as a query, concept-based video retrieval examines whether shots display concepts exemplified by example shots.

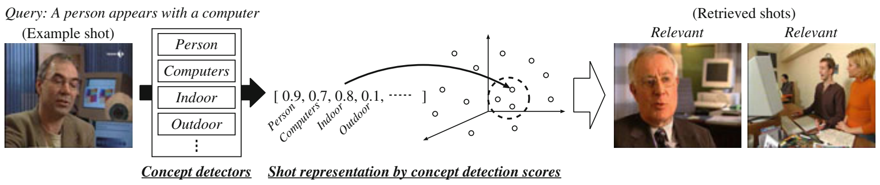


Fig. 12.3 An overview of concept-based video retrieval

Figure 12.3 illustrates an overview of concept-based video retrieval. First of all, for each concept, a *detector* is constructed in advance by analyzing a large amount of shots. The detector outputs *detection scores* each of which represents a scoring value between 0 and 1. Large and small detection scores highlight the concept's presence and absence, respectively. For example, in Fig. 12.3, the detector for *Person* provides the example shot with the score 0.9, meaning that a person probably appears in this shot. On the other hand, the score 0.1 obtained by the detector for *Outdoor* indicates that the example shot is unlikely to show an outdoor scene.

By combining such detection scores for different concepts, each shot is represented as a vector and located in a multi-dimensional space, as shown in the middle part of Fig. 12.3. In this space, shots near to the example shot are retrieved, that is, they are similar to the example shot in terms of detection scores. In particular, the irrelevant shot falsely retrieved in Fig. 12.1 is not retrieved, because it has high detection scores for neither *Computers* nor *Indoors*. Additionally, each concept can be robustly detected owing to the analysis on a large amount of shots. Thus, although the rightmost shot in Fig. 12.3 shows the side view of a computer, it is associated with a high detection score for *Computers* and can be retrieved. In this way, concept-based video retrieval can achieve state-of-the-art performance, where each concept detector works as a semantically meaningful filter of shots [6–10].

This chapter addresses concept-based video retrieval based on the following structure: The next section describes fundamental problems to achieve effective concept-based retrieval, and clarifies the problem that is mainly addressed in this chapter. Section 12.3 presents some ideas necessary for our method, which is closely presented in Sect. 12.4. Section 12.5 evaluates our method and compares it to state-of-the-art methods. In Sect. 12.6, our method is compared to related methods developed in fields other than concept-based retrieval. Finally, Sect. 12.7 concludes this chapter by providing some future directions.

12.2 Fundamental Problems in Concept-Based Video Retrieval

In this section, we explain three fundamental problems in concept-based video retrieval. The first is how to define a vocabulary of concepts, that is, what kinds of concepts should be detected for retrieval. The second problem is how to construct a detector which can accurately distinguish shots where a concept is present from the other shots. The last problem is how to utilize concept detection scores to achieve effective retrieval. Finally, we deeply elaborate the last problem to find the most relevant conclusion of this chapter.

12.2.1 Defining a Concept Vocabulary

A query is characterized by a set of concepts that are present in example shots. Thus, a concept vocabulary should be sufficiently rich for covering various queries. One of the most popular vocabularies is *Large-Scale Concept Ontology for Multimedia* (LSCOM) which defines a standardized set of 1,000 concepts in the broadcast news video domain [11]. These concepts are selected through the collaboration of multimedia researchers, library scientists and end users. Specifically, LSCOM concepts are selected based on their "utility" for classifying content in videos, their "coverage" for responding to a variety of queries, their "feasibility" for automatic detection, and the "availability" (observability) of a large amount of shots to build a detector.

Figure 12.4 shows a part of the hierarchy of LSCOM concepts under the category *OBJECT* (non-human objects). It should be noted that, to the best of our knowledge, there is no officially provided hierarchy of LSCOM concepts, except for the very simple one in [11]. The hierarchy in Fig. 12.4 has been created in our past work [12], where concepts written only with capital letters are newly defined by us to build a semantically meaningful hierarchy.[1] For example, in Fig. 12.4, *Ground_Vehicle* is categorized into *TWO_WHEELS*, *Car* and *Armored_Vehicle*, and *Car* is further classified into *Bus*, *Truck* and *Emergency_Vehicle*. This kind of hierarchy works as useful constraints for not only achieving semantically consistent detection of concepts, but also reducing computational costs [13]. Specifically, if *Bus* or *Truck* is detected in a shot like Fig. 12.2d or e, its parent concept *Car* should be also detected. In addition, if *Car* is not detected in a shot, it is not needed to detect its child concepts. As illustrated by the hierarchy in Fig. 12.4, LSCOM defines a variety

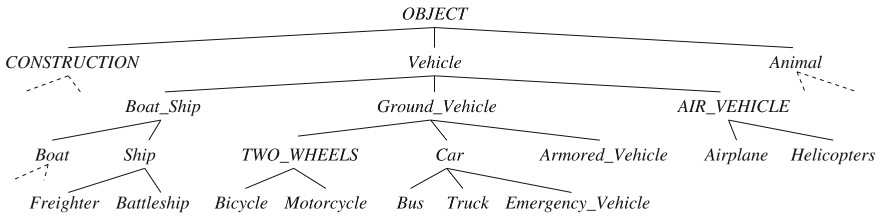


Fig. 12.4 An illustration of hierarchically organized LSCOM concepts under *OBJECT* [12]

[1] Since the purpose of this section is to provide an overview of concept-based retrieval, Fig. 12.4 only presents generalization/specialization relations. Please refer to [12] for other relations (e.g., part-of, attribute-of, and co-occurrence) and our approach for organizing LSCOM concepts.

of concepts ranging from general to specific ones. It is estimated that if the number of LSCOM concepts reaches an amount of 3,000, granting the quality of the new concepts remains similar to the existing ones, the retrieval performance becomes comparable to that of one best Web search engine in text information retrieval [14]. Considering the above logical and practical validity, we use LSCOM concepts.

Ideally, any query should be characterized by "specific" concepts. However, such concepts are not necessarily defined in a vocabulary. For example, *Birthday_Cake* and *Candle* seem useful for retrieving shots relevant to the query "birthday party", but they are not defined in LSCOM. Nonetheless, without using specific concepts, retrieval can be performed using "related" concepts. For instance, if *Indoor*, *Food*, *Table* and *Explosion_Fire* are shown in a shot, their combination implies *Birthday_Cake* and *Candle*, and the shot is probably relevant to "birthday party". To exhaustively cover specific concepts, Deng et al. are currently developing a huge-scale concept vocabulary called ImageNet [15]. This aims to assign in average 500–1,000 images to each of 80,000 concepts. Currently, 3.2 million images have been associated with 5,247 concepts using Amazon's Mechanical Turk [16], through which the assignment of images is outsourced to Web users.

It should be noted that concepts themselves are just linguistic terms. To utilize them in video retrieval, we need to examine whether each concept is contained in the audiovisual form of a shot. Hence, detectors described in the following section serve as mediators between linguistic concepts and their audiovisual forms.

12.2.2 Concept Detection

Concept detection can be formulated as a binary classification problem in machine learning, where for each concept a detector is trained to distinguish shots showing it from the others. This requires two types of training shots, *positive shots* and *negative shots*. The former and latter are shots that are annotated with the concept's presence and absence, respectively. By referring to these training shots, the detector examines *test shots* where neither the concept's presence nor absence is known. As described in Sect. 12.1, accurate concept detection requires a large amount of training shots. These are recently available as a result of Web-based collaborative annotation effort, where manual annotation on large-scale video data is distributed to many users on the Web [17]. Below, assuming that a large amount of training shots are available, we explain how to construct a detector for the concept.

Figure 12.5 shows an overview of concept detection. First, we need to consider how to describe shots that display diverse visual appearances of a concept. Regarding this, a shot is generally represented as a collection of many *descriptors*, each of which describes a local region like circles in Fig. 12.5a. The rationale behind this is that if many descriptors are sampled from a shot, some of them probably represent characteristic regions of the concept. For example, even if the *Car* in Fig. 12.5a is partially masked by other objects, descriptors that characterize a wheel, window, or headlight should be sampled from its visible part.

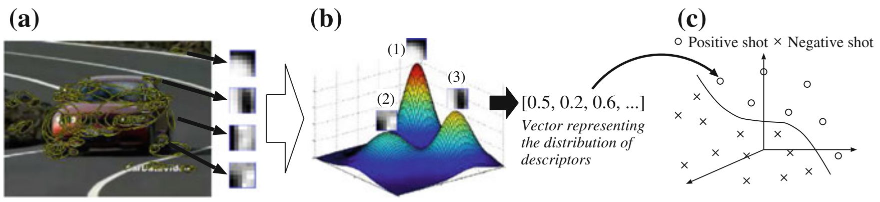

(a)
(b)
(1)
(2)
(3)
[0.5, 0.2, 0.6, ...]
Vector representing the distribution of descriptors
(c)
○ Positive shot × Negative shot

Fig. 12.5 An overview of concept detection based on local region descriptors. **a** Descriptor sampling. **b** Feature extraction. **c** Detector construction

Descriptor sampling involves the following two research themes. The first one is how to detect local regions, from which descriptors are sampled. Several region detectors such as *Harris-Affine region detector* and *Hessian-Affine region detector* [18], have been developed to detect local regions with characteristic edge shapes (e.g., corners and blobs). Another approach is *dense sampling* which exhaustively picks up local regions with a fixed interval. The second research theme is how to describe each detected region. Many descriptors, like *Scale-Invariant Feature Transform* (SIFT) [19], *Speeded Up Robust Features* (SURF) [20] and *RGB SIFT* [21], are currently available. These descriptors are designed to have invariance properties for various factors like changes in illumination, rotation, scaling, and viewpoint. This means that the same or similar descriptors can be sampled from shots, where a concept is present in different lighting conditions or with different directions.

The second problem in concept detection is how to organize sampled descriptors into an overall representation (i.e., feature) of a shot. As shown in Fig. 12.5b, this is generally conducted by computing the distribution of sampled descriptors. For example, the distribution in Fig. 12.5b indicates that the shot includes many descriptors similar to the one marked with (1), and few descriptors similar to (2). A vector representing the characteristic of such a distribution is extracted as the feature of a shot (see Fig. 12.5b). The simplest approach for feature extraction is to first extract characteristic descriptors, called *visual words*, by grouping descriptors into clusters of similar ones [22]. A histogram is then created to represent the number of sampled descriptors assigned to each visual word. A more sophisticated approach is to estimate a Gaussian Mixture Model (GMM) based on descriptors sampled from a shot [23]. Compared to "prespecified" visual words, the GMM precisely represents means and variances of sampled descriptors. Furthermore, the Fisher kernel approach is used not to directly represent the distribution of descriptors, but represent its first and second order differences to the reference distribution [24].

The last problem is how to construct a detector for a concept, using positive and negative shots represented by the above kind of features. The feature of each shot essentially becomes a high-dimensional vector (more than 10,000 dimensions) in order to precisely characterize the distribution of sampled descriptors. Thus, a concept detector is usually constructed as a Support Vector Machine (SVM), which is known as effective for high-dimensional data [25]. As shown in Fig. 12.5c, the SVM utilizes the "margin maximization" principle to place a decision boundary (hyperplane) in the middle between positive and negative shots. This principle theoretically makes the generalization error of the SVM independent of the number of dimensions, if this number is sufficiently large [25]. The decision boundary is used to determine the presence or absence of the concept in test shots. In particular, detection scores are computed as the SVM's probabilistic outputs, each of which approximates the distance between a shot and the decision boundary using a sigmoid function [26].

Finally, in recent days, there are several worldwide competitions like TRECVID [27], PASCAL VOC [28], and ILSVRC [29]. These aim to evaluate concept detectors (object recognizers) developed all over the world using the same large-scale benchmark data. The above competitions have been promoting the improvement of concept detectors. In this chapter, we use a slightly different variant of our concept detector, which achieved the top performance at TRECVID 2012 [30].

12.2.3 Retrieval Based on Concept Detection Scores

While concepts are primitive meanings, queries issued by users are high-level meanings. Thus, the last fundamental problem is how to utilize detection scores for different concepts in order to retrieve shots relevant to a query. This is formulated as a classification problem where a *classifier* (retrieval model), which discriminates between relevant and irrelevant shots to the query, is constructed based on concept detection scores in example shots. In other words, this classifier fuses detection scores for different concepts into a single *relevance score*, which indicates the relevance of a shot to the query. Below, we describe existing classifiers by putting them into four categories, *linear combination*, *discriminative*, *similarity-based* or *probabilistic*.

Linear combination classifiers compute the relevance score of a shot by weighting detection scores for multiple concepts. These classifiers assign large weights to concepts that are related to a query, while small weights are assigned to not related ones. One popular weighting method is to use concept detection scores in example shots. If the average detection score for a concept in example shots is large (small), this concept is regarded as related (not related) to the query and associated with a large (small) weight [10, 31]. Furthermore, text-based weighting methods have been developed by assuming that the text description of a query is available. For example, a concept is associated with a large weight if its name is lexically similar to a term in the text description of the query [8, 10, 31]. The lexical similarity between a concept name and a term can be measured using a lexical ontology like WordNet.

Discriminative classifiers (typically, SVMs) are constructed using example shots [8, 9]. Note that the construction of a discriminative classifier requires both example shots and "counter example shots", which serve as representatives of relevant and irrelevant shots to the query, respectively. In general, only a small number of shots are relevant to the query. Thus, counter example shots are usually collected as randomly sampled shots because almost all of them are irrelevant [32]. A discriminative classifier is constructed by contrasting the above two types of shots. Intuitively, this prioritizes concepts for which detection scores in example shots are significantly large or small, compared to those in counter example shots. The relevance score of a shot is obtained as the classifier's output.

Similarity-based classifiers compute the relevance score of a shot as its similarity to example shots in terms of concept detection scores. It should be noted that such a similarity cannot be "linearly" measured. For example, the difference 0.2 between the detection scores 0.3 and 0.5 means a semantically big difference, because the former indicates that a concept is probably absent, and the latter indicates it is possibly present. On the other hand, the same difference 0.2 between the detection scores 0.6 and 0.8 means a small difference, because both scores indicate that the concept is probably present. Hence, rather than the Euclidean distance, the cosine similarity which examines the correlation between concept detection scores in two shots, and an entropy which examines their probabilistic dependence, are used as similarity measures [7].

Probabilistic classifiers estimate a probability distribution of concepts using detection scores in example shots, and use it to compute the relevance score of a shot. Example shots show many concepts where some of them are related to a query, but the others are not. For example, an example shot for the query "a person appears with a computer" may include not related concepts, such as *Food* and *Windows* shown in the background. Probabilistic classifiers statistically differentiate concepts useful for characterising a query from the others. Rasiwasia et al. computed the relevance score of a shot as the similarity between the multinomial distribution of concepts estimated from that shot, and the one estimated from example shots [33].

The above existing classifiers take little account of the fact that, concept detection is not necessarily accurate. Even using the most effective detectors, it is difficult to accurately detect any kind of concept. For example, at TRECVID 2012 Semantic Indexing task [27], the top-ranked detectors achieved high performances for concepts like *Male_Person* and *Walking_Running*. On the other hand, the detection of concepts like *Bicycling* and *Sitting_down* was difficult. In addition, we target real-world videos which are "uncontrolled" in the sense that, shots can be taken by arbitrary camera techniques and in arbitrary shooting environments. They can display infinite visual appearances of a concept, thus detecting it with an accuracy of 100 % cannot be expected. Relying on such *uncertain* concept detection scores significantly degrades the retrieval performance. The following part of this chapter explores how to handle uncertainties in concept detection to achieve accurate retrieval.

12.3 Overview of Our Concept-Based Video Retrieval

Figure 12.6 illustrates uncertainties in detecting the concept *Crowd*. The horizontal axis represents detection scores where plus (+) and times (×) symbols represent detection scores of shots annotated with *Crowd*'s presence and absence, respectively. It should be noted that these shots are not positive and negative shots used to build the detector for *Crowd*, but are created by annotating shots sampled by our method described in Sect. 12.4.3. Below, for both concept detector construction and

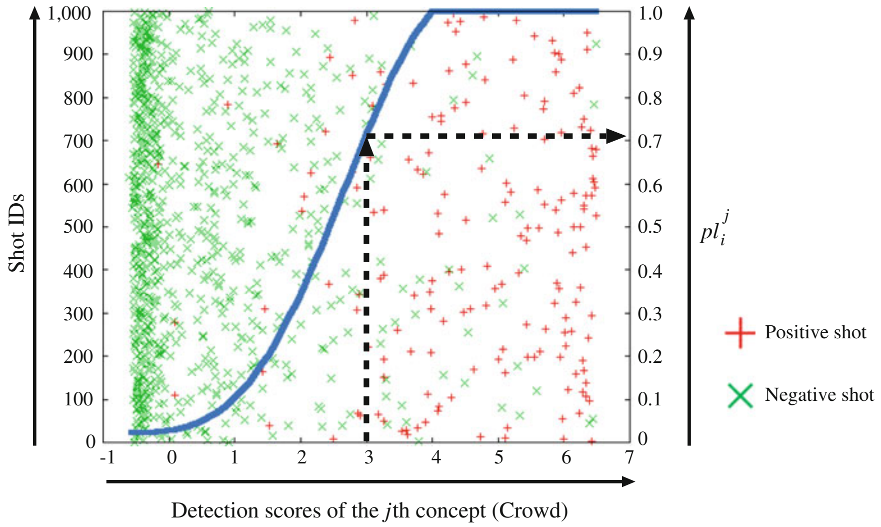


Fig. 12.6 An illustration of uncertainties in detecting *Crowd*

uncertainty exploration, we use the terms "positive shots" and "negative shots" as long as their usage is clear from the context. Although Fig. 12.6 shows detection scores that are normalized to have the mean 0 and the variance 1 (see Sect. 12.5), their magnitude relation is the same as that of original detection scores. Also, for better visualization, shots are distributed according to their IDs as shown in the left vertical axis. In other words, we only care about the distribution of shots on the horizontal axis. Figure 12.6 shows that more positive shots and more negative ones lie on higher and lower detection scores, respectively. However, several positive (negative) shots also exist on low (high) detection scores. We describe such mixtures between positive and negative shots as uncertainties. This section provides an overview of our method for handling uncertainties.

To deal with uncertainties in concept detection, we use *Dempster–Shafer Theory* (DST) which is a generalization of Bayesian theory, where a probability is not assigned to a variable, but instead to a subset of variables [34]. Let $C = \{c_1, \ldots, c_M\}$ be a set of M concepts where c_j $(1 \leq j \leq M)$ represents the jth concept. In addition, assume the ith shot $(1 \leq i \leq N)$ in a set of N shots. In Bayesian theory, two discrete variables indicating c_j's presence and absence are used. The most uncertain state of c_j's presence is represented by the probability 0.5. Compared to this, DST represents uncertainties using the following *mass function* m_i:

Definition 12.1 For any subset S of C $(S \subseteq C)$, the mass function m_i outputs the mass $m_i(S)$, which represents the probability that one concept in S could be present in the ith shot.

For example, the mass $m_i(\{c_j, c_{j'}\})$ $(j \neq j')$ represents the probability that either c_j or $c_{j'}$ could be present in the ith shot. In the extreme case, the mass $m_i(C)$

represents the probability that every concept could be present in the ith shot, that is, it is unknown which concept is present. Using this mass function, DST can represent uncertainties in concept detection, much more powerfully than Bayesian theory.

One big difficulty of DST is how to define a mass function. It is substantially infeasible to prepare training shots for deriving a mass function. The reason is that it is very subjective or impossible to annotate shots from the perspective that one of some concepts could be present. Thus, we avoid the mass function derivation by transforming the procedure of classifier construction based on the set-theoretic operation [35] (see Sect. 12.4.2). As a result, a classifier can be constructed based on the following *plausibility* pl_i^j:

$$pl_i^j = \sum_{S \ni c_j} m_i(S), \tag{12.1}$$

where pl_i^j is the sum of masses that are defined on sets including c_j. In other words, all possibilities that c_j is present in the ith shot are summed up. Thus, the plausibility pl_i^j represents the upper bound probability of c_j's presence in the ith shot. This is useful for recovering "false negative" shots where c_j is actually present, but their detection scores are falsely low.

Now, we describe how to compute plausibilities using detection scores for c_j. Supposing that s_i^j is the detection score of the ith shot for c_j, we derive the following *plausibility function* pl^j

Definition 12.2 The plausibility function pl^j maps any detection score s_i^j into a plausibility of c_j's presence, that is, $pl^j(s_i^j) = pl_i^j$.

Intuitively, a large s_i^j is mapped into a large pl_i^j. More importantly, to recover false negative shots, a relatively large pl_i^j should be obtained even for a small s_i^j, if some positive shots for c_j have similar detection scores to s_i^j.

We model pl^j as a *density ratio* function between positive and negative shots for c_j in terms of detection scores [36]. Here, pl_i^j for s_i^j is computed by checking whether positive shots have similar detection scores to s_i^j. In Fig. 12.6, the dense plot of asterisks shows the plausibility function estimated based on density ratios between positive and negative shots for *Crowd*. The right vertical axis indicates plausibilities. The dashed arrows in Fig. 12.6 show that, since there are several positive shots around the middle detection score $s_i^j = 3.0$, it is mapped into the plausibility $pl_i^j = 0.7$. This means that considering uncertainties in detecting *Crowd*, the middle detection score indicates a high possibility that it is present in a shot. Like this, false negative shots can be recovered.

However, to estimate an appropriate density ratio function, we have to overcome the *imbalanced problem* where the number of positive shots (minority class) is much smaller than that of negative shots (majority class) [37]. Generally speaking, many machine learning algorithms extract a hypothesis by checking its classification accuracy on training shots. This mechanism does not work when the number of positive shots and the one of negative shots are imbalanced. The reason is that, meaningless hypotheses that classify almost all shots into negative are favored, because their classification accuracies on training shots are high. For example, if 10 positive and 990 negative shots are used, the hypothesis that classifies all shots into negative gets 99 % accuracy. In our case, a concept c_j is present only in a small number of shots. Thus, if we annotate randomly selected shots, almost all shots do not show c_j (i.e., negative shots). What is worse, since most of randomly selected shots show meanings that are clearly irrelevant to c_j, their detection scores are very low. This makes it infeasible to estimate a density ratio function over various detection scores for c_j.

The above discussion suggests two important factors. The first one is to balance the number of positive shots and the one of negative shots, and the other is to collect shots from different ranges of detection scores. The latter enables a density ratio function to represent how the distribution of positive and negative shots changes depending on detection scores. To satisfy the above two factors, we have developed an *undersampling* method which selects a subset of the whole set of shots, so as to balance numbers of positive and negative shots [37]. Specifically, we select shots that not only have similar detection scores to those of already sampled positive shots, but also have dissimilar detection scores to those of already sampled shots. The former constraint preferentially selects shots where c_j is likely to be present, and the latter avoids selecting shots with similar detection scores.

Finally, our classifier based on DST is an extension of probabilistic classifiers described in Sect. 12.2.3. While probability theory can only represent the uncertainty of c_j's presence using a probability near to 0.5, DST can precisely represent it based on a mass function. Furthermore, in [10], the uncertainty of c_j's presence was modeled as the accuracy of the detector for c_j, and used to weight detection scores (i.e., linear combination classifier). However, this approach is supported by neither theoretical rationale nor proof. Compared to this, in the following section, we formulate a probabilistic classifier using a mass function in DST, and prove that this classifier can be built only using plausibilities of concepts' presences. To the best of our knowledge, such a precise modeling of uncertainties in concept detection has not yet been explored in existing works.

12.4 Video Retrieval Based on Uncertain Concept Detection

In this section, we first provide a brief explanation of our concept detection method. Then, our classifier based on DST is described. Afterwards, we present a method that derives a plausibility function for each concept, by estimating a density ratio function between positive and negative shots.

12.4.1 Concept Detection

We use a slightly different variant of our concept detection method which achieved the highest performance at TRECVID 2012 Semantic Indexing (light) task [30]. This section briefly presents it, please refer to [30] for more details. One key factor for accurate concept detection is to use a large amount of training shots for covering diverse visual appearances of a concept (see Sect. 12.2.2). Another key factor is attributed to the fact that, a concept appears in different regions in video frames, and it does not necessarily appear in all video frames in a shot. To manage such unclear appearances, it is required to exhaustively sample descriptors in both the spatial and temporal dimensions. Regarding this, Nowak et al. reported that the performance is improved as more descriptors are sampled from images [38]. In addition, Snoek et al. compared the method which uses descriptors extracted only from one video frame in each shot (one shot contains more than 60 frames), to the one that uses descriptors extracted every 15 frames [39]. They found that the latter outperforms the former by 7.5–38.8 %.

However, it requires high computational costs to process a large number of training shots in concept detector construction, and to extract a feature using millions of exhaustively sampled descriptors. Hence, we have developed a fast concept detector construction method and a fast feature exaction method based on matrix operation [30]. The former realizes batch computation of similarities among many training shots, and the latter computes probability densities for many descriptors in a batch. These methods make concept detector construction and feature extraction about 10–37 and 5–7 times faster than the normal implementation, respectively.

In accordance with these fast methods, we summarize our concept detection method. First, three image features (*SIFT-Har*, *SIFT-Hes* and *STD-RGB-SIFT*) and two motion features (*Traj-Disp* and *Traj-HOG*) are extracted from each shot. Image features are extracted by exhaustively sampling descriptors which characterize local regions in every other video frame within the shot. Here, different local regions are identified depending on each image feature. Motion features are extracted by tracking points in the shot based on optical flow fields [40]. These points are exhaustively located at every fifth pixel in each video frame. One motion feature (*Traj-Disp*) is extracted using descriptors each of which represents a sequence of displacements of a tracked point, and another (*Traj-HOG*) uses descriptors each represents the edge around a tracked point. Each feature is represented using a GMM which characterizes the distribution of descriptors in the shot.

For every concept, we construct five detectors. That is, for each of the above five features, a detector is constructed as an SVM using 30,000 training shots. These shots are collected from annotation data on TRECVID 2012 video data, where 545,873 shots are annotated with the presence or absence of an LSCOM concept [17]. Finally, we combine five detectors on different features into a final detector to achieve an improved performance. In total, we obtain final detectors for 351 concepts since annotation data contain more than one positive shots for them.

The above concept detection method and our method in [30] are different in the following two points: First, in [30], we used an audio feature in addition to five features described above. Second, to further cover diverse visual appearances of a concept, we adopted "bagging" where three detectors are constructed on each feature using different sets of randomly sampled training shots. Since the performance improvement using the audio feature and bagging is about 1 %, they are not used in concept detection in this chapter.

12.4.2 Classifier Based on DST

Given N example shots for a query, we construct a classifier by incorporating a mass function in DST into maximum likelihood estimation [35]. Assuming a set of M concepts $C = \{c_1, \ldots, c_M\}$, we represent the ith example shot $\mathbf{x}_i$ as an M-dimensional "complete" vector $(x_i^1, \ldots, x_i^M)^{\mathrm{T}}$. Here, the jth dimension x_i^j represents the presence or absence of the jth concept c_j with no uncertainty. That is, x_i^j is defined as a binary variable $x_i^j \in \{0, 1\}$ where 0 and 1 represent c_j's absence and presence, respectively. Classifier construction aims to estimate a probability distribution which characterizes each concept's presence specific to example shots.

More specifically, assuming that concepts are independent of each other, we model the probability distribution of their presences in $\mathbf{x}_i$ as follows:

$$p(\mathbf{x}_i; \boldsymbol{\theta}) = \prod_{j=1}^{M} (\theta^j)^{x_i^j}, \tag{12.2}$$

where $p(\mathbf{x}_i; \boldsymbol{\theta})$ is defined on the complete vector $\mathbf{x}_i$, and parameterized by $\boldsymbol{\theta} = \{\theta^1, \ldots, \theta^M\}$, in which θ^j represents the probability of c_j's presence. Equation (12.2) means that $p(\mathbf{x}_i; \boldsymbol{\theta})$ is computed by multiplying θ^j for which $x_i^j = 1$. In other words, $p(\mathbf{x}_i; \boldsymbol{\theta})$ is computed as the joint probability of presences of concepts displayed in $\mathbf{x}_i$. Note that c_j's presence and absence are "complementary" because c_j's presence guarantees its absence and vice versa, so considering both of them is redundant. Thus, we only consider c_j's presence in $p(\mathbf{x}_i; \boldsymbol{\theta})$. Based on the definition of $p(\mathbf{x}_i; \boldsymbol{\theta})$, classifier construction is equivalent to the estimation of the optimal $\boldsymbol{\theta}$.

However, since the detection of c_j's presence in $\mathbf{x}_i$ is uncertain, x_i^j incurs an uncertainty which is modeled by a mass function m_i. We define a set of mass functions $\mathbf{m} = \{m_1, \ldots, m_N\}$ where m_i is the mass function for the ith example shot $\mathbf{x}_i$, and outputs the mass that one concept in $S \subseteq C$ could be present in $\mathbf{x}_i$. Based on [35], we incorporate $\mathbf{m}$ into the following likelihood function:

$$L(\boldsymbol{\theta}; \mathbf{m}) = \prod_{i=1}^{N} \left(\sum_{S \subseteq C} m_i(S) \sum_{c_j \in S} \theta^j \right), \tag{12.3}$$

where the right summation in the parentheses picks up concepts included in S. Then, based on Eq. (12.2), probabilities of their presences are summed up as the overall probability that one concept in S is present in $\mathbf{x}_i$. This overall probability on the "complete" vector representation is weighted by the mass $m_i(S)$. The left summation sums such weighted probabilities for all possible subsets of concepts. Thus, the inside of the parentheses can be considered as the likelihood of $\boldsymbol{\theta}$ given m_i for $\mathbf{x}_i$. Hence, assuming that example shots are independent of each other, the overall likelihood is obtained by multiplying the above likelihoods.

By considering the inclusive relation between S and c_j, Eq. (12.3) can be transformed by swapping two summations:

$$\prod_{i=1}^{N}\left(\sum_{j=1}^{M}\theta^j \sum_{S\ni c_j} m_i(S)\right) \tag{12.4}$$

$$= \prod_{i=1}^{N}\left(\sum_{j=1}^{M}\theta^j pl_i^j\right) = L(\boldsymbol{\theta}; \mathbf{pl}). \tag{12.5}$$

In Eq. (12.4), the right summation sums all masses which are defined on S including c_j. This is exactly the plausibility of c_j's presence in $\mathbf{x}_i$, so Eq. (12.4) can be transformed into (12.5). As a result, the likelihood of $\boldsymbol{\theta}$ given $\mathbf{m}$ becomes the one given a set of plausibility functions for M concepts $\mathbf{pl} = \{pl^1, \ldots, pl^M\}$. Therefore, instead of using $\mathbf{m}$, $\boldsymbol{\theta}$ can be estimated using $\mathbf{pl}$. Equation (12.5) indicates that, estimating $\boldsymbol{\theta}$ which maximizes $L(\boldsymbol{\theta}; \mathbf{pl})$ is equivalent to maximizing the agreement between the probability of c_j's presence on the complete vector representation, and the uncertain detection of its presence. In other words, the latter works to recover shots where the detector misses c_j's presence, while the former examines the statistical validity of pl_i^j to alleviate recovering many irrelevant shots.

To estimate $\boldsymbol{\theta}$, we use the modified Expectation-Maximisation (EM) algorithm [35]. It iteratively updates $\boldsymbol{\theta}$ to maximize the expectation of the log-likelihood, where $\mathbf{pl}$ is used to weight the probability that $\mathbf{x}_i$ is observed based on the current $\boldsymbol{\theta}$. It is guaranteed that an updated $\boldsymbol{\theta}$ always improves the likelihood in Eq. (12.5). The algorithm iterates "Expectation-step" (E-step) and "Maximization-step" (M-step). The former computes the above expectation of the log-likelihood using the current $\boldsymbol{\theta}$, and the latter updates $\boldsymbol{\theta}$ so as to maximize that expectation. These E-Step and M-Step are repeated until the improvement of the likelihood becomes smaller than the threshold (please refer to [35] for more details).

Finally, using the estimated $\boldsymbol{\theta}$, retrieval is performed by computing the relevance score of a shot $\mathbf{x}_t$ as follows:

$$\mathrm{rel}(\mathbf{x}_t) = \sum_{j=1}^{M}\theta^j pl_t^j, \tag{12.6}$$

where pl_t^j is the plausibility of c_j's presence in $\mathbf{x}_t$. Thus, rel($\mathbf{x}_t$) examines the agreement between plausibilities of concepts' presences in $\mathbf{x}_t$ and probabilities of their presences with no uncertainty. From another perspective, θ^j can be considered as the weight to represent whether the plausibility of c_j's presence is useful for characterizing the query. The set of 1,000 shots with the largest rel($\mathbf{x}_t$) is returned as a retrieval result.

12.4.3 Modeling Plausibilities

In this section, we describe a method which derives a plausibility function pl^j for the jth concept c_j by estimating a density ratio function between positive and negative shots. As described in Sect. 12.3, pl^j computes pl_i^j which is the plausibility of c_j's presence in the ith shot $\mathbf{x}_i$, using its detection score s_i^j (i.e., $pl_i^j = pl^j(s_i^j)$). We define pl_i^j based on the following density ratio function $w^j(s_i^j)$:

$$pl_i^j = w^j(s_i^j) = \frac{p_1(s_i^j)}{p_0(s_i^j)}, \tag{12.7}$$

where $p_1(s_i^j)$ and $p_0(s_i^j)$ are probability density functions for c_j's presence and absence on s_i^j, respectively. Since we have no prior knowledge about the distribution of positive or negative shots with respect to s_i^j, we use the method, called *unconstrained Least-Squares Importance Fitting* (uLSIF), which can directly estimate $w^j(s_i^j)$ without estimating $p_1(s_i^j)$ or $p_0(s_i^j)$ [36]. Specifically, uLSIF performs least-square fitting to approximate $w^j(s_i^j)$ as the following linear combination of basis functions:

$$w^j(s_i^j) = \sum_{l=1}^{b} \alpha_l \phi_l(s_i^j), \tag{12.8}$$

where α_l is a weight for the lth basis function $\phi_l(s_i^j)$, which is defined as a Gaussian function. And, b is the number of basis functions used to approximate $w^j(s_i^j)$. We set b to the number of integers between the maximum and minimum normalized detection scores. Based on Eq. (12.8), $w^j(s_i^j)$ is estimated by computing the optimal $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_b\}$ using positive and negative shots for c_j. This optimal $\boldsymbol{\alpha}$ can be efficiently obtained because least-square fitting of $w^j(s_i^j)$ is formulated as a system of linear equations (please refer to [36] for details).

To estimate an appropriate density ratio function, we need to prepare a proper set of positive and negative shots by solving the imbalanced problem. To this end, we develop *k-Nearest Neighbor-based Undersampling* (kNNU) to selectively sample shots, which not only are likely to show c_j but also have different detection scores. First, to avoid sampling several shots with the same detection score, kNNU creates a set of shots by retaining only one shot for each of unique detection scores. Then, for each shot, it computes the *priority score* which represents the usefulness for sampling that shot. The shot with the highest priority score is sampled. This process is repeated until the prespecified number of shots are sampled. Finally, a user annotates shots sampled by kNNU to create positive and negative shots for c_j.

In kNNU, the priority score of a shot $\mathbf{x}$, $p(\mathbf{x})$, is computed as follows:

$$p(\mathbf{x}) = \frac{1}{k_1} \sum_{s=1}^{k_1} d(\mathbf{x}, \mathbf{x}_s) - \frac{1}{k_2} \sum_{p=1}^{k_2} d(\mathbf{x}, \mathbf{x}_p), \qquad (12.9)$$

where k_1 shots in $\{\mathbf{x}_s\}_{s=1}^{k_1}$ are already sampled shots having the most similar detection scores to that of $\mathbf{x}$. Similarly, k_2 shots in $\{\mathbf{x}_p\}_{p=1}^{k_2}$ are already obtained positive shots having the most similar scores to that of $\mathbf{x}$. We heuristically set both k_1 and k_2 to 3. The function d represents the Euclidean distance between two shots in terms of their detection scores for c_j. The first term in Eq. (12.9) is the average of distances between $\mathbf{x}$ and $\{\mathbf{x}_s\}_{s=1}^{k_1}$, in order to avoid sampling shots with similar detection scores. The second term is the average of distances between $\mathbf{x}$ and $\{\mathbf{x}_p\}_{p=1}^{k_2}$. This gives high priorities to shots which are similar to already obtained positive shots, so that they are likely to show c_j. We average distances using the k_1 or k_2 most similar shots, to avoid sampling shots having exceptionally very similar detection scores. By annotating sampled shots, we can examine inaccuracies on various detection scores by alleviating the influence of too many shots where c_j is absent.

By annotating shots sampled by kNNU, an appropriate density ratio function can be estimated and used as a plausibility function pl^j. In this way, for all concepts, a set of plausibility functions $\mathbf{pl} = \{pl^1, \ldots, pl^M\}$ is obtained and used to construct a classifier described in Sect. 12.4.2.

12.5 Experimental Results

Our concept-based video retrieval method is evaluated on TRECVID 2009 video data, which consist of 36,106 shots in 219 development videos and 97,150 shots in 619 test videos. According to the official instruction of TRECVID 2009 Search task [27], for each of 24 queries (see Appendix for more details), a classifier is constructed using four to eight example shots in development videos, and then applied to shots in test videos. We mainly evaluate the retrieval performance using a *precision* which represents the percentage of relevant shots within 1,000 retrieved shots. In other words, the precision indicates the number of retrieved relevant shots. Our main purpose is to examine whether diverse relevant shots can be retrieved by recovering shots where detectors missed concepts' presences. If two methods use the same concept detection scores and the precision of one method is larger than that of another, it can be considered that the former retrieved more relevant shots by managing uncertain concept detection better than the latter.

We describe detailed configurations to derive plausibility functions. First of all, 351 concept detectors are built using shots in TRECVID 2012 video data, and then applied to shots in TRECVID 2009 video data.[2] For each concept, a plausibility function is derived by sampling shots from TRECVID 2012 video data, and annotating them as positive or negative. Subsequently, the plausibility function is applied to shots in TRECVID 2009 video data. However, we found that these two datasets are characterized by different distributions of detection scores. This means that depending on datasets, the same detection score indicates a different density ratio between shots where the concept is present and shots where it is absent. Thus, we uniform (normalize) the distribution of detection scores in each dataset, so as to have the mean 0 and the variance 1. Afterwards, we derive a plausibility function from TRECVID 2012 video data, and apply it to TRECVID 2009 video data.

In addition, a density ratio function is estimated using 1,000 shots sampled by our undersampling method (kNNU). Here, if sampled shots do not cover diverse detection scores, we further sample additional 1,000 shots. This is repeated until sampled shots range over most detection scores or 5,000 shots are sampled. It may happen that even annotating 5,000 shots, only a small number of positive shots are collected. They are insufficient for estimating a density ratio function. Thus, if only less than 30 positive shots are collected, we do not estimate a density ratio function and directly use detection scores as plausibilities.

Furthermore, for some concept, the estimated density ratio function may not monotonically increase in terms of detection scores. Specifically, the maximum density ratio lies at a certain detection score, and density ratios at larger detection scores are smaller. This results in deriving a semantically inconsistent plausibility function. To avoid this, we force to continuously assign the maximum density ratio to the above larger detection scores.

Our retrieval method adopts the following two small extensions: First, many of 351 concepts are not so related to a query. Such not related concepts are redundant and degrade the retrieval performance. Thus, we select 10 related concepts as the ones having the highest averages of detection scores in example shots. Then, a classifier is built using plausibilities for selected 10 concepts. The above concept selection has been used in several state-of-the-art concept-based retrieval methods [31, 41]. Second, a plausibility is the upper bound probability of a concept's presence. We found that for most of 10 selected concepts, plausibilities in example shots are commonly large. This makes it difficult to prioritize concepts in terms of how useful they are for characterizing a query. Hence, for each concept, the average detection score in example shots is used to weight original plausibilities. A classifier is constructed

[2] Since the search task has been stopped at TRECVID 2009, videos of this year are the latest ones where the retrieval performance using example shots can be evaluated.

and tested using these weighted plausibilities for example shots and shots in test videos. Our preliminary experiment showed that weighted plausibilities always lead to better performance than original ones.

12.5.1 Evaluation for Video Retrieval Using Plausibilities

In order to examine the effectiveness for handling uncertain concept detection based on plausibility functions, our method denoted by *PL* is compared to the method *Direct*, which constructs a classifier directly using concept detection scores. Specifically, in *Direct*, a classifier is constructed by replacing pl_i^j in Eq. (12.5) with the detection score s_i^j. Table 12.1 shows the performance comparison between *PL* and *Direct* in terms of their precisions. Due to the space limitation, Table 12.1 is divided into three parts, in each of which the top, second, and third rows represent IDs of 24 queries, *PL*'s precisions and *Direct*'s precisions, respectively. In addition, for each method, we show the Mean of Precision (MP) over 24 queries. Moreover, for queries where *PL* outperforms *Direct*, we depict the latter's precisions in italics.

As can be seen from Table 12.1, for 17 of 24 queries, *PL* can retrieve more relevant shots than *Direct*. Also, precisions of *PL* and *Direct* are the same on three queries, and the former is outperformed by the latter on the remaining four queries. Overall, as seen from MPs of *PL* and *Direct* in Table 12.1, the former can recover much more relevant shots than the latter, by modeling uncertainties in concept detection using plausibility functions.

Table 12.1 Performance comparison between *PL*, *Direct* and *Random* in terms of precisions

Query ID	269	270	271	272	273	274	275	276	277
PL	0.108	0.323	0.202	0.008	0.023	0.076	0.014	0.015	0.050
Direct	*0.104*	*0.304*	0.209	*0.007*	0.025	*0.071*	0.015	*0.011*	*0.013*
Random	*0.104*	*0.304*	0.209	0.015	0.025	*0.055*	0.015	*0.011*	0.091
Query ID	278	279	280	281	282	283	284	285	286
PL	0.300	0.002	0.013	0.129	0.055	0.003	0.235	0.235	0.195
Direct	*0.256*	0.002	*0.005*	*0.122*	*0.046*	0.003	*0.217*	0.236	0.195
Random	*0.257*	0.002	0.013	*0.122*	*0.047*	0.003	*0.217*	0.236	0.195
Query ID	287	288	289	290	291	292	MP		
PL	0.130	0.081	0.133	0.338	0.022	0.005	0.112		
Direct	*0.066*	*0.022*	*0.106*	*0.234*	*0.005*	*0.002*	0.094		
Random	*0.119*	*0.049*	0.182	*0.234*	*0.005*	0.006	0.104		

12.5.2 Evaluation for Plausibility Modeling Using Selective Sampling

In order to examine the effectiveness of our undersampling method (kNNU), we compare *PL* in the previous section to *Random*, which uses plausibility functions derived by annotating randomly sampled shots. Here, for each of *PL* and *Random*, the same number of shots are annotated. For both of them, if only less than 30 positive shots for a concept are collected, detection scores are directly used as plausibilities. The bottom row in Table 12.1 shows *Random*'s precisions. As like *Direct*, *Random*'s precisions in italic indicate that *PL* achieves larger precisions.

As shown in Table 12.1, *PL* outperforms *Random* on 12 queries, and their precisions are the same on four queries. For queries where *PL* is outperformed by *Random*, the difference in their precisions are small except for the 277th and 289th queries. Thus, we can say that compared to randomly sampled shots, shots sampled by kNNU yield more useful plausibility functions for managing uncertainties in concept detection. In particular, targeting 158 concepts which are selected for at least one of 24 queries, we collected total 10, 448 positive shots using kNNU. On the other hand, from randomly selected shots, we only collected 6,288 positive shots due to the imbalanced problem.

Table 12.1 shows one interesting observation. First, the comparison between *Random* and *Direct* indicates that, plausibility functions even by annotating randomly sampled shots yield improved performance, compared to directly using concept detection scores. And, further improvement can be achieved by annotating shots sampled by kNNU, as seen from the comparison between *PL* and *Random*.

12.5.3 Performance Comparison with Other Methods

In this section, we compare *PL* to state-of-the-art video retrieval methods, especially 88 methods developed at TRECVID 2009 Search task (fully automatic category) [27]. For each of 24 queries, Fig. 12.7 shows the comparison of *PL*'s precision to the highest and median precisions among the above 88 methods. Overall, the MP of *PL* (0.1124) is ranked at the 18th position among 88 methods. Also, in TRECVID 2009 Search task, an Average Precision (AP) was used as the official evaluation measure [27]. It represents the average of precisions at every position where a relevant shot is ranked. The AP becomes large when relevant shots are ranked at higher positions. The mean of APs (MAP) over 24 queries is used as the overall performance evaluation measure. Regarding this, *PL*'s MAP (0.0772) is ranked at the 16th positions among 88 methods. Thus, overall, *PL* is ranked within the top quartile.

We consider the above result as notable because of the following big handicaps for *PL*: Each query in TRECVID 2009 Search task consists of the text description,

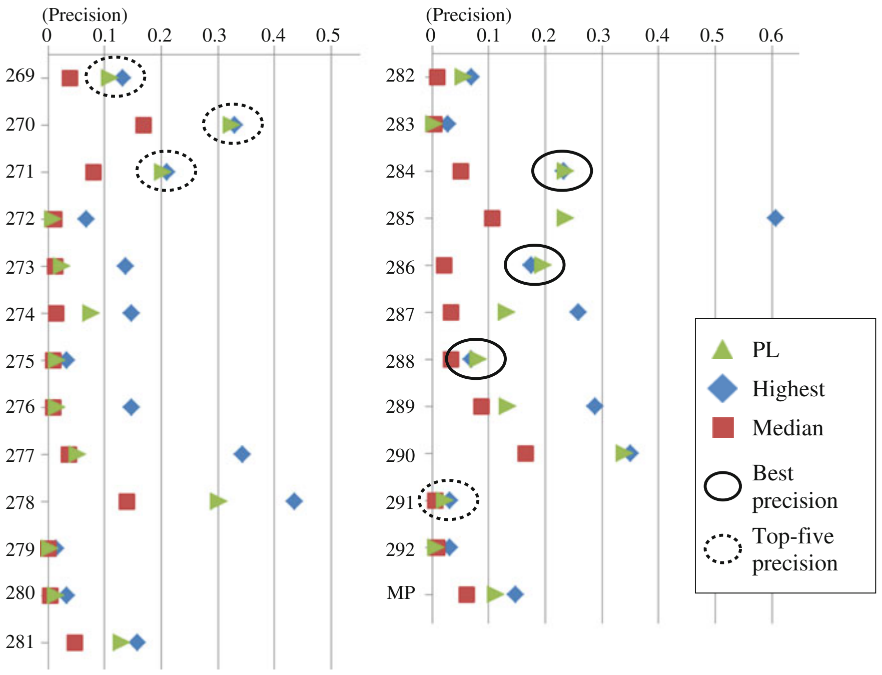


Fig. 12.7 Comparison of *PL*'s precisions with the highest and median precisions at TRECVID 2009 Search task

example shots from development videos, and images collected on the Web. Many methods use all of these query representations to improve the retrieval performance. On the other hand, *PL* only uses example shots since processing text data is out of the scope of this chapter, and motion features (*Traj-Disp* and *Traj-HOG*) for concept detection cannot be extracted from images. In addition, many methods combine concept-based retrieval with retrieval based on Automatic Speech Recognition (ASR) and retrieval using low-level features. We did not add the latter two techniques to *PL*, because adding them makes it vague to examine whether our main purpose (i.e., managing uncertain concept detection) is achieved or not. Thus, it is notable that *PL* only using concept-based retrieval with example shots achieved the performance better than the top-quartile.

In particular, despite the above big handicaps, *PL* achieves the best precision among 88 methods for three queries, marked by solid circles in Fig. 12.7. In other words, for these queries, *PL* covers the largest number of relevant shots. Furthermore, for four queries marked with dashed circles, precisions of *PL* are ranked within the fifth position. Therefore, it can be concluded that *PL* successfully manages uncertain concept detection to recover shots where detectors missed concepts' presences, so that diverse relevant shots can be retrieved. For the other queries, especially the ones where *PL*'s precision is significantly lower than the highest one, we think that *PL* needs to be extended by adopting retrieval techniques based on ASR and low-level features.

12.6 Existing Methods for Uncertain Data

This section aims to clarify the characteristics of our method. To this end, irrespective of research fields, we review existing methods for treating uncertain data, and compare them to our method. Although an uncertainty has been addressed in fields of data mining and machine learning, it is generally defined as the variance around a data point in a multi-dimensional space [42–44]. Examples of such uncertainties are noises and transmission errors arising from a sensor network, and positional information predicted by a mobile device. Existing data mining and machine learning methods probabilistically model these uncertainties, using a probability that a data point lies in a certain region [43], using a probability that the distance between two data points is in a certain range [44], and measuring the density around a data point [42]. Compared to this, we define an uncertainty as the inaccuracy of assigning a certain value (i.e., concept's presence or absence) to a shot. Thus, the above methods cannot be used to deal with uncertainties in this chapter.

Several methods have been proposed to model uncertain data by deriving mass and plausibility functions. However, most of them assume special kinds of data like multivariate (transactional) data [45] and data with nested structures [46], or assume an underlying data distribution like Gaussian distribution [47]. Compared to this, we target multi-dimensional categorical data where each dimension represents a concept's presence or absence. In addition, we do not have any prior knowledge about the data distribution. Hence, we adopt a "supervised" approach which derives a plausibility function for each concept (dimension) using positive and negative shots. Moreover, existing methods do not consider the imbalanced problem underlying the data distribution.

Some researchers addressed uncertainties in "multimodal" concept detection where detection results on different features (modalities) are combined to improve the performance [48, 49]. Here, uncertainties arise when only using a single feature. In [49], concept detection results on different features are combined based on Portfolio theory, so that for each feature, the expected detection accuracy is maximized and the uncertainty is minimized. Note that in [49] the uncertainty for each feature is defined as the variance of correctly classified training shots. Compared to this, an uncertainty in this chapter is inaccurate detection of a concept. Also, Benmokhtar and Huet used DST to represent uncertainties in concept detection using a single feature [48]. However, handcrafted mass functions were used, so their appropriateness for representing uncertainties is not guaranteed. In this chapter, we define a plausibility function for each concept based on density ratios between positive and negative shots in terms of detection scores. This function represents statistically supported uncertainties of the concept's presence.

12.7 Conclusion and Future Work

In this chapter, we introduced a concept-based video retrieval method which can manage uncertainties in concept detection based on DST. Considering the difficulty of directly deriving a mass function, we theoretically proved that a classifier can be built using plausibility functions instead of the mass function. Then, we presented a method which derives a plausibility function for each concept as a density ratio function between positive and negative shots. In particular, to overcome the imbalanced problem between these two types of shots, an undersampling method has been developed to select a subset of shots, which not only are likely to display the concept but also have diverse detection scores. Experimental results verified that using plausibility functions yields better retrieval performance than directly using detection scores, and our undersampling method is useful for deriving effective plausibility functions. Furthermore, we showed that our method achieves state-of-the-art retrieval performance, only using much smaller amount of information than other methods.

In the future, we will explore the following two issues: First, our current classifier is a "generative" model which estimates parameters by maximizing the probability of observing (generating) example shots. However, many publications report that in addition to example shots, using counter example shots significantly improves the retrieval performance [50]. Thus, we plan to incorporate plausibility functions into a "discriminative" model where parameters are estimated to maximize the conditional probability of example (or counter example) shots [51].

Second, we feel that until now retrieval methods have been sophisticated only from the computational perspective. In other words, they adopt little knowledge about human interpretation of semantic meanings. For example, if a *Person* moves his/her *Hand* near a moving *Ball*, one can infer that this person throws the ball. However, this kind of "reasoning" has not yet been explored in large-scale video retrieval. One main reason can be considered as uncertainties in concept detection. Therefore, we aim to apply our probabilistic classifier to reasoning based on concept definitions, properties, and relations defined in an ontology.

Acknowledgments The research work by Kimiaki Shirahama has been funded by the Postdoctoral Fellowship for Research Abroad by Japan Society for the Promotion of Science (JSPS). Also, this work was in part supported by JSPS through Grand-in-Aid for Scientific Research (B): KAKENHI (26280040).

Appendix

We evaluate our video retrieval method on 24 queries specified at TRECVID 2009 Search task [27]. For each query, shots in test videos are manually assessed based on the following criteria: A shot is relevant to the query if it contains a sufficient evidence for humans to recognize the relevance. In other words, such an evidence may be shown only in a region on some video frames in a shot. Below, we show the ID and text description of each query:

269:	Find shots of a road taken from a moving vehicle through the front window
270:	Find shots of a crowd of people, outdoors, filling more than half of the frame area
271:	Find shots with a view of one or more tall buildings (more than four stories) and the top story visible
272:	Find shots of a person talking on a telephone
273:	Find shots of a closeup of a hand, writing, drawing, coloring, or painting
274:	Find shots of exactly two people sitting at a table
275:	Find shots of one or more people, each walking up one or more steps
276:	Find shots of one or more dogs, walking, running, or jumping
277:	Find shots of a person talking behind a microphone
278:	Find shots of a building entrance
279:	Find shots of people shaking hands
280:	Find shots of a microscope
281:	Find shots of two more people, each singing and/or playing a musical instrument
282:	Find shots of a person pointing
283:	Find shots of a person playing a piano
284:	Find shots of a street scene at night
285:	Find shots of printed, typed, or handwritten text, filling more than half of the frame area
286:	Find shots of something burning with flames visible
287:	Find shots of one or more people, each at a table or desk with a computer visible
288:	Find shots of an airplane or helicopter on the ground, seen from outside
289:	Find shots of one or more people, each sitting in a chair, talking
290:	Find shots of one or more ships or boats, in the water
291:	Find shots of a train in motion, seen from outside
292:	Find shots with the camera zooming in on a person's face

References

1. Petkovic M, Jonker W (2002) Content-based video retrieval: a database perspective. Kluwer Academic Publishers, Boston
2. Smeulders A, Worring M, Santini S, Gupta A, Jain R (2000) Content-based image retrieval at the end of the early years. IEEE Trans Pattern Anal Mach Intell 22(12):1349–1380
3. Djordjevic D, Izquierdo E, Grzegorzek M (2007) User driven systems to bridge the semantic gap. In: Proceedings of the EUSIPCO 2007, pp 718–722
4. Staab S, Scherp A, Arndt R, Troncy R, Grzegorzek M, Saathoff C, Schenk S, Hardman L (2008) Semantic multimedia. In: Baroglio C, Bonatti PA, Małuszyński J, Polleres A, Schaffert S (eds) Reasoning Web. LNCS.Springer, Berlin

5. Naphade MR, Smith JR (2004) On the detection of semantic concepts at TRECVID. In: Proceedings of the MM 2004, pp 660–667
6. Snoek CGM, Worring M (2009) Concept-based video retrieval. Found Trends Inf Retr 2(4):215–322
7. Li X, Wang D, Li J, Zhang B (2007) Video search in concept subspace: a text-like paradigm. In: Proceedings of the CIVR 2007, pp 603–610
8. Natsev AP, Haubold A, Tešić J, Xie L, Yan R (2007) Semantic concept-based query expansion and re-ranking for multimedia retrieval. In: Proceedings of the MM 2007, pp 991–1000
9. Ngo C et al (2009) VIREO/DVMM at TRECVID 2009: high-level feature extraction, automatic video search and content-based copy detection. In: Proceedings of the TRECVID 2009, pp 415–432
10. Wei XY, Jiang YG, Ngo CW (2011) Concept-driven multi-modality fusion for video search. IEEE Trans Circuits Syst Video Technol 21(1):62–73
11. Naphade M, Smith J, Tesic J, Chang SF, Hsu W, Kennedy L, Hauptmann A, Curtis J (2006) Large-scale concept ontology for multimedia. IEEE Multimed 13(3):86–91
12. Shirahama K, Uehara K (2011) Constructing and utilizing video ontology for accurate and fast retrieval. Int J Multimed Data Eng Manag (IJMDEM) 2(4):59–75
13. Zhu S, Wei X, Ngo C (2013) Error recovered hierarchical classification. In: Proceedings of the MM 2013, pp 697–700
14. Hauptmann A, Yan R, Lin WH, Christel M, Wactlar H (2007) Can high-level concepts fill the semantic gap in video retrieval? A case study with broadcast news. IEEE Trans Multimed 9(5):958–966
15. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. In: Proceedings of the CVPR 2009, pp 248–255
16. Kittur A, Chi EH, Suh B (2008) Crowdsourcing user studies with mechanical turk. In: Proceedings of the CHI 2008, pp 453–456
17. Ayache S, Quénot G (2008) Video corpus annotation using active learning. In: Proceedings of the ECIR 2008, pp 187–198
18. Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, Gool LV (2005) A comparison of affine region detectors. Int J Comput Vis 65(1–2):43–72
19. Lowe D (1999) Object recognition from local scale-invariant features. In: Proceedings of the ICCV 1999, pp 1150–1157
20. Bay H, Tuytelaars T, Gool L (2006) SURF: speeded up robust features. In: Proceedings of the ECCV 2006, pp 404–417
21. van de Sande KEA, Gevers T, Snoek CGM (2010) Evaluating color descriptors for object and scene recognition. IEEE Trans Pattern Anal Mach Intell 32(9):1582–1596
22. Csurka G, Bray C, Dance C, Fan L (2004) Visual categorization with bags of keypoints. In: Proceedings of the ECCV 2004 SLCV, pp 1–22
23. Inoue N, Shinoda K (2012) A fast and accurate video semantic-indexing system using fast MAP adaptation and GMM supervectors. IEEE Trans Multimed 14(4):1196–1205
24. Perronnin F, Dance C (2007) Fisher kernels on visual vocabularies for image categorization. In: Proceedings of the CVPR 2007, pp 1–8
25. Vapnik V (1998) Statistical learning theory. Wiley-Interscience, New York
26. Lin HT, Lin CJ, Weng RC (2007) A note on Platt's probabilistic outputs for support vector machines. Mach Learn 68(3):267–276
27. Smeaton AF, Over P, Kraaij W (2006) Evaluation campaigns and TRECVid. In: Proceedings of the MIR 2006, pp 321–330
28. The PASCAL Visual Object Classes Homepage. http://pascallin.ecs.soton.ac.uk/challenges/VOC/
29. ImageNet Large Scale Visual Recognition Competition (2013) (ILSVRC2013). http://www.image-net.org/challenges/LSVRC/2013/
30. Shirahama K, Uehara K (2012) Kobe university and Muroran institute of technology at TRECVID 2012 semantic indexing task. In: Proceedings of the TRECVID 2012, pp 239–247

31. Snoek CGM et al (2009) The MediaMill TRECVID 2009 semantic video search engine. In: Proceedings of the TRECVID 2009, pp 226–238
32. Natsev AP, Naphade MR, Tešić J (2005) Learning the semantics of multimedia queries and concepts from a small number of examples. In: Proceedings of the MM 2005, pp 598–607
33. Rasiwasia N, Moreno P, Vasconcelos N (2007) Bridging the gap: query by semantic example. IEEE Trans Multimed 9(5):923–938
34. Shafer G (1976) A mathematical theory of evidence. Princeton University Press, Princeton
35. Denoeux T (2013) Maximum likelihood estimation from uncertain data in the belief function framework. IEEE Trans Knowl Data Eng 25(1):119–130
36. Kanamori T, Hido S, Sugiyama M (2009) A least-squares approach to direct importance estimation. J Mach Learn Res 10(7):1391–1445
37. He H, Garcia E (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
38. Nowak E, Jurie F, Triggs B (2006) Sampling strategies for bag-of-features image classification. In: Proceedings of the ECCV 2006, pp 490–503
39. Snoek CGM, Worring M, Geusebroek JM, Koelma D, Seinstra F (2005) On the surplus value of semantic video analysis beyond the key frame. In: Proceedings of the ICME 2005, pp 386–389
40. Wang H, Klaser A, Schmid C, Liu CL (2011) Action recognition by dense trajectories. In: Proceedings of the CVPR 2011, pp 3169–3176
41. Peng Y et al (2009) PKU-ICST at TRECVID 2009: high level feature extraction and search. In: Proceedings of the TRECVID 2009
42. Aggarwal C, Yu P (2009) A survey of uncertain data algorithms and applications. IEEE Trans Knowl Data Eng 21(5):609–623
43. Bi J, Zhang T (2005) Support vector classification with input data uncertainty. In: Proceedings of the NIPS 2004, pp 161–168
44. Kriegel HP, Pfeifle M (2005) Density-based clustering of uncertain data. In: Proceedings of the KDD 2005, pp 672–677
45. Wang H, McClean S (2008) Deriving evidence theoretical functions in multivariate data spaces: a systematic approach. IEEE Trans Syst Man Cybern B Cybern 38(2):455–465
46. Aregui A, Denoeux T (2008) Constructing consonant belief functions from sample data using confidence sets of pignistic probabilities. Int J Approx Reason 49(3):575–594
47. Zribi M (2003) Parametric estimation of Dempster-Shafer belief functions. In: Proceedings of the ISIF 2003, pp 485–491
48. Benmokhtar R, Huet B (2008) Perplexity-based evidential neural network classifier fusion using MPEG-7 low-level visual features. In: Proceedings of the MIR 2008, pp 336–341
49. Wang X, Kankanhalli M (2010) Portfolio theory of multimedia fusion. In: Proceedings of the MM 2010, pp 723–726
50. Li X, Snoek CG (2009) Visual categorization with negative examples for free. In: Proceedings of the MM 2009, pp 661–664
51. Quattoni A, Wang S, Morency L, Collins M, Darrell T (2007) Hidden conditional random fields. IEEE Trans Pattern Anal Mach Intell 29(10):1848–1852

Chapter 13
Multimodal Fusion: Combining Visual and Textual Cues for Concept Detection in Video

Damianos Galanopoulos, Milan Dojchinovski, Krishna Chandramouli, Tomáš Kliegr and Vasileios Mezaris

Abstract Visual concept detection is one of the most active research areas in multimedia analysis. The goal of visual concept detection is to assign to each elementary temporal segment of a video, a confidence score for each target concept (e.g. forest, ocean, sky, etc.). The establishment of such associations between the video content and the concept labels is a key step toward semantics-based indexing, retrieval, and summarization of videos, as well as deeper analysis (e.g., video event detection). Due to its significance for the multimedia analysis community, concept detection is the topic of international benchmarking activities such as TRECVID. While video is typically a multi-modal signal composed of visual content, speech, audio, and possibly also subtitles, most research has so far focused on exploiting the visual modality. In this chapter, we introduce fusion and text analysis techniques for harnessing automatic speech recognition (ASR) transcripts or subtitles to improve the results of visual concept detection. Since the emphasis is on late fusion, the introduced algorithms for handling text and the fusion can be used in conjunction with standard algorithms for visual concept detection. We test our techniques on the TRECVID 2012 Semantic indexing (SIN) task dataset, which is made of more than 800 h of heterogeneous videos collected from Internet archives.

D. Galanopoulos (✉) · V. Mezaris
Centre for Research and Technology Hellas, Information Technologies Institute,
6th Km. Charilaou - Thermi Road, P.O. Box: 60361, 57001 Thermi-Thessaloniki, Greece
e-mail: dgalanop@iti.gr

V. Mezaris
e-mail: bmezaris@iti.gr

M. Dojchinovski
Web Engineering Group, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic
e-mail: milan.dojchinovski@fit.cvut.cz

T. Kliegr · M. Dojchinovski
Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, Prague, Czech Republic
e-mail: tomas.kliegr@vse.cz

K. Chandramouli
Division of Enterprise and Cloud Computing, VIT University, Vellore, India
e-mail: krishna.c@vit.ac.in

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_13

13.1 Introduction

Visual concept detection is one of the most active research areas in multimedia analysis. The goal of visual concept detection is to assign to each elementary temporal segment of a video, a confidence score for each target concept (e.g., forest, ocean, sky, etc.). Due to its significance for the multimedia analysis community, concept detection is the topic of international benchmarking activities such as TRECVID [21].

While video is typically a multi-modal signal composed of visual content, speech, audio, and possibly also subtitles, most research has so far focused on exploiting the visual modality. In this chapter, we will introduce fusion and text analysis techniques for harnessing automatic speech recognition (ASR) transcripts or subtitles to improve the results of visual concept detection. Since the emphasis is on late fusion, the introduced algorithms for handling text and the fusion can be used in conjunction with standard algorithms for visual concept detection.

This chapter is organized as follows. Section 13.2 introduces the visual concept detection task, recounting the most common algorithms used to process the visual modality. Section 13.3 gives a brief overview of algorithms for exploiting the textual content (ASR transcripts or subtitles) for concept detection and motivates the choice of the Explicit Semantic Analysis (ESA) method. The main focus of this chapter, the strategies for fusion of the textual and visual content, are covered in Sect. 13.4. Section 13.5 covers the experimental evaluation. The conclusion provides the discussion of results and suggestions for further work.

13.2 Visual Concept Detection

Concept detection using the video's visual cues typically follows the processing pipeline of Fig. 13.1. In the first step, the videos are segmented into shots using methods such as threshold-based approaches [3, 23] or more sophisticated statistical learning algorithms, e.g., SVM [5, 16] or Adaboost [34]. For each shot, a meaningful subset of the visual information is selected for further analysis, and visual low-level

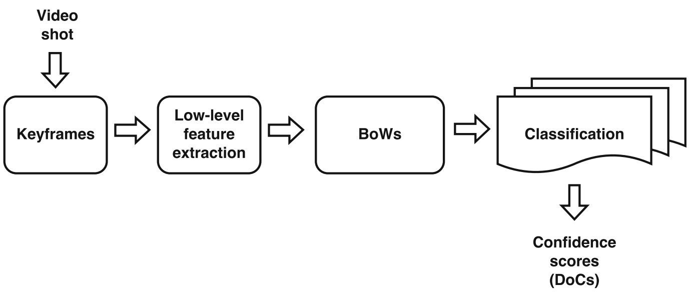


Fig. 13.1 The pipeline of a typical visual concept detection system

Table 13.1 The 25 employed shot representations

Shot representations	
Keyframe-based	12 representations, created by considering all possible combinations of 3 descriptors (SIFT, Opponent-SIFT, RGB-SIFT) × 2 sampling strategies (Dense, Harris-Laplace) × 2 BoW strategies (soft-, hard-assignment)
	1 representation based on color histograms
Tomograph-based	12 representations, created by considering all possible combinations of two types of video tomographs (horizontal, vertical) × three descriptors (SIFT, Opponent-SIFT, RGB-SIFT) × two BoW strategies (soft-, hard-assignment)

features such as SIFT [17] or SURF [2] are extracted from it. These features are used as input to a number of classifiers, which are based on machine learning techniques such as SVM [6], regression [1] or Bayesian networks [19]. Finally, the output of these classifiers is fused in order to produce the final visual concept detection score.

In our framework, a video is initially segmented into shots using the method of [29]. To represent a shot's visual content, one keyframe and two other 2D cross-selections of the video volume, termed tomographs [28], are selected and used for subsequent extraction of local image features. The latter local feature extraction procedure is based on a multitude of combinations of different interest point detectors (Harris Laplace corner detector [11] or dense sampling), interest point descriptors (SIFT [17], RGB-SIFT and Opponent SIFT [30]) and techniques for aggregating the local descriptors into a global image representation (Bag-of-Words using hard or soft-assignment, 3 × 1 pyramidal decomposition [13]). Overall, for each shot, 25 such combinations, each one resulting in the representation of the same shot in a different low-level feature space, are calculated (Table 13.1), as detailed in [20].

Subsequently, the necessary mappings between shot representations and each considered visual concept are established with the help of SVM classifiers. For each of these 25 different combinations, a set of five linear SVMs (LSVM) is trained, each using a subset of the available ground-truth annotated training data, following the Bag of Models (BoM) approach as in [18]. In this way, 125 LSVM classifiers in total (25 × 5) are trained for each concept. The output of each classifier is a Degree of Confidence (DoC) in the range [0, 1] for the corresponding concept. A late fusion strategy (calculating their arithmetic mean) is finally used for combining these 125 intermediate scores in a single DoC per concept.

13.3 Exploiting Textual Information for Concept Detection

The ASR transcripts constitute the most common explicit textual information that can be expected to be readily available with many videos. In this section, we will use them instead of visual features to perform concept detection in video. Similarly to the previous section, the concept detection task is treated as a soft classification problem: to each video shot we want to assign a confidence score for every target concept. Here, however, as input we use not only a set of videos (segmented to shots) and the set of target concepts, but also the video's ASR transcripts as well as the short textual description that accompanies every concept and fully explains it (this can be up to several sentences long).

This ASR-based video concept detection task is supported by two breeds of algorithms: text categorization and word similarity (relatedness) computation.

13.3.1 Text Categorization

The problem can be cast as a conventional document categorization problem, where the target concepts correspond to the classes and the ASR fragments are the documents to be categorized (Example 1). There is a large body of research on text categorization, for a review of selected approaches please refer to [26]. Many common algorithms are based on the bag-of-words representation, Term Frequency—Inverse Document Frequency (TF-IDF) and cosine matching [25].

Example 1 **ASR text**: "... looming clouds of smoke and fireballs were visible, possibly on the islands of the West a New Jersey Turnpike intention was brought down", **target classes**: *Explosion-Fire*, *Basketball*, *Car-Racing*

Here, for the textual modality, a vector representation with as many features as there are distinct words in all the descriptions of target concepts is constructed for each target concept. This feature space is then used to represent also the ASR transcripts as vectors. Finally, the confidence score for each pair of concept and ASR fragment is computed as the cosine of the angles between the respective vectors.

The problem with the application of text categorization on the visual concept detection is the sparsity of the input data. Unlike in the typical text categorization setting, both input texts (concept description and ASR fragments) tend to be very short. In the following, we give a brief review of prospective approaches for handling the textual modality. These approaches motivate our choice of the ESA algorithm.

13.3.2 Semantic Concept Mapping

In [12] we have introduced Semantic Concept Mapping (SCM) as a method for utilizing WordNet and Wikipedia to overcome the problem of sparseness. SCM maps the noun phrases representing the entities as well as the target classes to WordNet. Graph-based WordNet similarity measures are used to assign the closest class to the noun phrase. If a noun phrase does not match any WordNet concept, a *Targeted Hypernym Discovery* (THD) algorithm is executed. The THD algorithm extracts a hypernym from a Wikipedia article defining the noun phrase using lexico-syntactic patterns. This hypernym is then used to map the noun phrase to a WordNet synset, but it can also be perceived as the classification result by itself, resulting in an unsupervised classification system. A certain disadvantage of SCM is that since typically multiple entities (noun phrases) are contained in an ASR fragment, an additional fusion step is required to produce a single confidence score per concept and ASR fragment (Example 2).

Example 2 **ASR text**: "Maradona scores a goal!", **target classes**: athlete, writer. The identified objects are the entities "Maradona" and "goal". Since "Maradona" is not in WordNet, THD uses Wikipedia to map this word to (a WordNet-covered) "soccer player". Finally, the similarity between each identified entity and each class is computed, producing *sim*(Maradona, athlete), *sim*(Maradona, writer), *sim*(goal, athlete), *sim*(goal, writer). Finally, the results are fused to overall similarity *sim*("Maradona scores a goal", "athlete") and *sim*("Maradona scores a goal", "writer").

13.3.3 Explicit Semantic Analysis

Explicit Semantic Analysis (ESA) [8] is one of the new breed of algorithms for measuring semantic relatedness that are based purely on Wikipedia. In the ESA method [8], the input text T is represented as a weight vector, with positions corresponding to Wikipedia articles and the entries to the relevance of the corresponding concepts to text T.[1] Since ESA estimates relatedness between two text fragments (or two words), it can be easily adapted for text classification. Considering that each class has a textual description, we can use ESA to estimate the semantic similarity between the document in question and each class description.

While there are other Wikipedia-based algorithms, such as Wikipedia Link Measure (WLM) [32], these are mostly less suitable for the intended fusion setup, since they compute similarity only between individual words (or Wikipedia articles), while ESA naturally handles computation of similarity between words as well as texts. Until very recently, when it was overtaken by Temporal Semantic Analysis [24], ESA had best results (correlation with human judgment) on the standard WordSim353 dataset. Of all Wikipedia-based word similarity/relatedness algorithms, ESA has the highest amount of follow-up applied research in various areas of information retrieval, including cross-language information retrieval. In image and video processing, ESA was used in supporting the task of automatic image tagging [14] as well as video retrieval [31].

[1] Thus the name *Explicit* Semantic Analysis—due to the use of natural concepts (Wikipedia articles), the model is easy to explain to human users.

13.3.4 Generating Audio DoCs

For the generation of vectors from the audio modality in our framework, we have selected the ESA algorithm as justified in the previous subsection.[2] The input for ESA comprises the ASR fragment and each of the target concepts. The result of the computation—the relatedness value—is used as the degree of confidence.

Example 3 **ASR text**: "... looming clouds of smoke and fireballs were visible, possibly on the islands of the West a New Jersey Turnpike intention was brought down." **Concepts** with their textual description: *Car-Racing*—"Shot of scenes at car races", *Explosion-Fire*—"Shots of an explosion or a fire", and *Basketball*—"One or more people playing basketball." The text similarity vector between the ASR text and each individual concept is $\mathbf{sim}(ASRText, [car, fire, basketball]) = [0.04912, 0.0814, 0.0379]$.

In the above example (Example 3), computed relatedness values using the ESA algorithm between the fragment text and each concept description document will be: 0.04912 for the "Car-Racing", 0.0814 for the "Explosion-Fire" and 0.0379 for the "Basketball" concept. The highest confidence score is assigned to the concept "Explosion-Fire", followed by the concepts "Car-Racing" and "Basketball". The concept "Explosion-Fire" receives the highest confidence score because in its description there are same or similar concept(s) as in the ASR fragment (e.g., *smoke, fireballs*).

13.4 Multimodal Fusion for Improved Concept Detection

As a result of the previous procedures covered in Sects. 13.2 and 13.3, each video shot has two scores for each concept, one from each modality (i.e., visual and audio) that expresses the DoC to this concept. Since N concepts were taken under consideration for the evaluation test, the ith video shot is represented by two feature vectors $\mathbf{x}^i_m = [x^i_{1,m}, x^i_{2,m}, \ldots, x^i_{N,m}]$, where m is the modality index (in our case $m = \{V, A\}$, where V stands for *Visual* and A for *Audio*).

The target outcome of multimodal fusion is that the ith video shot is represented by a new feature vector $\mathbf{z}^i = [z^i_1, z^i_2, \ldots, z^i_N]$, where z^i_n is the combination result of $x^i_{n,V}$ and $x^i_{n,A}$ DoCs and n is the concept ID in the range of $[1, N]$.

13.4.1 Post-processing Audio DoCs

In order to improve on the results, we choose to post-process the audio DoCs. The motivation behind this is that the concept-based annotation of the shots that we want

[2] The ESAlib implementation obtained from http://ticcky.github.io/esalib/ with ESA background built from Wikipedia snapshot from 2005.

to generate refers to the visual information (i.e., describes what is visible in the shot). It is therefore reasonable to consider that a concept which appears in shot i is not necessarily included in the text information of this shot, but might be mentioned in the audio transcript or subtitle within a broaden time window, i.e., in a transcript temporally overlapping with one of the preceding or following shots $[i-\alpha, \ldots, i+\alpha]$.

To handle this issue, the following procedure was devised. Let us consider a video shot i with audio DoC for N concepts $\mathbf{x}_A^i = [x_{1,A}^i, x_{2,A}^i, \ldots, x_{N,A}^i]$. We introduce the $\mathbf{x}_{pA}^i$ DoC, which is the average of audio DoCs in a range of $\pm\alpha$ shots ($\mathbf{x}_{pA}^i = \langle \mathbf{x}_A^{i-\alpha}, \ldots, \mathbf{x}_A^i, \ldots, \mathbf{x}_A^{i+\alpha} \rangle$). So, the post-processed audio includes the information of $2\alpha + 1$ shots. These vectors are then used as audio results (DoCs) for shot i, instead of the original ones.

To illustrate the post-processing procedure, the following example is given (Example 4).

Example 4 **The audio DoC post-processing procedure**.
Let $\mathbf{x}_A^i = [0.0018, 0.0025, 0.0021]$ be the audio DoC vector (outcome of the process of Sect. 13.3) for the ith shot for $N = 3$. For a range of $\alpha = 2$ shots, we will have the following vectors, $\mathbf{x}_A^{i-2} = [0.0011, 0.0014, 0.002]$, $\mathbf{x}_A^{i-1} = [0.0015, 0.0008, 0.0023]$, $\mathbf{x}_A^{i+1}$=[0.0009, 0.0011, 0.0015], $\mathbf{x}_A^{i+2}$=[0.0016, 0.0024, 0.0014]. So, the $\mathbf{x}_{pA}^i$ will be the average of these five vectors. In the case of arithmetic averaging the final result will be $\mathbf{x}_{pA}^i = [0.0014, 0.0016, 0.0019]$.

13.4.2 Fusion Strategies

In order to find the best strategy to combine the visual and audio DoCs, three classes of late fusion techniques were originally considered: linear combination, meta-classification, and second level linear fusion.

To perform *linear combination*, three types of averaging were examined: arithmetic, geometric, and harmonic mean. Since it was observed that every DoC of the visual baseline ($\mathbf{x}_V$) is higher than the respective DoC from the audio baselines ($\mathbf{x}_A, \mathbf{x}_{pA}$), techniques such as choosing the maximum of individual scores, or voting, cannot be used directly, without some normalization step, since the result of the fusion would always be identical with the dominating visual baseline (Example 5).

Example 5 Let $\mathbf{x}_V^i = [0.1169, 0.1657, 0.07, 0.134]$ and $\mathbf{x}_A^i = [0.009, 0.01, 0.01, 0.008]$ be the visual and the audio baseline for the ith video shot. So, the new DoC after the fusion with arithmetic mean will be $\mathbf{z}_{\text{arith.}}^i = [0.063, 0.0878, 0.04, 0.071]$. If the vectors are fused with harmonic or geometric mean, the results will be $\mathbf{z}_{\text{harm.}}^i = [0.0167, 0.0189, 0.0175, 0.0151]$ and $\mathbf{z}_{\text{geom.}}^i = [0.0324, 0.0407, 0.0265, 0.0327]$ respectively. Each value of $\mathbf{z}^i$ is the new DoC of shot i for one of the four concepts.

Another class of techniques we considered for fusing the visual and audio DoCs, alternative to the linear combination discussed above, was *meta-classification* techniques [15]. Let $\mathbf{O}^i_{VA}$ denote the concatenation of the visual and audio baselines for the shot i, $\mathbf{O}^i_{VA} = [\mathbf{x}^i_V\ ,\ \mathbf{x}^i_A]$. This can be considered as a new representation of shot i. Consequently, a kernel SVM or a two-class logistic regression model were trained on the training portion of a properly partitioned [27] video shot dataset. In this way, we train N SVM or regression models, one for each concept. As each model corresponds to a concept, the output of each model $\mathbf{z}_n$ is in the range [0, 1]. This value is used as the new DoC for the corresponding concept. Thus, for each video shot we have N new DoCs $\mathbf{z}^i$.

The result of meta-classification can be considered as the fusion between concepts and it is a new classifier [10]. Empirically, fusing this classifier with the initial visual baseline classifier, which is a strong classifier, was shown in our experiments to further improve the results. So, *second level linear fusion* is used as an additional

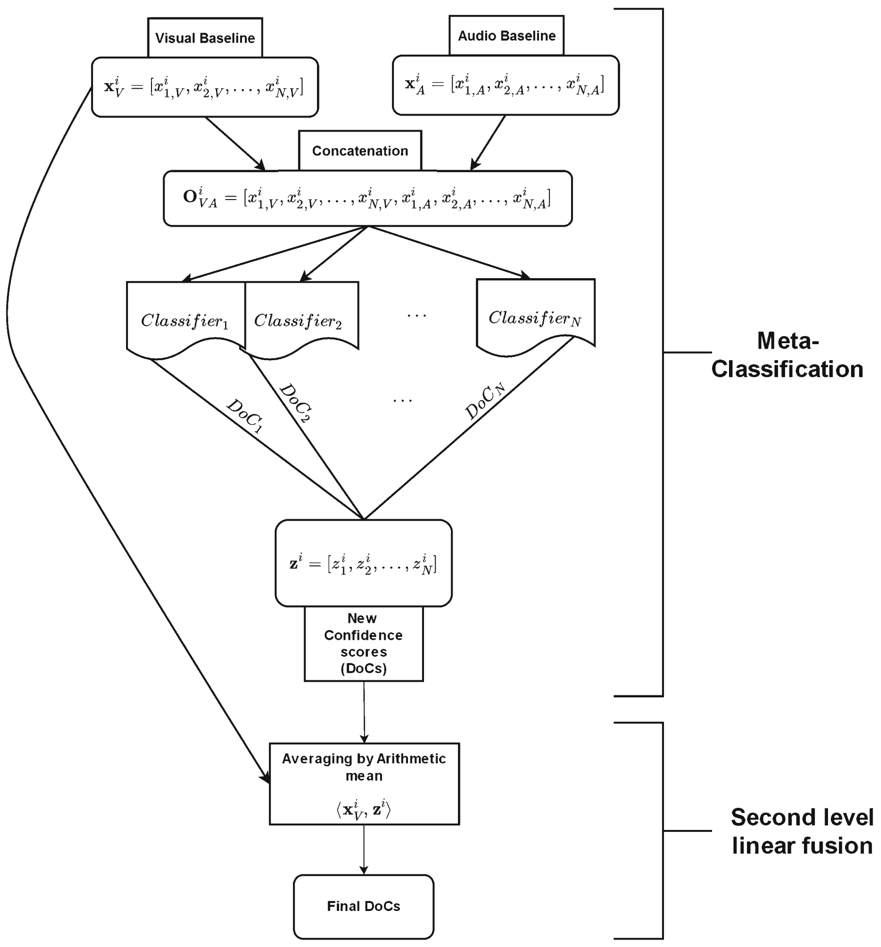

Visual Baseline
$\mathbf{x}^i_V = [x^i_{1,V}, x^i_{2,V}, \ldots, x^i_{N,V}]$
Audio Baseline
$\mathbf{x}^i_A = [x^i_{1,A}, x^i_{2,A}, \ldots, x^i_{N,A}]$
Concatenation
$\mathbf{O}^i_{VA} = [x^i_{1,V}, x^i_{2,V}, \ldots, x^i_{N,V}, x^i_{1,A}, x^i_{2,A}, \ldots, x^i_{N,A}]$
$Classifier_1$
$Classifier_2$
...
$Classifier_N$
DoC_1
DoC_2
...
DoC_N
$\mathbf{z}^i = [z^i_1, z^i_2, \ldots, z^i_N]$
New Confidence scores (DoCs)
Averaging by Arithmetic mean
$\langle \mathbf{x}^i_V, \mathbf{z}^i \rangle$
Final DoCs
Meta-Classification
Second level linear fusion

Fig. 13.2 Meta-classification and second level linear fusion pipeline

step to the meta-classification approach: the new feature vector $\mathbf{z}^i$ produced by meta-classification is further fused with the visual baseline $\mathbf{x}_V^i$ with arithmetic mean averaging in order to produce the final DoC. The meta-classification approach is visualized in Fig. 13.2 (and also explained in Example 6). The same procedure is followed for the combination of visual and post-processed audio baselines.

In order to explain the above procedure, a brief example is presented for $N = 4$ concepts (Example 6).

Example 6 In case of meta-classification, after concatenation of the baselines, we have the vector $\mathbf{O}_{VA}^i = [0.1169, 0.1657, 0.07, 0.134, 0.009, 0.01, 0.01, 0.008]$, which is the new representation of video shot i. The vector $\mathbf{O}_{VA}^i$ is the input to 4 trained SVMs, where their outputs are 4 new DoCs $\mathbf{z}^i = [0.04, 0.034, 0.02, 0.07]$. Finally, at the *second level linear fusion* stage, the $\mathbf{z}^i$ and $\mathbf{x}_V^i$ are averaged with arithmetic mean, to produce the final DoCs [0.0785, 0.0998, 0.0450, 0.1020].

13.5 Experiments and Results

13.5.1 Dataset

We test our framework on the TRECVID 2012 Semantic indexing (SIN) task dataset [22]. This dataset is made of 19,861 videos (>600 h) and 8,262 videos (>200 h) for training and testing, respectively. Its videos are short videos collected from Internet archives, and add up to more than 400,000 video shots in total.

The ASR data [9] provided for the purpose of the TRECVID competition contains the transcripts of the speech in the videos. Since not all videos include speech, TRECVID provides ASR files only for 14,507 training videos and 5,587 testing videos.

13.5.2 Experiment Setup

We apply our framework on 34 concepts (Table 13.2) of the TRECVID SIN task. Most concepts are defined with one sentence of text (e.g., "Shots of an airplane", "One or more people singing"), and the remaining few concepts have a somewhat longer description (about 2–5 sentences). The objective is to detect these concepts in non-annotated video shots.

Following the application of the unimodal concept detection techniques of Sects. 13.2 and 13.3, each video shot is represented by two feature vectors in the $\mathbb{R}^{34}$ space (one for the visual and one for the audio content), where each vector value is a DoC for the corresponding concept. We try to combine these two feature vectors, to improve the performance of our framework.

Table 13.2 TRECVID concepts used in this work

Concept ID	Label	Concept ID	Label
1	Airplane	2	Basketball
3	Bicycling	4	Boat ship
5	Boy	6	Bridges
7	Chair	8	Computers
9	Girl	10	Government leader
11	Greeting	12	Highway
13	Instrumental musician	14	Kitchen
15	Meeting	16	Motorcycle
17	Nighttime	18	Office
19	Press conference	20	Roadway junction
21	Singing	22	Sitting down
23	Stadium	24	Throwing
25	Baby	26	Fields
27	Forest	28	George Bush
29	Hill	30	Lakes
31	Military airplane	32	Oceans
33	Skier	34	Soldiers

As evaluation measure, we use the Extended Inferred Average Precision (XinfAP) [33] which is the same as what TRECVID 2012 SIN task uses to evaluate the performance of the participants. XinfAP is an extended version of infAP which is an approximate Average Precision (AP) measure, used instead of AP when the ground-truth annotation of a dataset is not complete.

13.5.3 Results

The first step was to find the optimum value α for the post-processing of the audio features. In Sect. 13.4.1, we introduced the value $\mathbf{x}^i_{pA}$, which is the average of $(2\alpha+1)$ shots. We test the concept detection system with the post-processed audio baseline for the values of $\alpha \in [4, 30]$, and also with the initial audio baseline ($\alpha = 0$). In this step we use DoCs only from audio and post-processed audio content and we do not perform any further step such as meta-classification or second level fusion. The results are displayed in Fig. 13.3 in terms of Mean XinfAP (MXinfAP). Three types of averaging were used: arithmetic, geometric, and harmonic.

The results clearly indicate that the averaging with arithmetic mean performs better than the other types. We decided to use the post-processed audio DoC, which was produced with the arithmetic average. Henceforth when $\mathbf{x}^i_{pA}$ is indicated, it denotes the arithmetic mean of the audio DoCs for $2\alpha + 1$ neighboring shots.

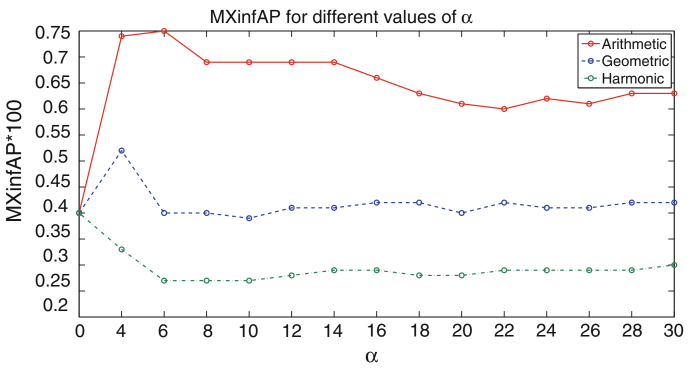


Fig. 13.3 MXinfAP for the different values of α

In the next stage, we try to combine the visual and audio DoCs. As was mentioned above (Sect. 13.4.2), three types of averaging were used in order to fuse the DoCs. In any case, by audio DoC we mean here the post-processed audio with arithmetic mean, and define α in a range of [4, 12] because of the good performance of this range of values in the previous experiment. In Fig. 13.4 we can see the results of fusion for various values of α (audio and visual results fused using the arithmetic mean). As expected, fusing the post-processed audio with the visual baseline (i.e., the visual concept detection method of Sect. 13.2) performs better than the fusion of the audio ($\alpha = 0$) and visual baselines.

In Table 13.3 we compare the overall performance of the fusion of visual and audio DoCs using different α values (Sect. 13.4.1), by averaging them with arithmetic mean. MXinfAP is used for quantifying the performance. We can see that the combination of visual and post processed audio with $\alpha = 6$ performs much better than the combination of the visual and audio baseline ($\alpha = 0$) and slightly better than the post-processed audio when values of α higher than six are used.

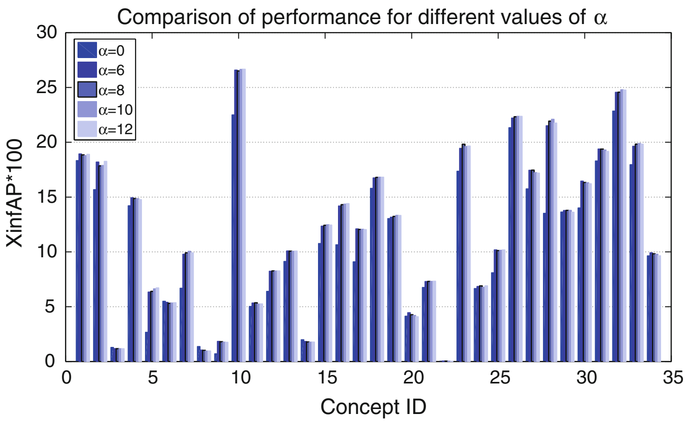


Fig. 13.4 XinfAP per concept for the fusion of the audio and visual baselines by calculating the arithmetic mean of the corresponding DoCs and using various values of α

Table 13.3 MXinfAP performance for the fusion of the audio and visual baselines with arithmetic mean and various vales of α

α value	MXinfAP
0	10.61
6	12.16
8	12.10
10	12.12
12	12.11

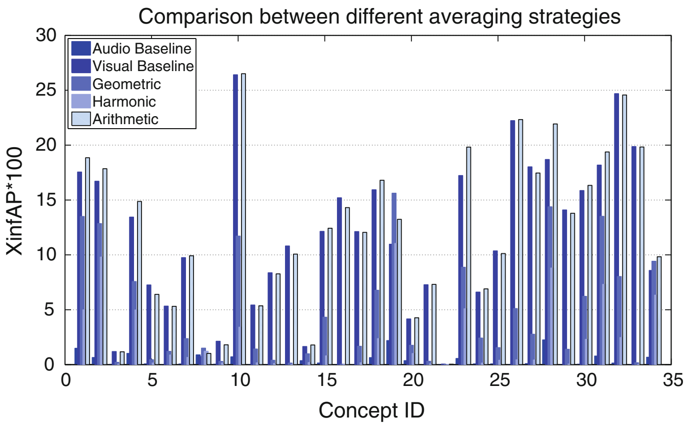


Fig. 13.5 XinfAP performance per concept, for audio-visual averaging strategies (arithmetic, geometric, and harmonic mean) and comparison with the audio and visual baselines

Table 13.4 MXinfAP performance for averaging fusion strategies with $\alpha = 6$ and comparison with the audio and visual baselines

Average strategy	MXinfAP
Visual baseline	11.75
Arithmetic mean	12.16
Harmonic mean	2.23
Geometric mean	4.68

Figure 13.5 compares the three different averaging strategies with fixed $\alpha = 6$ against the performance of the visual baseline. The MXinfAP of these three and the visual baseline can be found in Table 13.4. The averaging with arithmetic mean performs better than the other two averaging methods and, most importantly, gives an improvement compared to the visual baseline. In all experiments described so far, no meta-classification has been performed.

In the meta-classification approach, we try two different classification techniques, kernel SVM [4] and logistic regression [7]. For each classification technique, three different feature vectors were tested. As was mentioned in Sect. 13.4, $\mathbf{x}_V^i$ and $\mathbf{x}_A^i$ are the visual and audio baselines, $\mathbf{x}_{pA}^i$ the post-process audio baseline and $\mathbf{O}_{VA}^i = [\mathbf{x}_V^i, \mathbf{x}_A^i]$. So we have three feature vectors $\mathbf{x}_V^i$, $\mathbf{O}_{VA}^i$ and $\mathbf{O}_{VpA}^i$, where the latter is defined as $\mathbf{O}_{VpA}^i = [\mathbf{x}_V^i, \mathbf{x}_{pA}^i]$. These vectors are the inputs for 34 kernel SVM or logistic regression models, which are trained in order to learn the relations between the concepts and improve the evaluation results. After the training phase, the models were tested on the evaluation set and produced a new set of 34 DoCs for every video shot. In the second level linear fusion, these DoCs were fused with the visual baseline using arithmetic averaging and produced the final DoC for every video shot.

Figure 13.6 shows the performance of the meta-classification approach with SVM for different audio baselines ($\mathbf{O}_{VA}$, $\mathbf{O}_{VpA}$) compared with the visual baseline. It is clear from these results that there is an improvement for the majority of concepts. However, some concepts do not benefit from the post-processing step. For example, in concepts such as basketball (id = 2), boat ship (id = 4), chair (id = 7), fields (id = 26), etc. the performance of Visual+Audio_Meta is better than Visual+PostAudio_Meta.

In Table 13.5, the overall results of every classification method are shown. We notice that the SVM classification performs better when it takes as input the combination of visual and post-processed audio baselines ($\mathbf{O}_{VpA}$) (13.6 %), rather than the combination of visual and audio baselines ($\mathbf{O}_{VA}$) (12.4 %). In the second level linear fusion, there was a significant improvement of about 20.2 % on top of the meta-classification's performance and 36.6 % improvement in comparison to the visual baseline. In contrast, the performance of $\mathbf{O}_{VA}$ is improved by 19.8 and 34.7 % compared to the meta-classification and the visual baseline, respectively.

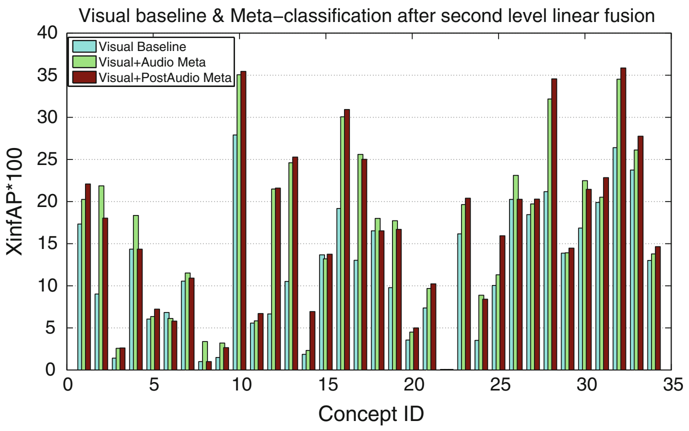


Fig. 13.6 XinfAP per concept for meta-classification for visual baseline, $\mathbf{O}_{VA}$ and $\mathbf{O}_{VpA}$ after the second level linear fusion

Table 13.5 MXinfAP performance for meta-classification fusion

	Kernel SVM			Logistic regression		
	$\mathbf{x}_V$	$\mathbf{O}_{VA}$	$\mathbf{O}_{VpA}$	$\mathbf{x}_V$	$\mathbf{O}_{VA}$	$\mathbf{O}_{VpA}$
Visual baseline	11.97	11.97	11.97	11.97	11.97	11.97
Meta-classification	12.00	13.46	**13.60**	12.24	12.33	12.54
Second level linear fusion	15.36	16.12	**16.35**	14.46	14.48	14.55

Using logistic regression as a classification method instead of SVM still resulted in an improvement compared to the baselines, but this improvement is lower than that is gained when using SVM. More specifically, the improvement from the visual baseline was 3 % for the $\mathbf{O}_{VA}$ and 4.7 % for the $\mathbf{O}_{VpA}$. After the second level linear fusion, the final improvement for the $\mathbf{O}_{VA}$ was 17.4 % from the meta-classification and 21 % from the visual baseline, and for $\mathbf{O}_{VpA}$, the improvement was 16 and 21.6 % from meta-classification and visual baseline performance respectively.

13.6 Conclusion

In this chapter, we examined if and how visual concept detection in video can be improved by effectively combining the visual information, which is typically used to this end, with textual information coming from the application of ASR techniques on the audio signal. To support this combination, we started with a short overview of concept detection using only visual features. We then proceeded with discussing audio-based concept detection, and particularly the pre-processing of the raw ASR results that is necessary for building a reasonably well-performing uni-modal audio-based classifier.

Subsequently, combining these visual and audio based concept detectors was the main focus, and to this end we considered and experimentally evaluated a number of strategies, achieving significant accuracy boosts over the uni-modal baselines, and thoroughly presented how different fusion strategies compare with each other.

Our future work includes introducing even stronger uni-modal classifiers in the proposed fusion frameworks, e.g., visual classifiers based on Fisher Vector encoding of the low-level features; examining the usefulness of introducing additional word relatedness/similarity algorithms, such as the WLM, in the textual information analysis pipeline; and taking into account the correlation between the concepts, which in uni-modal concept detection approaches has shown significant promise.

Acknowledgments This work was supported by the European Commission under contract FP7-287911 LinkedTV.

References

1. Bao L, Yu SI, Lan ZZ, Overwijk A, Jin Q, Langner B, Garbus M, Burger S, Metze F, Hauptmann A (2011) Informedia@ TRECVID 2011 multimedia event detection, semantic indexing. TRECVID compet 1:107–123
2. Bay H, Tuytelaars T, Van Gool L (2006) SURF: speeded up robust features. In: Computer vision-ECCV 2006. Springer, Heidelberg, pp 404–417
3. Cernekova Z, Pitas I, Nikou C (2006) Information theory-based shot cut/fade detection and video summarization. IEEE Trans Circuits Syst Video Technol 16(1):82–91
4. Chang CC, Lin CJ (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2:27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

5. Chavez GC, Precioso F, Cord M, Philipp-Foliguet S, Araujo AdA (2006) Shot boundary detection at TRECVID 2006. In: Proceedings of the TREC video retrieval evaluation, p 1–8
6. Delezoide B, Precioso F, Gosselin PH, Redi M, Mérialdo B, Granjon L, Pellerin D, Rombaut M, Jégou H, Vieux R et al (2011) IRIM at TRECVID 2011: semantic indexing and instance search. In: Notebook papers of the TREC video retrieval evaluation workshop (TRECVID)
7. Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ (2008) LIBLINEAR: a library for large linear classification. J Mach Learn Res 9:1871–1874
8. Gabrilovich E, Markovitch S (2007) Computing semantic relatedness using Wikipedia-based explicit semantic analysis. IJCAI 7:1606–1611
9. Gauvain JL, Lamel L, Adda G (2002) The LIMSI broadcast news transcription system. Speech Commun 37(1):89–108
10. Hamadi A, Mulhem P, Quénot G (2013) Conceptual feedback for semantic multimedia indexing. In: Proceedings of the 11th international workshop on content-based multimedia indexing (CBMI). IEEE, pp 53–58
11. Harris C, Stephens M (1988) A combined corner and edge detector. In: Alvey vision conference, vol 15. Manchester, p 50
12. Kliegr T, Chandramouli K, Nemrava J, Svatek V, Izquierdo E (2008) Combining image captions and visual analysis for image concept classification. In: Proceedings of the 9th international workshop on multimedia data mining: held in conjunction with the ACM SIGKDD 2008, MDM '08ACM, New York, pp 8–17
13. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, vol 2. IEEE, pp 2169–2178
14. Leong CW, Mihalcea R, Hassan S (2010) Text mining for automatic image tagging. In: Proceedings of the 23rd international conference on computational linguistics: posters. Association for Computational Linguistics, pp 647–655
15. Lin WH, Hauptmann A (2002) News video classification using SVM-based multimodal classifiers and combination strategies. In: Proceedings of the 10th ACM international conference on multimedia. ACM, pp 323–326
16. Liu C, Liu H, Jiang S, Huang Q, Zheng Y, Zhang W (2006) JDL at TRECVID 2006 shot boundary detection. In: TRECVID 2006 workshop
17. Lowe DG (1999) Object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE international conference on computer vision, vol 2. IEEE, pp 1150–1157
18. Markatopoulou F, Moumtzidou A, Tzelepis C, Avgerinakis K, Gkalelis N, Vrochidis S, Mezaris V, Kompatsiaris I (2013) ITI-CERTH participation to TRECVID 2013. In: Proceedings of TRECVID 2013 workshop. TRECVID 2013
19. Mittal A, Cheong LF (2004) Addressing the problems of Bayesian network classification of video using high-dimensional features. IEEE Trans Knowl Data Eng 16(2):230–244
20. Moumtzidou A, Gkalelis N, Sidiropoulos P, Dimopoulos M, Nikolopoulos S, Vrochidis S, Mezaris V, Kompatsiaris I (2012) ITI-CERTH participation to TRECVID 2012. In: Proceedings of TRECVID 2012 workshop. TRECVID 2012
21. Over P, Awad G, Michel M, Fiscus J, Sanders G, Kraaij W, Smeaton AF, Quénot G (2013) TRECVID 2013—an overview of the goals, tasks, data, evaluation mechanisms and metrics. In: Proceedings of TRECVID 2013. NIST
22. Over P, Awad G, Michel M, Fiscus J, Sanders G, Shaw B, Kraaij W, Smeaton AF, Quénot G (2012) TRECVID 2012—an overview of the goals, tasks, data, evaluation mechanisms and metrics. In: Proceedings of TRECVID 2012. NIST
23. Quénot G, Moraru D, Besacier L (2003) Clips at TRECVID: shot boundary detection and feature detection. In: TRECVID 2003 workshop notebook papers. Citeseer
24. Radinsky K, Agichtein E, Gabrilovich E, Markovitch S (2011) A word at a time: computing word relatedness using temporal semantic analysis. In: Proceedings of the 20th international conference on world wide web. ACM, pp 337–346
25. Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. Inf Process Manag 24(5):513–523

26. Sebastiani F (2002) Machine learning in automated text categorization. ACM Comput Surv (CSUR) 34(1):1–47
27. Sechidis K, Tsoumakas G, Vlahavas I (2011) On the stratification of multi-label data. In: Machine learning and knowledge discovery in databases. Springer, Berlin, pp 145–158
28. Sidiropoulos P, Mezaris V, Kompatsiaris I (2013) Enhancing video concept detection with the use of tomographs. In: Proceedings of the 20th IEEE international conference on image processing (ICIP), pp 3991–3995
29. Tsamoura E, Mezaris V, Kompatsiaris I (2008) Gradual transition detection using color coherence and other criteria in a video shot meta-segmentation framework. In: Proceedings of the 15th IEEE international conference on image processing (ICIP), pp 45–48
30. Van De Sande KE, Gevers T, Snoek CG (2010) Evaluating color descriptors for object and scene recognition. IEEE Trans Pattern Anal Mach Intell 32(9):1582–1596
31. Wan KW, Yau WY, Roy S (2013) Metadata enrichment for news video retrieval: a graph-based propagation approach. In: Proceedings of the 21st ACM international conference on multimedia. ACM, pp 373–376
32. Witten I, Milne D (2008) An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: Proceeding of AAAI workshop on Wikipedia and artificial intelligence: an evolving synergy. AAAI Press, Chicago, pp 25–30
33. Yilmaz E, Kanoulas E, Aslam JA (2008) A simple and efficient sampling method for estimating AP and NDCG. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval. ACM, pp 603–610
34. Zhao ZC, Cai AN (2006) Shot boundary detection algorithm in compressed domain based on adaboost and fuzzy theory. In: Advances in natural computation. Springer, Berlin, pp 617–626

Chapter 14
Mining Videos for Features that Drive Attention

Farhan Baluch and Laurent Itti

Abstract Certain features of a video capture human attention and this can be measured by recording eye movements of a viewer. Using this technique combined with extraction of various types of features from video frames, one can begin to understand what features of a video may drive attention. In this chapter we define and assess different types of feature channels that can be computed from video frames, and compare the output of these channels to human eye movements. This provides us with a measure of how well a particular feature of a video can drive attention. We then examine several types of channel combinations and learn a set of weightings of features that can best explain human eye movements. A linear combination of features with high weighting on motion and color channels was most predictive of eye movements on a public dataset.

14.1 Background

Videos are made up of a stream of running frames each of which has a unique set of spatial and textural features that evolve over time. Each video therefore presents a viewer with a large amount of information to process. The human visual system has limited capacity and evolution has incorporated several mechanisms into the visual processing systems of animals and humans to allow only the most important and behaviorally relevant information to be passed on for further processing. The first stage is the limited amount of high resolution area in the eye, i.e., the fovea. When we want to focus on a different spatial region of a video we make an eye movement, also

F. Baluch (✉)
Research and Development Group, Opera Solutions,
12230 El Camino Real, San Diego, CA 92130, USA
e-mail: farhanbaluch@gmail.com

L. Itti
Department of Computer Science, Psychology & Neuroscience Graduate Program,
University of Southern California,
3641 Watt Way, HNB 10, Los Angeles, CA 90089, USA
e-mail: itti@usc.edu

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_14

known as a saccade, to bring the area of interest into alignment with the fovea. Within the fovea too, attention can focus our perception on features that either interest us or that are intrinsically visually conspicuous or *salient*. The former is termed top-down attention and the latter bottom-up attention [2].

In this chapter we discuss in detail how a video may be decomposed into a set of features that coarsely map to features computed in the human brain. Using this neuromorphic approach to predicting eye movements, we can begin to understand what features of a video attract human attention. This understanding is not only essential for answering the scientific question of how attention works in the brain, but, in addition, this understanding can also help us build better computer vision systems and furthermore has other applications. A model that can successfully predict where humans allocate attention can be used to enhance marketing displays [5], provide a means to intelligently compress videos [21], speed up object recognition [32], and also improve video based security systems [37].

14.1.1 Human Attention and Eye Movements

The study of eye movements as a measure of human attention dates back to over 100 years ago; however, it was Yarbus [35] who first reported the manner in which eye movements may reveal the goal of an observer and those items in the scene that are determined to be interesting. Often objects that are of interest functionally (or cognitively) also inherently possess the visual attributes that attract attention, i.e., these objects are considered salient both visually and psychologically [7]. Therefore, studying the eye movements of human subjects while they view static or dynamic scenes can reveal a lot about the cognitive processes underlying human visual perception. Despite the proliferation of tools now available to study the brain, eye movements provide a simple, quick and non-invasive method to probing human attention using experimental means. Eye movements are monitored using infra-red or high definition cameras that can detect and continually track the pupil of the eye. By calibrating the change in position of the pupil with certain calibration points on a screen, a mapping or transformation can be used to translate the movement detected by the eye tracker to screen coordinates. Using this method, an observer's eye movement traces can be overlaid on the image or video being presented, thereby providing a means of locating the observer's attentional allocation to the scene.

Features of the visual world attract human attention and gaze. There is necessarily a relationship between the visual world and a particular human or animal's behavioral goals that results in eye movements and shifting of attention. The study of the origin of an eye movement has been the subject of numerous studies and is a very actively debated and active area of research [2, 22]. Broadly, however, the cause of an eye movement or attention shift is categorized as *bottom-up* if it is a result of the visual appeal of the sensory input (e.g., orientation of attention towards a light that suddenly blinks brightly), and *top-down* if it is a result of the behavioral goal of the human or animal in question (e.g., a volitional shift of gaze to the left when nothing in the visual environment has changed). While this distinction helps us model and understand attention, the separation of the two purported sources (i.e., top-down and bottom-up) is a very challenging question in neuroscience. Since the onset of visual experience, a human or animal begins to form a subjective percept which, depending on experience, may force certain stimuli to appear a certain way that may be different from another individual's percept. Subjective experience and perception therefore can challenge the existence of a "normative" visual percept, and, therefore, make it very difficult to separate bottom-up and top-down influences on attention [2, 6, 9].

When modeling the human processes of attention, eye movements serve as the empirical evidence used to validate and quantify the quality of model. Any model of visual attention serves to indicate with faithfulness the likelihood of a human observer allocating attention to certain salient parts of the scene, i.e., a model generates a saliency map. Similar to the manner in which eye movements can be overlaid on an image, these eye movement traces can also be overlaid on a saliency map generated by a model. In this manner, we can find models that have an output that closely corresponds with human eye movements. Furthermore, we can use the deviation between the model output and the human eye movements to construct a cost function that can be optimized to fit parameters of new models developed.

14.1.2 Models of Human Attention

The development of saliency models lies at the interface of understanding human perception and developing visual intelligence in machines. In several domains, engineers have built systems that mimic or are inspired by biological systems. Biologically-inspired computer vision has a similar goal. In particular, modeling of attention has been an area of interest with numerous publications dedicated to the topic over the years [4]. Modeling attention is equivalent to computing the most conspicuous or salient regions of an image, that are likely to drive attention and, as a result, elicit an orientation of gaze towards the location in the image. Two approaches can be taken to building a model that can best explain a human viewers' eye movements. In the first approach, the functioning of the human visual system can be studied and modeled to arrive at a computational model of human attention, several models take this approach [14]. The second approach is to examine the patches of an image that are fixated by human observers and understand their features to build a dictionary of patches that are likely to elicit eye movements [17, 26]. In this chapter, we describe in detail a model that follows the first approach and attempts to arrive at a model based on the functioning and anatomy of the visual systems in biological organisms.

The Itti and Koch [16] model of salience has been widely successful [14] in accounting for overt allocation of attention in natural images and has become a benchmark for comparing other models. The model builds on previous theories [18,

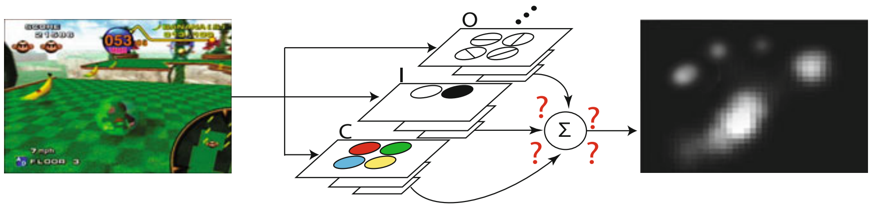


Fig. 14.1 Computation of features and saliency maps from video frames. Multi-scale features are computed from each video frame e.g. color, intensity, orientation etc. The feature maps are then combined generally using a simple linear combination to form a conspicuity maps also known as a saliency map. This example shows a video frame from a recording of a video game and the resulting saliency map for this frame after linear combination of features

27] of how attention is allocated by computing individual feature maps and then combining these maps into a single conspicuity map, where salient locations in the map attract attention. The general framework of this model consists of a decomposition of an image into various feature maps, by computing filter based features. Each computed feature forms a channel in the model that can be added or removed from the final saliency computation. Examples of these features include intensity contrast, motion energy, color opponent contrast, etc. Numerous such feature channels can be computed, and, since the development of the original model, a large number of channels have been added based on neuroscience discoveries of mechanisms of vision in the brain as well as useful features based on computer vision. Figure 14.1 illustrates the manner in which an image is decomposed into a set of features computed at multiple scales and then finally combined to form a saliency map. The saliency map can be viewed as an attention probability map that assigns high probability to regions of the image that are inherently interesting or likely to elicit human attention. The figure shows the color (C), intensity (I) and orientation (O) channels [16]. In a similar manner, several other channels can be computed and these have been listed in Table 14.1.

Each channel from this large set may contribute toward the salience of a location in the image and, therefore, the potential to elicit a gaze shift from a human. Each channel outputs a feature map that consists of pixels corresponding to the image. Each pixel in the feature map indicates the energy that the feature in question contributes at that location. In the standard implementation, feature maps output from all channels are linearly summed to form a final saliency map. This saliency map, after some normalization, serves as a probability map that consists of the same number of pixels as the input image and the value at each pixel indicates the likelihood of that pixel eliciting an attention orientation towards it by a human viewer. There are several strategies to combining the features maps into a final saliency map [15] and this continues to be an active area of research. In this chapter we will also focus on methods to combine feature maps and build a saliency map that maximizes the probability of predicting human gaze.

Table 14.1 List of feature channels

Channel name	Abbrev	Refs	Description
Color	C	[16]	Double-opponent color center-surround, for red-green and blue-yellow contrasts
Flicker	F	[12]	Flicker center-surround channel based on frame by frame differences
Multi-color band	G	[24]	Multi-color band channel with *N* Gaussian bands spanning the hue axis
H2SV	H	[23]	A variant over the HSV color space
Intensity	I	[16]	Intensity center-surround channel
DKL Color	J	[36]	A biologically-inspired color model
Skin hue	K	[31]	Skin hue detector
L-junction	L	[20]	Channel tuned to L-shaped corner edges
Motion	M	[12, 31]	Motion channel based on frame by frame differences
Intensity-band	N	[24]	Intensity channel with *N* Gaussian bands spanning the intensity axis
Orientation	O	[16]	Gabor-based orientation channel with *N* orientations
CIELab Color	Q	[10]	Color channel using the CIE L*a*b* color model
Pedestrian	R	[31]	Pedestrian channel based on simple template matching for humans
Single-opp. color	S	[24]	Composite of single-opponent color center-surround computed separately in the red, green, blue and yellow color bands
T-junction	T	[20]	Detector tuned to T-shaped edge junctions
Foreground	U	[13]	Foreground/background detection channel
Contour	W	[23, 25]	Elongated contour detection channel
X-junction	X	[20]	Detector tuned to X-shaped crossings of edges

In addition to the specific references listed below, papers [12, 16, 20, 25, 28] provide summary descriptions of collections of channels, and [13] provides reference source code implementation

14.2 Experimental Study of Attention

To evaluate a model of attention we need to obtain evidence of correspondence between the output of the model, i.e., its prediction of attention allocation within a scene, and human attention allocation. As discussed above, one means of measuring human attention allocation is by examining human eye movements using an eye tracker. Typically in experiments a specific set of stimuli is chosen and displayed on the screen. Study participants are given instructions on how to observe the scenes. Instructions can make a large difference on eye movements, in particular different types of instructions can emphasize either bottom-up or top-down aspects of the scene. For example, asking subjects to look for a yellow road sign in scenes may influence their eye movements spatially towards expected locations of road signs (spatial bias) and also may influence them to fixate on items that are yellow (feature bias). On the other hand, providing minimal instructions and asking subjects to watch and enjoy the scenes may emphasize bottom-up aspects of attention allocation by recording eye movements based on scene changes. While efforts can be made to emphasize bottom-up aspects of a scene there is no way to completely eliminate the influence of top-down aspects such as the viewers' personal bias and preferences.

14.2.1 Methods

We will discuss a study where three females and five males aged 23–32 with normal or corrected-to-normal vision were recruited. This data set, including both the videos as well as the recorded eye movement traces, are available openly to the public through the CRCNS program [11] for exploration. All subjects were USC students or staff members. Subjects gave written consent under a protocol approved by the Institutional Review Board and were paid for participating in the study. The stimuli for this study consisted of 50 video clips between 6 and 90 s each shown at 30 fps. A total of 46,000 video frames and 25 min of total video time. The videos contain a mix of indoor and outdoor scenes including park scenes, crowds, rooftop bars, TV news, sports, commercials, and video game footage. Figure 14.2 shows an example of these stimuli. The stimuli were presented on a monitor at 640 $\times$ 480 resolution running at 60 Hz. An ISCAN RK-464 eye tracker was used to track the subjects' eyes at 240 Hz. A nine point calibration was performed every five clips.

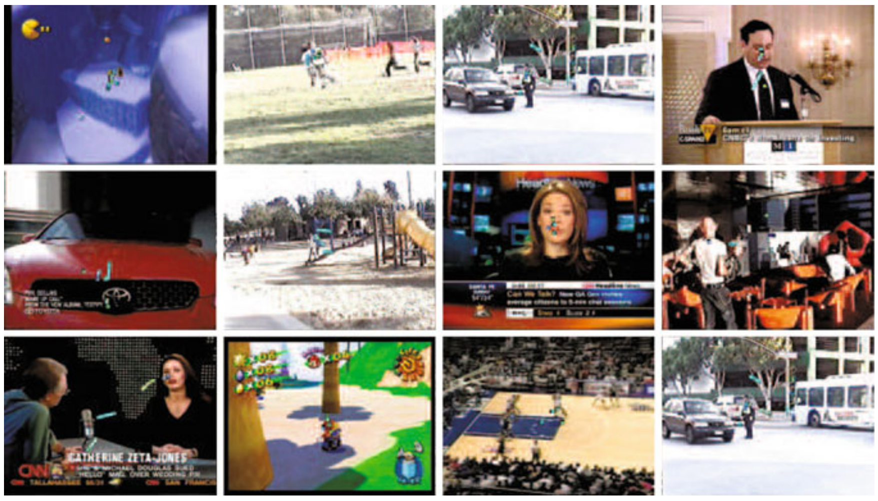


Fig. 14.2 Sample frames from video stimuli consisting of videos of different scenes including video game, TV adverts, outdoor and indoor scenes

Subjects were seated in a comfortable chair and asked to view the clips while their eyes were tracked. The instructions to the subjects were: "Follow the main actors and actions, try to understand overall what happens in each clip. We will ask you questions about the main contents. Do not worry about details". This instruction aimed to emphasize the bottom up component of the visual input being presented to the subjects. If they were asked to look for anything specific this would introduce a heavy top-down component and subjects' eye movements would reflect their own search strategies more than the inherent ability of the stimulus to draw attention. The goal of our modeling effort is to model bottom up or purely sensory components of the environment that can explain attentional shifts and allocation. Therefore subjects are instructed to focus more on the general scene rather than any specific targets.

As described earlier, the eye movements can be overlaid on the images being displayed in a post-processing step and in this manner we can observe the viewer's location of gaze on the scene and thus infer attentional allocation. The eye traces recorded during the viewing of the stimuli by the subjects were parsed into saccades based on a threshold of velocity as described before [1]. A total of 11,430 saccades were extracted and analyzed. Using the saliency model, we were able to extract feature maps and saliency maps using different combination rules. We then sampled these feature and saliency maps at the saccade endpoints to look for correlations between gaze location and saliency/feature values.

14.2.2 The Inter-observer Model and AUC Metric

To set an upper bound for the performance of models we built an inter-observer model. To build this model we grouped together the eye movements of all the subjects at each video frame and added them into a map consisting of all zeros and ones at locations of eye movement end points as shown in Fig. 14.3. A Gaussian centered at the location of each saccade endpoint or eye movement was defined with radius 5 pixels and applied to the map. This generated smooth "salience" maps defining the output of an inter-observer model. Since we know that each human observer will be different and we do not expect all to be the same we build this map as an average location of where we expect humans to fixate in a scene. The expectation is that a group of humans should predict the eye movements of a new observer who was not in the set of observers used to build the inter-observer model. To assess the quality of the model, however, we need a metric.

To quantify the performance of a particular model in predicting gaze, we use an ROC (receiver operating characteristics) like measure called area under the curve (AUC). This measure is computed by plotting the values generated in the models map at saccade end points against the values at 100 random locations on the map [3, 29]. Once these samples are drawn we can slide a threshold of salience and ask what percentage of human versus random locations were selected by the model at this threshold. A good model would result in a larger number of human fixated locations containing high values of and few random locations. The plot serves as an ROC curve

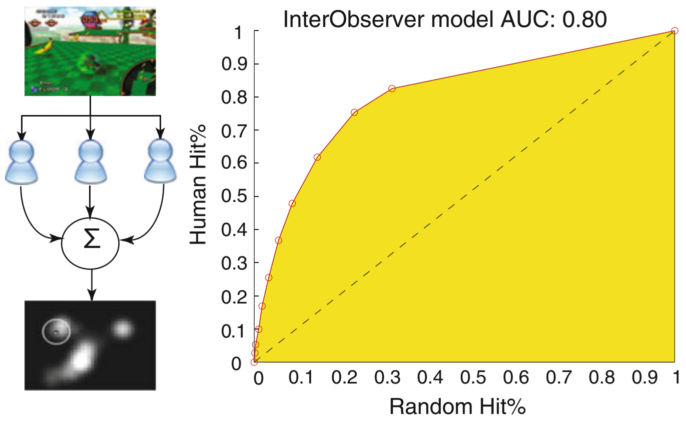


Fig. 14.3 Inter-observer model. *Left* shows a schematic of how the inter-observer model saliency maps were generated by pooling together eye movements from all subjects and then applying a gaussian at the saccade end points. *Right* shows the ROC curve by predicting saccades based on the inter-observer model. This *curve* was computed by computing inter-observer maps from seven subjects and then predicting the saccades of the one left out subject

and the area under this curve gives a measure of the quality of the model in question. A value of 1 indicates a model that completely accounts for saccade allocation while a value of 0.5 indicates a model that is no better than chance at predicting the location of gaze. All models we discuss in later sections will be gauged using this metric. Note that the theoretical maximum AUC of 1 is not achievable with a generic model that is not tailored to each particular individual, because all humans do not always agree, hence a single model cannot perfectly capture attention allocation of every single human.

Calculating the AUC metric for the inter-observer model, we obtain a very high AUC score of 0.80, indicating high (though not perfect) inter-observer agreement. This AUC score was significantly higher than all individual channels computed as well as various trained and untrained models as we examine in later sections. The inter-observer model, therefore sets the upper bound on the performance of the models. Intuitively, we do not expect a computational model of attention to be any better (or as good) at predicting human attention than a model constructed from the eye movements of a group of human observers.

14.3 Analysis of Feature Contributions

To understand the manner in which features interact to guide attention we decomposed each video frame into a set of feature maps that when combined would provide a saliency map [14] as discussed in earlier sections. Each so called channel provided a single feature map for each video frame. The channels computed were color, intensity, orientation, flicker, motion, and several others including complex junction channels

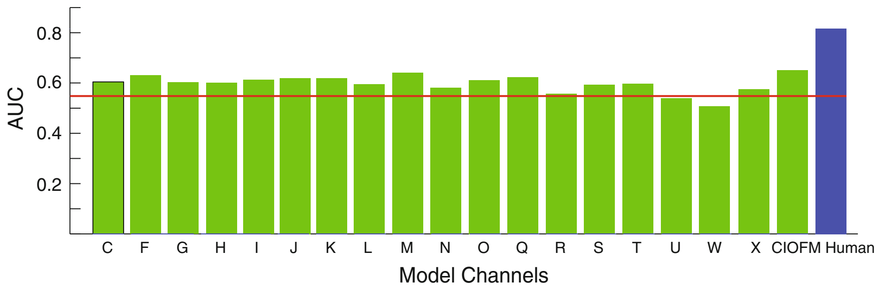


Fig. 14.4 Individual channel AUC. Each bar represents the performance of models built form individual channels in predicting gaze. See Table 14.1 for channel descriptions. CIOFM represents a linear combination of the C, I, O, F and M channels without any weighting. The bar labeled *Human* represents the inter-observer model. The *red line* indicates chance level i.e. AUC = 0.5

as listed in Table 14.1. We first analyzed the performance of each of these channels individually at predicting gaze. This is done by computing a feature map or channel on each video frame and then applying the AUC metric to test for performance. The lower bound for AUC is 0.5 i.e. a model is at chance at predicting whether a location will receive human attention or not, while the upper bound is set by the inter-observer model.

Figure 14.4 plots the AUC scores for individual channels as well as the inter-observer model. It is clear that individual channels fall short of the AUC score obtained from the inter-observer model. As expected humans are able to predict the attention allocation of other humans better than models of attention. A simple linear combination of the color (C), intensity (I), orientation (O), flicker (F) and motion (M) channels results in a model that performs reasonably well at predicting human attention allocation in scenes. In the rest of our discussion we focus on methods of finding highly predictive combinations of features using linear and non-linear combinations. We focus on the C, I, O, F, M channels because those have been historically prevalent, and we examine how different combinations of these would provide differences in gaze prediction.

When comparing features, we found that the motion channel was most predictive among the five analyzed channels (C, I, O, F and M). Figure 14.5 plots a histogram of the number of locations versus saliency value assigned by the model. One set of bars indicate the number of random locations that were assigned a certain saliency value while another set indicates the number of locations that were targets of human attention/saccades assigned that same level of saliency. The inset ROC curve in Fig. 14.5 plots is used to compute the AUC score which for the motion channel is 0.64, a reasonably good score.

Since the videos contain a significant amount of motion and the instruction to the subjects was to follow the main actors and actions it is intuitive that the motion channel is predictive of locations where subjects make saccades. While the inter-observer map sets the absolute upper bound on models, the motion channel with its

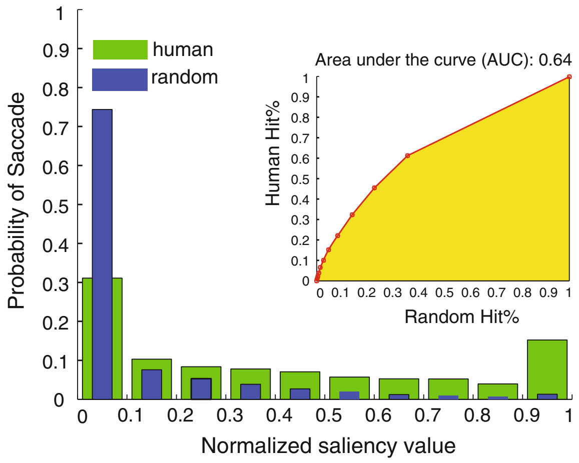


Fig. 14.5 Performance of the motion channel. Histogram shows probability of saccade (*green*) or random location hit (*blue*) towards locations of different saliency values from 0 to 1. The histogram is for saliency of the motion channel. Inset is an ROC curve computed by sliding a threshold along saliency values. The *yellow area* shows the area over which the AUC score is computed

high AUC score also sets a benchmark for other channels and models that combine channels.

14.4 Results

14.4.1 Linear Model with Trained Weights

To test the prediction that is driven by a weighted linear combination of features we trained a linear model to predict saccades by optimizing an objective function defined by the AUC cost. We used a genetic algorithm to find the optimal combination of weights for five features C, I, O, F and M. A genetic algorithm approach was used to enable comparison of this model with the larger optimization of a model with 20 features as discussed below. We started with random weights and enforced a constraint of allowing a weight to vary between 0 and 1. A population of 100 candidates was used and each individual consisted of five values of weights for each of the features. At each iteration each individual provided five weights for features which were used to build a saliency model and output a final saliency map. This map was used to compute the AUC score and determine how well the model did at predicting human gaze. Each individual's AUC score was computed and this was used as the fitness value for this candidate. Standard mutation and cross-over operators were used to breed new individuals. The results of the optimization with five features are shown in Fig. 14.6.

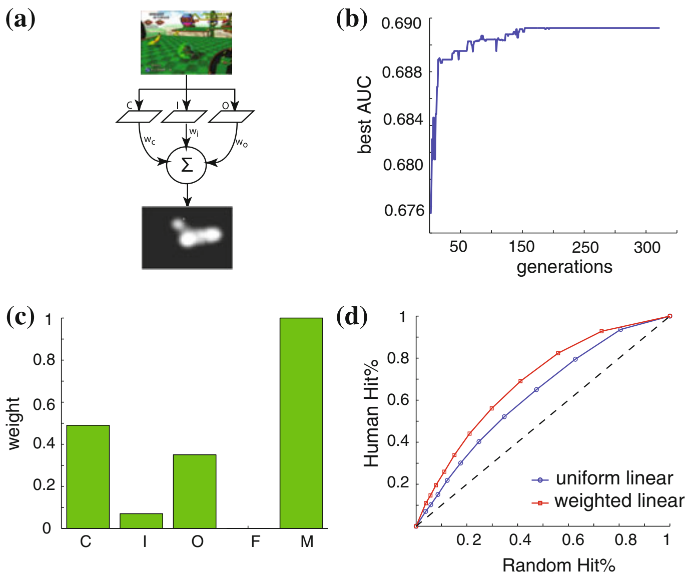
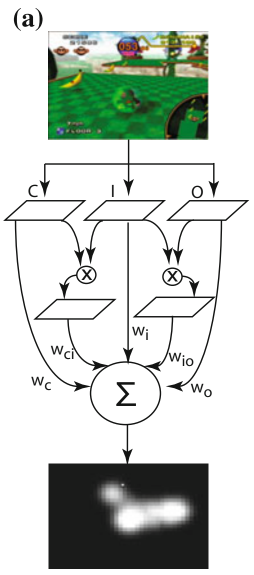


Fig. 14.6 Linear combination with learned weights. **a** Multi-scale features are computed from each video frame e.g. color, intensity, orientation etc. The feature maps are then combined generally using a simple linear combination to form a conspicuity maps also known as a saliency map. **b** The evolution of the best AUC of the population of individuals used in the genetic algorithm optimization. **c** The final weights learned by the genetic algorithm. It can be seen that motion has the highest weight at the end of learning. **d** A comparison of ROC curves between a linear model with uniform weights versus a linear model with learned weights

The genetic algorithm converged on a solution in about 200 generations. The fittest individuals had an average AUC value of 0.69. As predicted, the weight for the motion channel was the highest. The color and orientation channels also show a significant contribution in predicting salience. The weight for the flicker channel was very low, probably because this channel is highly correlated with the motion channel [36], and hence the optimization algorithm discarded it as redundant. Our results are in line with studies that have found that color and motion are among the top features that attract gaze [1, 34]. The weighted linear model with first order features performed significantly better than the uniform linear model that combined the features with uniform weights. As discussed below this turns out to be the most predictive model among the ones analyzed in this study.

14.4.2 Second Order Feature Interaction Model

To study the effect of non-linear interactions we generated second order features by point-to-point multiplication of each of the five features studied (CIOFM). This



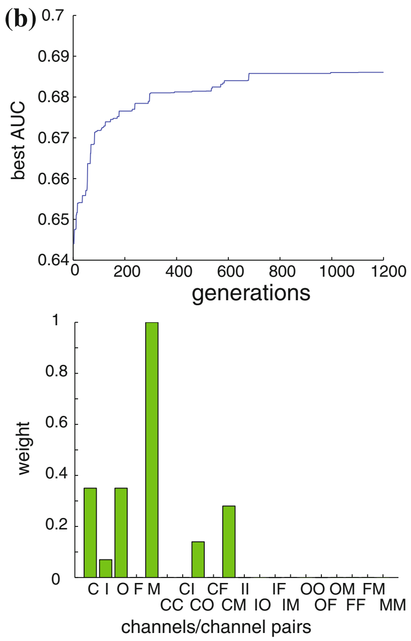


Fig. 14.7 Inter-observer model. **a** The manner in which features were combined to form second order terms and were then linearly combined. 20 weights were learned by the genetic algorithm. **b** *Top* shows the evolution of AUC scores as a function of generation of the population, the score converges around 800 generations. *Bottom* shows the weights for each term

generated a total of 20 features including combinations such as CO (color orientation combination), CI (color intensity interaction) etc. Once again we used the genetic algorithm approach to search for the parameters for this model. In this case we had 20 features to learn and therefore this was a much larger optimization problem which took longer to converge on a solution. Figure 14.7 shows the manner in which the genetic algorithm learned the best weights and converged to a solution after about 800 generations. This is significantly longer than the linear weighted model with five terms as would be expected since a much higher dimensional space (20) was explored in this experiment.

The results from this experiment are surprising in that second-order complex features do not help boost performance and the genetic algorithm converges to a solution that is similar to the linear case with only first order terms. Motion again is the strongest feature. The best AUC score 0.69 is similar to the model with only first order term. Second order features therefore did not improve the score and while one would expect this additional interaction information to help predict eye movements better, the principle of parsimony compels us to consider the first order model the better one. This is somewhat consistent with evidence from the physiology of the visual system in that there is a very small number of cells that might be tuned to second order combinations of features. While there is evidence of hierarchy of

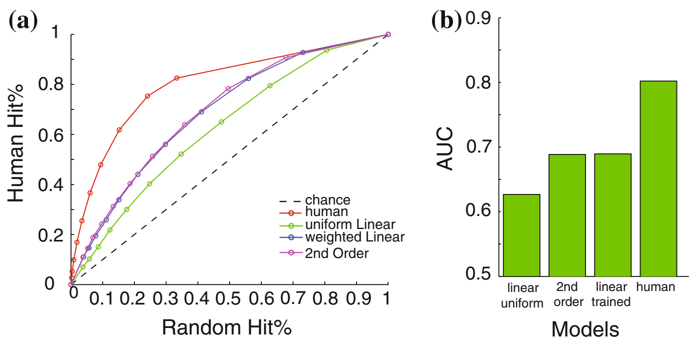


Fig. 14.8 Model comparison. **a** The ROC *curves* for the different models computed. Chance is represented by the *dashed black line* the *red line* labeled *human* in the legend represents the inter-observer model which performs the best. The uniform linear model represents the model using CIOFM features in a simple linear combination with uniform weights. The weighted linear model is the one that was trained using a genetic algorithm but consists only of linear terms. The 2nd order model uses both first order and 2nd order terms to define feature combinations. **b** The AUC computed from the *curves* in **a**. As can be seen the second order model does not perform any better than the linear model

features building up to a single percept, neurons tuned to combinations of features within our set are limited to color-orientation and color-motion cells [19, 30]. Even in lower brain areas like the superior colliculus, a key structure that enables the mechanisms of attention, there exist cells that are responsive to motion and even color [8, 33].

14.4.3 Model Comparison

We compared the performance of all the models studied (Fig. 14.8), i.e., linear model with uniform weights, linear model with learned weights, trained model with 2nd order terms and the human inter-observer model. The linear model with trained weights performs the best, but the inter-observer model has the highest performance in predicting human attention. This shows that a linear model with no higher level knowledge of the semantic content of the scene performs the best when compared to other models.

It is important and interesting to note that the model that included both linear and second-order terms performed no better than a model that consisted of linear terms but these terms were weighted. The weights were learned by using a genetic algorithm that maximized the AUC score. These results suggest that a biologically inspired model of attention that combines features in a linear manner to form a saliency map is likely to be closely related to mechanisms of attention in the brain that give rise to the observed eye movements.

14.5 Conclusions

In this chapter we processed and decomposed videos into several features and then searched for a good combination of features that can predict human attention allocation. Human attention consists of a volitional top-down component and an image driven bottom-up component. We presented a study that focused on the bottom-up aspects of attention. By recording of human observers as they watched natural videos we established a means to validate various models explored.

A linear combination of features was sufficient to provide prediction of human gaze, and second-order interactions of these features did not help performance. Therefore salience of a region in an image is determined through a linear combination of features and we can account for almost 70 % of the variance through a weighted linear model. Top-down attention then may act by providing the weights that were learned by our genetic algorithm.

While the linear combination model did reasonably well in predicting human gaze, an inter-observer model built from the eye movements of several observers outperformed the linear model. Humans are therefore better at predicting the eye movements of each other when compared to such a model of saliency. There is much further research to be done to both elucidate the mechanisms of attention in humans as well as build models that can mine videos for features that drive attention.

Acknowledgments This work was supported by the National Science Foundation (grant numbers CCF-1317433), the Office of Naval Research (N00014-13-1-0563), and the Army Research Office (W911NF-12-1-0433). The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

1. Baluch F, Itti L (2010) Training top-down attention improves performance on a triple-conjunction search task. PLoS One 5(2):e9127
2. Baluch F, Itti L (2011) Mechanisms of top-down attention. Trends Neurosci 34(4):210–240
3. Berg DJ, Boehnke SE, Marino RA, Munoz DP, Itti L (2009) Free viewing of dynamic stimuli by humans and monkeys. J Vis 9 5(19):1–15
4. Borji A, Itti L (2013) State-of-the-art in visual attention modeling. IEEE Trans Pattern Anal Mach Intell 35(1):185–207
5. Chiang A-YD, Berg D, Itti L (2011) Saliency, memory, and attention capture in marketing. J Vis 11(11):493–493
6. Connor CE, Egeth HE, Yantis S (2004) Visual attention: bottom-up versus top-down. Curr Biol 14(19):R850–R852
7. Elazary L, Itti L (2008) Interesting objects are visually salient. J Vis 8(3):3
8. Fecteau J, Bell A, Munoz D (2004) Neural correlates of the automatic and goal-driven biases in orienting spatial attention. J Neurophysiol 92(3):1728–1737
9. Gilbert C, Sigman M (2007) Brain states: top-down influences in sensory processing. Neuron 54(5):677–696

10. Grant WS, Itti L (2012) Saliency mapping enhanced by symmetry from local phase. In: Proceedings of IEEE international conference on image processing (ICIP). Florida, pp 653–656
11. Itti L (2008) Crcns data sharing: eye movements during free-viewing of natural videos. In: Collaborative research in computational neuroscience annual meeting. California
12. Itti L, Dhavale N, Pighin F (2003) Realistic avatar eye and head animation using a neurobiological model of visual attention. In: Bosacchi B, Fogel DB, Bezdek JC (eds) Proceedings of SPIE 48th annual international symposium on optical science and technology, vol 5200. SPIE Press, Bellingham, pp 64–78
13. Itti L et al (1998) The ilab neuromorphic vision C++ toolkit (INVT), http://ilab.usc.edu/toolkit
14. Itti L, Koch C (2001) Computational modelling of visual attention. Nat Rev Neurosci 2(3): 194–203
15. Itti L, Koch C (2001) Feature combination strategies for saliency-based visual attention systems. J Electron Imaging 10(1):161–169
16. Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Mach Intell 20(11):1254–1259
17. Kienzle W, Franz MO, Schölkopf B, Wichmann FA (2009) Center-surround patterns emerge as optimal predictors for human saccade targets. J Vis 9(5):7
18. Koch C, Ullman S (1985) Shifts in selective visual attention: towards the underlying neural circuitry. Hum Neurobiol 4(4):219–227
19. Koene AR, Zhaoping L (2007) Feature-specific interactions in salience from combined feature contrasts: evidence for a bottom-up saliency map in v1. J Vis 7(7):6
20. Li Z, Itti L (2011) Saliency and gist features for target detection in satellite images. IEEE Trans Image Process 20(7):2017–2029
21. Li Z, Qin S, Itti L (2011) Visual attention guided bit allocation in video compression. Image Vis Comput 29(1):1–14
22. Moore T (2006) The neurobiology of visual attention: finding sources. Curr Opin Neurobiol 16(2):159–165
23. Mundhenk TN, Itti L (2005) Computational modeling and exploration of contour integration for visual saliency. Biolo Cybern 93(3):188–212
24. Navalpakkam V, Itti L (2006) An integrated model of top-down and bottom-up attention for optimal object detection. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR). New York, pp 2049–2056
25. Peters RJ, Iyer A, Itti L, Koch C (2005) Components of bottom-up gaze allocation in natural images. Vis Res 45(8):2397–2416
26. Rajashekar U, Bovik AC, Cormack LK (2006) Visual search in noise: revealing the influence of structural cues by gaze-contingent classification image analysis. J Vis 6(4):7
27. Treisman A, Gelade G (1980) A feature-integration theory of attention. Cogn Psychol 12(1): 97–136
28. Tseng P, Cameron IGM, Pari G, Reynolds JN, Munoz DP, Itti L (2013) High-throughput classification of clinical populations from natural viewing eye movements. J Neurol 260: 275–284
29. Tseng P, Carmi R, Cameron IGM, Munoz D, Itti L (2009) Quantifying center bias of observers in free viewing of dynamic natural scenes. J Vis 9 7(4):1–16
30. Ts'o D, Gilbert CD (1988) The organization of chromatic and spatial interactions in the primate striate cortex. J Neurosci 8(5):1712–1727
31. Walther D (2006) Interactions of visual attention and object recognition: Computational modeling, algorithms, and psychophysics. PhD thesis, California Institute of Technology
32. Walther D, Itti L, Riesenhuber M, Poggio T, Koch C (2002) Attentional selection for object recognition—a gentle way. In: Biologically motivated computer vision. Springer, pp 472–479
33. White BJ, Boehnke SE, Marino RA, Itti L, Munoz DP (2009) Color-related signals in the primate superior colliculus. J Neurosci 29(39):12159–12166

34. Wolfe JM, Horowitz TS (2004) What attributes guide the deployment of visual attention and how do they do it? Nat Rev Neurosci 5(6):495–501
35. Yarbus AL, Haigh B, Rigss LA (1967) Eye Mov Vis. Plenum Press, New York
36. Yoshida M, Itti L, Berg DJ, Ikeda T, Kato R, Takaura K, White BJ, Munoz DP, Isa T (2012) Residual attention guidance in blindsight monkeys watching complex natural scenes. Curr Biol 22(15):1429–1434
37. Yubing T, Cheikh FA, Guraya FFE, Konik H, Trémeau A (2011) A spatiotemporal saliency model for video surveillance. Cogn Comput 3(1):241–263

Chapter 15
Exposing Image Tampering with the Same Quantization Matrix

Qingzhong Liu, Andrew H. Sung, Zhongxue Chen and Lei Chen

Abstract Image tampering, being readily facilitated and proliferated by today's digital techniques, is increasingly causing problems regarding the authenticity of images. As the most popular multimedia data, JPEG images can be easily tampered without leaving any clues; therefore, JPEG-based forensics, including the detection of double compression, interpolation, rotation, etc., has become an active research topic in multimedia forensics. Nevertheless, the interesting issue of detecting image tampering and its related operations by using the same quantization matrix has not been fully investigated. Aiming to detect such forgery manipulations under the same quantization matrix, we propose a detection method by using shift-recompression-based reshuffle characteristic features. Learning classifiers are applied to evaluating the efficacy. Our experimental results indicate that the approach is indeed highly effective in detecting image tampering and relevant manipulations with the same quantization matrix.

15.1 Introduction

While being widely used, transmitted and enjoyed, digital multimedia can be easily manipulated without leaving a clue. In recent years, multimedia forensics has emerged as a new discipline as it has important applications in law, crime

Q. Liu (✉) · L. Chen
Department of Computer Science, Sam Houston State University,
Huntsville, TX77341, USA
e-mail: liu@shsu.edu

L. Chen
e-mail: chen@shsu.edu

A.H. Sung
School of Computing, The University of Southern Mississippi,
Hattiesburg, MS 39406, USA
e-mail: Andrew.Sung@usm.edu

Z. Chen
Department of Epidemiology and Biostatistics, Indiana University Bloomington,
Bloomington, IN 47405, USA
e-mail: zc3@indiana.edu

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_15

investigations, and national security, etc. In multimedia forensics, steganalysis and forgery detection are two important, interrelated topics. While many promising and effective steganalysis methods have been proposed and several steganographic systems have been successfully steg-analyzed [12, 19–22, 24–29, 31, 32, 35, 42], it seems that the research on forgery detection has fallen behind.

In digital multimedia data, JPEG is one of the most popular compression standards. While we enjoy huge volumes of JPEG images, our traditional confidence in the integrity of the media via our eyes and ears has been greatly undermined since doctored pictures, video clips, and audio streams are easily manipulated. For instance, a state-run newspaper in Egypt published a doctored picture, attempting to create the illusion that its country's president was leading the group in Middle East peace talks in 2010 at Washington DC [16].

Generally, tampering manipulation in digital media involves several basic operations such as image resize, rotation, splicing, double compression; and the detection of these fundamental manipulations and relevant forgery has been studied extensively [2–10, 13, 17, 24, 30, 33, 34, 36–40]. Incidentally, double JPEG compression is a very common manipulation: while one decodes the bit stream of a JPEG image and implements some manipulation in spatial domain, and then compresses the modified image back into JPEG format, if the new quantization matrix is different from the one used by the original JPEG image, the modified JPEG image is processed by "double JPEG compression." Although double JPEG compression does not by itself prove malicious or unlawful tampering, it provides an evidence of image manipulation. The detection of double JPEG compression has been well studied [4, 24, 31]. However, if the original image sources are encoded with the same quantization matrix and the doctored images are also encoded with the same quantization matrix, the detection of such forgery becomes much more challenging. The existing methods for exposing double JPEG compressions are not effective in detecting the forgery with the same quantization matrix.

Although the detection of the forgery with the same quantization matrix is challenging, some efforts have already been focused on it. Huang et al. presented a method to detect double JPEG compression with the same quantization matrix, unfortunately it cannot tell us the double-compressed JPEG image is composited or not [17]. Luo et al. designed a set of block artifact characteristics matrix features (BACM) to detect the JPEG images once cropped and recompressed [33]. Chen and Hsu analyzed the periodicity of compression artifacts for tampering detection [3]. Both methods are impressive for the detection of cropping and recompression with different quantization matrices, however, they are not effective in detecting the cropping and recompression with the same quantization matrix, as shown by the results in [3]. To our knowledge, all existing methods do not work well to detect doctored images with the recompression by using the same quantization matrix, which was once used to encode the original images.

In this chapter, a shifted-recompression-based approach is introduced to detect the image tampering with the same quantization matrix. We introduce related study in Sect. 15.2, describe our proposed shift-recompression-based approach and experiments in Sect. 15.3, and conclude in Sect. 15.4.

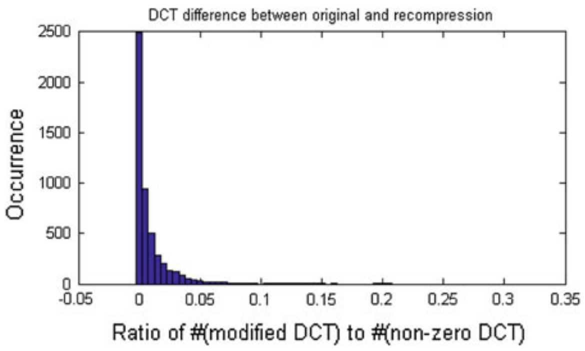


Fig. 15.1 Statistics of the modification after double recompression with the same compression matrix by using the image database, a total of 5,000 images [24, 27]

15.2 Related Study

To detect double JPEG compression with the same quantization matrix, Huang et al. designed a method based on the observation that in the process of recompressing a JPEG image with the same quantization matrix over and over again, the number of different JPEG coefficients between the two consecutive versions will monotonically decrease in general. There are, however, serious flaws with the method. Firstly, the observation is far from being always true. For example, by using the image database in the references [24, 27, 32], there are about 20 % JPEG images without any modification of the quantized DCT coefficients between the first compression and the second compression by using the same quantization matrix. If we consider the ratio of the number of modified quantized DCT coefficients to the number of nonzero DCT coefficients among the 5,000 images, most of these values are zeros or very close to zeros, as shown in Fig. 15.1. In other words, the foundation of the detection method designed by Huang et al. [17] may not be statistically sound. Secondly, the detection cannot tell us whether the doubly compressed images were tampered or not.

To detect the copy-paste tampering wherein a patch from a source is pasted onto a target, Luo et al. developed a set of features based on artifact characteristics matrix (BACM) [33]. It claims that the BACM shows regular symmetrical shape for the original JPEG images. However, the regular symmetrical property of the BACM is destroyed, while the images are tampered and resaved as JPEG images. The authors of [33] exploited the BACM property and designed representation features from the BACM. In their experiments, 2,256 singly compressed JPEG images were obtained at each given quality factor QF_2. Tampered images were obtained by converting original TIFF images into JPEG images with a random quality factor QF_1, then cropping and resaving it with the given quality factor QF_2. In our view, however, the detection is actually about misaligned double compression with different compression matrices.

Here we briefly introduce the construction of BACM matrix that is based on the assumption that the pixel difference across different DCT blocks will be different from that inside of the DCT blocks. The pixel difference within a block and spanning

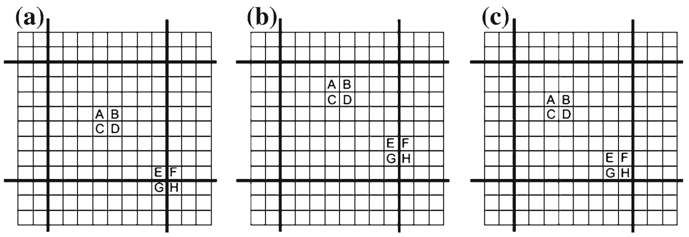


Fig. 15.2 Coordinate (x, y) and the positions of *A*–*H*

across a block boundary is given by

$$Z'_{(x,y)} = |A + D - B - C|\,; Z''_{(x,y)} = |E + H - F - G|\,. \tag{15.1}$$

Figure 15.2 shows the position of the pixels A, B, C, D, E, F, G and H, and the coordinate of A, (x, y), in each block. The coordinates of A to H in each block change according to the coordinate of A.

Let $H^{I}_{(x,y)}$and $H^{II}_{(x,y)}$denote the histograms of $Z'_{(x,y)}$and $Z''_{(x,y)}$, the energy K of the difference between $H^{I}_{(x,y)}$and $H^{II}_{(x,y)}$with the value n $(n = 0, 1, \ldots, 510)$ is calculated by

$$K^{(n)}_{(x,y)} = \left| H^{I}_{(x,y)}(n) - H^{II}_{(x,y)}(n) \right|. \tag{15.2}$$

The average of $K_{(x,y)}$ is denoted as $M_{(x,y)}$,

$$M_{(x,y)} = \frac{\sum_n K_{(x,y)}(n)}{255 \times 2 + 1}. \tag{15.3}$$

Lastly, we normalize the matrix $M_{(x,y)}$, and the normalized matrix is called blocking artifact characteristics matrix (BACM).

The BACM features are extracted by the following steps:

1. Crop the 7×7 block from the BACM and divide the 7×7 block to 7 nonoverlapping parts: R_1, R_2, R_3, R_4, Horizontal direction H, vertical direction V, and the center point C, as shown by Fig. 15.3.
2. The following 14 features are extracted from these 7 nonoverlapping regions.
   (a) The symmetry of H and V around the center point C, a subtotal of 2 features;
   (b) The symmetry of the four flat region R_1, R_2, R_3, and R_4 around H, V, and C, a subtotal of 6 features;
   (c) The percentage of the center point C occupying the region R_1, R_2, R_3, R_4, V, and H respectively, a subtotal of 6 features.

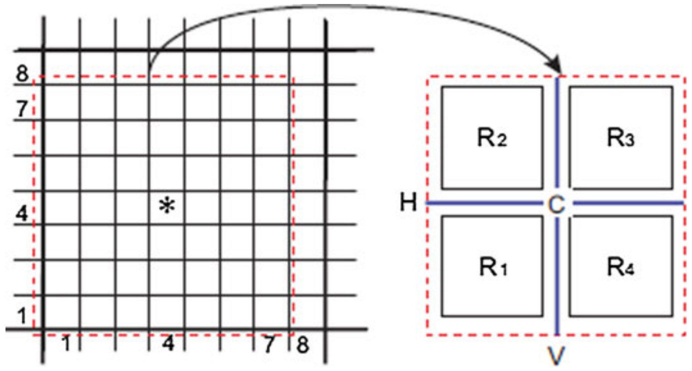


Fig. 15.3 BACM is cropped and then divided into 7 nonoverlapping parts for BACM feature extraction

The symmetry feature is the sum of energy of differences between the values in the matrix. The details are given in the Ref. [33]. For example, the symmetry of H around C is: $|M(1,4) - M(7,4)| + |M(2,4) - M(6,4)| + |M(3,4) - M(5,4)|$.

15.3 Shift-Recompression-Based Approach

15.3.1 Misaligned Cropping and Recompression

To prevent a forgery manipulation on JPEG images from being detected, a crafty forger may try to avoid double JPEG compression during the manipulation since the detection of JPEG double compression has been well studied with satisfactory results. It is not difficult for a forger to obtain the two source JPEG images with the same compression matrix. During the tampering, source JPEG bit streams are decoded to spatial domain first, and manipulation takes place in spatial domain. The doctored image is recompressed to JPEG format by using the same quantization matrix that was once used by the source images.

We describe the manipulation operations as follows: Source images S_1 and S_2 are encoded into JPEG format using the same quantization matrix. To create a forgery from S_1 and S_2, both source images are decoded into spatial domain, a region of interest R_1 from S_1 is copied and pasted to S_2. The modified S_2 is recompressed to JPEG format by using the same quantization matrix.

As shown by Fig. 15.4, the original region R_1 consists of several 8×8 JPEG-compression blocks r_{ij} in S_1. Assuming region R_1 is randomly pasted in S_2, then the original JPEG-compression blocks r_{ij} will be reshuffled with the neighboring 8×8 blocks (misaligned cropping and recompression) at a high probability ($63/64 = 98.4\,\%$) as new 8×8 blocks; if the block r_{ij} will not be reshuffled with the neighboring

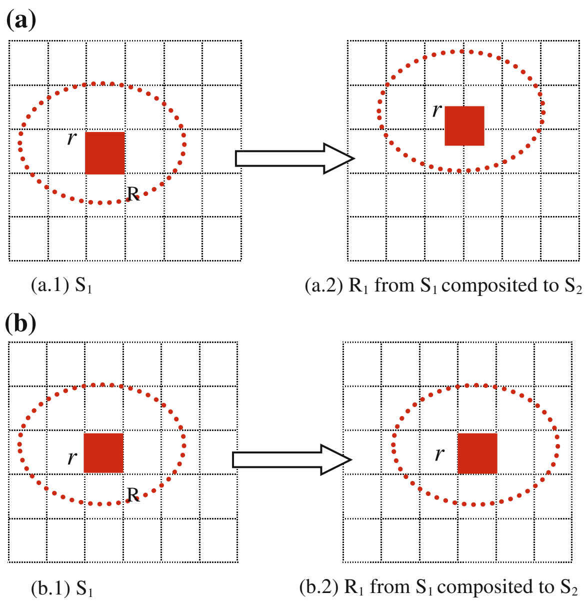

(a)
r
R
(a.1) S_1
r
(a.2) R_1 from S_1 composited to S_2
(b)
r
R
(b.1) S_1
r
(b.2) R_1 from S_1 composited to S_2

Fig. 15.4 Two types of JPEG-based composition. Misaligned cropping and recompression (**a**) occurs at a high probability. **a** Misaligned cropping and recompression. **b** Aligned cropping and recompression

8 × 8 blocks, it will be recompressed by itself as an entire 8 × 8 block (aligned cropping and recompression), such manipulation hits the probability of 1/64.

If S_1 and S_2 are encoded by using different quantization matrices, or if S_1 and S_2 are encoded with the same quantization matrix but the recompression uses different quantization matrix, then double compression takes place and we may reveal such a forgery by detecting the double compression. However, if S_1, S_2, and the composited image are all encoded with the same quantization matrix, the detection of such a compositing manipulation may not be easy.

15.3.2 A Shift-Recompression-Based Approach

A shift-recompression-based algorithm was proposed to detect misaligned cropping and recompression with the same quantization matrix in JPEG images [24], as described below.

We surmise that the reshuffle, shown in Fig. 15.4a, will leave clues for us to detect such manipulation behaviors in the final doctored JPEG image, although double JPEG compression has been avoided. Accordingly, we design a shift-recompression-based algorithm to identify the inconsistency of block artifacts due to the reshuffling, and finally identify the forged area in encoded JPEG formats. The feature exaction is proceeded according to the following steps:

SRSC feature extraction algorithm [24]

1. Decode an JPEG image under examination to spatial domain, which is denoted by matrix $S(i, j)(i = 1, 2, \ldots, M; j = 1, 2 \ldots, N)$;
2. Shift the matrix $S(i, j)$ by d_1 rows and d_2 columns in the spatial domain, $(d_1, d_2) \in \{(0, 1), \ldots, (0, 7), (1, 0), \ldots, (7, 7)\}$ and generate a shifted spatial image $S'(d_1, d_2)$, $S'(d_1, d_2) = S(i - d_1, j - d_2)(i = d_1 + 1, d_1 + 2, \ldots, M; j = d_2 + 1, q + d_2, \ldots, N)$, then compress the shifted spatial image $S'(d_1, d_2)$ to JPEG format at the same quantization matrix;
3. Decode the shifted JPEG image to spatial domain, denoted by a matrix $S''(d_1, d_2)$;
4. Calculate the difference $D(d_1, d_2) = S'(d_1, d_2) - S''(d_1, d_2)$;
5. **S**hift-recompression based **R**e**S**huffle **C**haracteristic features (**SRSC**) on the region of interest R, $\mathbf{SRSC}_R$ are defined by:

$$SRSC_R(d_1, d_2) = \frac{\sum |D_R(d_1, d_2)|}{\sum \left|S'_R(d_1, d_2)\right|}, \tag{15.4}$$

where $(d_1, d_2) \in \{(0, 1), \ldots, (0, 7), (1, 0), \ldots, (7, 7)\}$. There are a total of 63 features.

If an image was cropped or misaligned by p rows and q columns, mod $(p, 8) \neq 0$ or mod $(q, 8) \neq 0$, $0 \leq p \leq 8$, $0 \leq q \leq 8$, and then recompressed by using the same quantization matrix that was used for the original JPEG image, we expect that the SRSC features will be distinct due to the misalignment, and the values of p and q can be determined by the SRSC features. The example shown in Fig. 15.5 confirms our conjecture and preliminarily validates our algorithm. Figure 15.5a shows an original JPEG image and Fig. 15.5b shows a cropped and recompressed image ($p = 4$ and $q = 4$). The SRSC features from the original image (a) and modified image (b) are shown in Fig. 15.5c, d. The circle highlights the major differences of SRSC features between the original image and manipulated one. It shows that SRSC features may be effective in exposing the tampering.

15.3.3 Experiments of Binary Classification

To test our proposed shift-recompression-based SRSC features, we select 5,150 singly compressed JPEG images at quality factor 85, and 5,150 singly compressed JPEG images at quality factor 40. Respectively, we cropped these JPEG images by all misalignment combinations (p, q), from $(0, 1), \ldots, (0, 7)$; $(1, 0), \ldots, (1, 7)$; ... to $(7, 7)$, a total of 63 combinations, and produced $5{,}150 \times 63 = 324{,}450$ cropped JPEG images at quality 85, based on the singly compressed images of the quality 85; and produced 324,450 cropped JPEG image at quality 40, based on the singly compressed JPEG images of the quality 40. In a fair manner, since these cropped

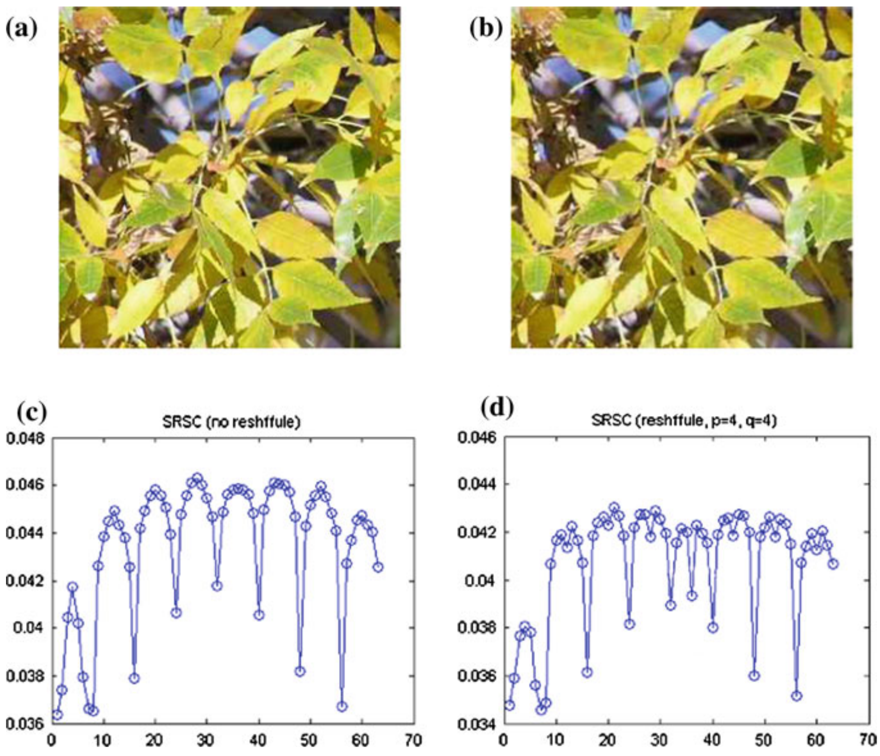


Fig. 15.5 A comparison of SRSC features from an original JPEG image and a cropped JPEG image. X-label shows the SRSC feature index and y-label indicates the value [24]. **a** An original image. **b** A cropped image ($p = 4$, $q = 4$). **c** SRSC (original image). **d** SRSC (cropped, $p = 4$, $q = 4$)

JPEG images were processed by twice JPEG compressions with the same quantization matrix, the singly compressed JPEG images are also uncompressed and then recompressed by using the same quantization matrix. Then we extract SRSC features from all these JPEG images. Support vector machines [43] are employed in our algorithm to discriminate each type of misaligned cropping from no cropping, which is a binary classification from the perspective of pattern recognition. In our experiments, we apply two popular SVM techniques, SVMlight [18], and LibSVM [15], with linear kernel, polynomial kernel, and RBF kernel individually to the features, for training and validation. The ratio of training to testing is 50–50 %, 50 experiments are operated in each type of detection. In each experiment, training samples are randomly selected and remaining samples are selected for validation. Validation (testing) results can be divided into true positive (TP), false negative (FN), false positive (FP), and true negative (TN), based on the ground truth and prediction results. In our experiments, the classification results by using LibSVM are generally better than SVMlight, hence we only list the results using LibSVM in Table 15.1. We calculate

Table 15.1 Binary classification accuracy on testing sets by using LibSVM with linear, polynomial, and RBF kernels; the best average testing accuracy by using these kernels is shown ($Q = 40$) [24]

p	q feature set	0	1	2	3	4	5	6	7
0	BACM		64.5 %	55.7	54.8	54.2	55.3	56.9	62.2
	SRSC		**96.6**	**99.0**	**98.9**	**99.3**	**98.8**	**98.8**	**96.9**
1	BACM	61.9	57.5	55.7	55.9	56.5	53.9	54.9	57.2
	SRSC	**95.7**	**96.7**	**99.1**	**99.4**	**99.4**	**99.3**	**99.0**	**97.5**
2	BACM	56.9	57.9	57.5	56.3	57.7	55.3	56.2	57.2
	SRSC	**98.6**	**99.0**	**99.5**	**99.5**	**99.5**	**99.6**	**99.5**	**99.0**
3	BACM	55.7	58.0	58.2	59.5	56.9	53.9	56.8	56.9
	SRSC	**98.5**	**99.4**	**99.4**	**98.8**	**98.9**	**98.7**	**99.5**	**99.3**
4	BACM	54.3	57.6	58.0	58.1	57.1	54.1	55.1	56.2
	SRSC	**98.9**	**99.5**	**99.5**	**98.8**	**98.9**	**98.6**	**99.4**	**99.4**
5	BACM	59.9	56.5	58.6	58.6	55.8	53.9	56.8	57.0
	SRSC	**98.7**	**99.4**	**99.5**	**98.9**	**98.9**	**98.7**	**99.3**	**99.2**
6	BACM	60.0	56.8	57.0	55.2	55.6	54.1	55.6	56.2
	SRSC	**98.7**	**99.0**	**99.4**	**99.5**	**99.5**	**99.4**	**99.3**	**98.9**
7	BACM	60.9	57.9	56.3	56.8	55.9	56.6	54.7	56.3
	SRSC	**95.9**	**97.3**	**99.1**	**99.4**	**99.4**	**99.1**	**99.1**	**97.5**

testing accuracy by one-half of the sum of true positive rate and true negative rate, or 0.5 * (TP/(TP + FN) + TN/(TN + FP)).

A recent periodicity analysis of compression artifacts for tampering detection takes advantage of BACM, and the detection results on misaligned cropping and recompression with the same quantization matrix are not effective, as shown in [3]. In our experiments, we only compare our approach to BACM feature set [33].

As shown in Tables 15.1 and 15.2 SRSC-based approach leads to very impressive results where most average testing accuracy values are over 98 %, while the detection accuracy values based on BACM are around 60 %.

15.3.4 Experiments of Multilabel Classification

In Fig. 15.4a, if S_2 is cropped, for example, the pixels on the boundary are stripped off, then the region R_1 from S_1 is composited to S_2, and the doctored image is compressed with the same quantization matrix. In this case, how do we identify the forged area in the compositing? The binary classification is not good enough. If we can identify the misalignment of S_2 from the misalignment of R_1, then we can reveal the different cropping manipulations and locate the forged area in the compositing.

In these type of experiments, we select 2,000 singly compressed JPEG images at quality factor 85, and 2,000 singly compressed JPEG images at quality factor 40.

Table 15.2 Binary classification accuracy on testing sets by using LibSVM with linear, polynomial, and RBF kernels; the best average testing accuracy by using these kernels is shown ($Q = 85$) [24]

p	q feature set	0	1	2	3	4	5	6	7
0	BACM		63.6 %	62.6	62.2	61.4	65.2	65.2	62.5
	SRSC		**99.1**	**99.3**	**99.6**	**99.7**	**99.4**	**99.1**	**98.8**
1	BACM	62.5	58.0	59.2	59.8	61.2	60.9	59.6	58.8
	SRSC	**98.6**	**98.7**	**99.0**	**99.2**	**99.3**	**99.1**	**98.8**	**98.6**
2	BACM	60.4	58.5	58.1	59.1	59.6	60.5	59.3	58.5
	SRSC	**99.0**	**99.1**	**98.9**	**99.1**	**99.2**	**98.9**	**98.6**	**98.9**
3	BACM	62.3	59.3	58.9	60.5	60.6	63.7	60.2	59.9
	SRSC	**99.2**	**99.3**	**99.1**	**98.9**	**98.9**	**98.6**	**98.7**	**99.2**
4	BACM	60.6	61.0	61.0	63.2	64.7	65.4	62.3	61.2
	SRSC	**99.2**	**99.3**	**99.1**	**98.8**	**98.9**	**98.6**	**98.8**	**99.2**
5	BACM	65.9	62.5	62.8	63.3	66.4	66.0	63.7	63.5
	SRSC	**99.1**	**99.1**	**98.9**	**98.7**	**98.7**	**98.5**	**98.7**	**98.9**
6	BACM	63.1	58.8	58.7	60.5	63.4	63.7	60.5	59.4
	SRSC	**98.9**	**98.9**	**98.7**	**98.9**	**99.0**	**98.7**	**98.3**	**98.6**
7	BACM	62.9	59.4	59.1	59.5	61.6	62.3	59.7	58.8
	SRSC	**98.5**	**98.8**	**99.1**	**99.1**	**99.2**	**99.0**	**98.6**	**98.6**

Respectively, we cropped these JPEG images with the all displacement combinations (p, q), from $(0, 1), \ldots, (0, 7); (1, 0), \ldots, (1, 7); \ldots$ to $(7, 7)$, a total of 63 combinations, and produced $2{,}000 \times 63 = 126{,}000$ cropped JPEG images at the quality 85, and 126,000 cropped JPEG image at quality 40. We apply a logistic regression classifier for the multilabel classification. Hundred experiments are operated for each detection. In each experiment, training feature set is randomly selected and remaining feature set is used for testing. The ratio of training samples to testing samples is 50–50 %.

Logistic regression or logit regression is a probabilistic statistical classification model. In order to distinguish m classes on the basis of an input vector of length d, if x is an input vector $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$ and y represents the class label vector y $= [y^{(1)}, y^{(2)}, \ldots, y^{(m)}]^T$. The training samples are represented as a set of training data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$. Under a multinomial logistic regression model, the probability that x belongs to class i is written as

$$\mathrm{P}\left(y^{(i)} = 1\,|x, w\right) = \frac{\exp\left({w^{(i)}}^T x\right)}{\sum_{j=1}^{m} \exp\left({w^{(j)}}^T x\right)}, \tag{15.5}$$

where $w^{(i)}$ is the weight vector corresponding to class i. In supervised learning, the components of w are estimated from the training data. To perform maximum likelihood estimation of w, we may simply maximizes the log-likelihood function, which is typically achieved using Newton's method as iteratively reweighted least squares.

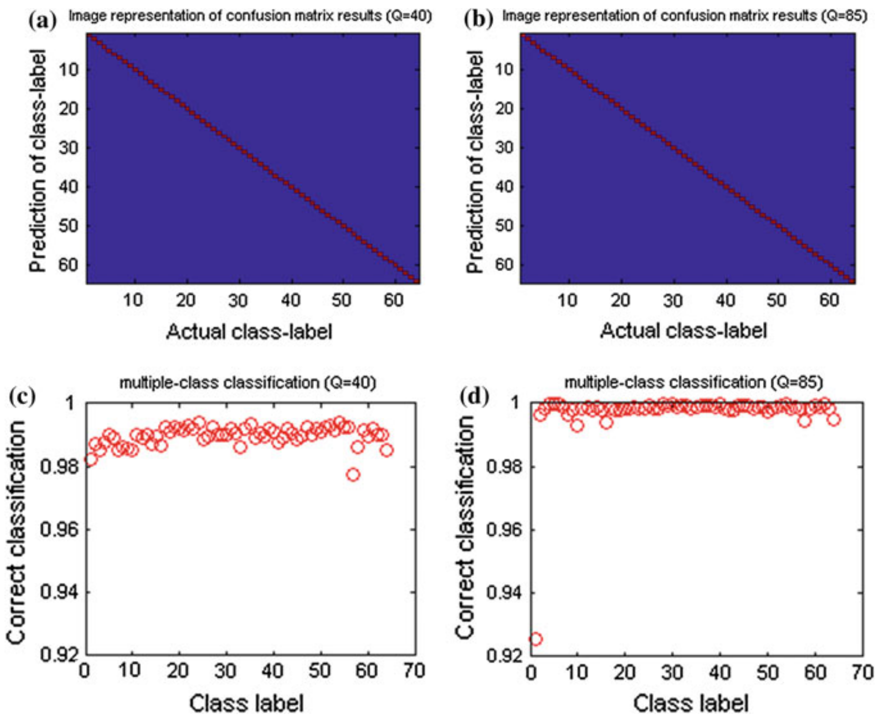


Fig. 15.6 Confusion matrix of average accuracy values on the testing (*first row*) and correct classification for each type of displaced JPEG images (*second row*) in multilabel classification by using logistic regression classifier [24]

$$\ell(w) = \sum_{j=1}^{n} \log P\left(y_j|x_j, w\right) = \sum_{j=1}^{n} \left[\sum_{i=1}^{m} y_j^{(i)} w^{(i)^T} x_j - \log \sum_{i=1}^{m} \exp\left(w^{(i)^T} x_j\right) \right]. \tag{15.6}$$

Based on multinomial logistic regression, Krishnapuram et al. [23] introduced a multiclass formulation and derived fast exact algorithms for learning sparse multiclass classifiers by combining a bound optimization approach with a component-wise update procedure, which scale favorably in both the number of training samples and the feature dimensionality.

We assign the label 1 to the untouched images, and the labels 2–64 to touched images under different combination of the offsets in the shift recompression. We obtained the confusion matrix of average accuracy over 100 experiments in the multilabel classification (total 64 labels, containing $64 \times 64 = 4{,}096$ average accuracy values at each quality factor), the accuracy values are shown in image format by Fig. 15.6a, b. The average accuracy values along the diagonal direction or correct recognition for each cropping type of JPEG images are given by Fig. 15.6c, d. The x-label indicates

Table 15.3 Average detection performance over 50 times using a linear LibSVM and logistic regression [24]

Classifier	True negative rate (%)	True positive rate (%)
LogitReg	99.6	99.5
LibSVM	99.5	99.4

The detection results in Table 15.2 were obtained without any error-reduction

the class label (class 1 represents original image, class 2 to class 64 denote the misalignment distance coordinates from (0, 1) to (7, 7), respectively), and y-label shows the correct classification for each class in all combinations.

15.3.5 Detecting Relevant Copy-Paste and Composite Forgery

To create copy-paste forgery and composite forgery database, we select 2,000 singly compressed JPEG images at the quality factor 85 and create 2,000 copy-paste JPEG forgery at their central 64×64 regions, and 2,000 composite JPEG forgery at central 64×64 regions, with a random selection of displacement, and the manipulated images are encoded in JPEG format at the same compression quality (or by using the same quantization table) to the premanipulated image. To detect such copy-paste and composite manipulations, we extract the SRSC features from different regions of an image by using the following procedure (Table 15.3).

(1) Extract $\mathbf{SRSC}_R$ features from each region of interest R. Let $\mathbf{R}(r_1, r_2)$ stand for the (r_1, r_2) subregion of S, and four horizontal and vertical neighbor regions are denoted by $\mathbf{R}(r_1 - 1, r_2)$, $\mathbf{R}(r_1 + 1, r_2)$, $\mathbf{R}(r_1, r_2 - 1)$, and $\mathbf{R}(r_1, r_2 + 1)$. We slide a window over the image under examination from the upper-left to the right-bottom, in the horizontal direction first and then in the vertical direction. Each movement of the window shifts 8 pixels. In our experiment, the window size is set 64×64, $\mathbf{R}(r_1 - 1, r_2)$, $\mathbf{R}(r_1 + 1, r_2)$, $\mathbf{R}(r_1, r_2 - 1)$, and $\mathbf{R}(r_1, r_2 + 1)$ have 87.5 % overlap with $\mathbf{R}(r_1, r_2)$, as shown by Fig. 15.7.

(2) The multilabel classification models are loaded to classify the features from each region, and all prediction results are organized as a two-dimensional array, in terms of the region indices.

(3) Based on the class-label occurrence, we can automatically detect a forged image, and approximately locate the tampering area in the image. Though the detection accuracy in our multilabel classification is very impressive and mostly higher than 98 %, it is not 100 %. Therefore, for practical applications, the sparsely distributed class labels with the label value larger than 1 but not constant with the neighbor class labels imply that the predictions are probably the classification errors (due to the high portion overlapping of the neighboring regions, as shown by Fig. 15.7). It is reasonable to rule out these regions from the list of the forgery. This processing may be called error-reduction or noise-removal.

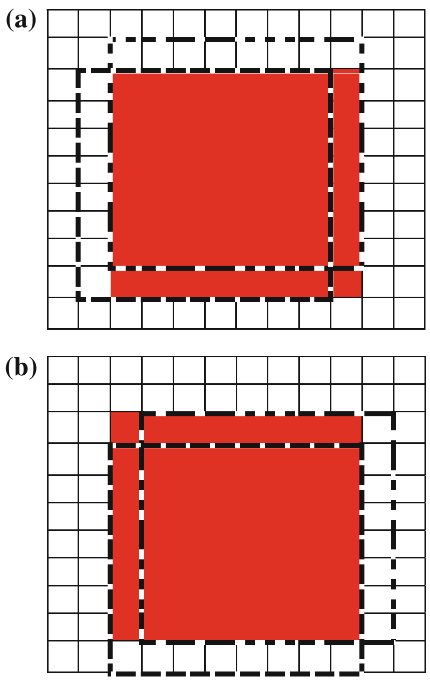


Fig. 15.7 *Shaded* region $\mathbf{R}(r_1, r_2)$ and the *dashed* neighbors $\mathbf{R}\ (r_1 - 1, r_2)$, $\mathbf{R}(r_1, r_2 - 1)$ in **a**, and $\mathbf{R}(r_1 + 1, r_2)$, $\mathbf{R}(r_1, r_2 + 1)$ in **b**

Table 15.2 shows the detection results without error-reduction process when distinguishing copy-paste and composite forgery from untouched JPEG images by using a linear LibSVM and logistic regression classifiers, respectively. To obtain the results in Table 15.2, we first predict the class-label of each subregion of the image under examination. In each experiment, we randomly select 60 % feature sets for training and other 40 % feature sets are tested. Fifty experiments are performed for each testing. It is noted that the classification accuracy values shown in Table 15.2 are the results without applying any error-reduction.

After applying error-reduction, the detection results may improve. Figure 15.8a shows an untouched image and Fig. 15.8b shows a doctored one with the copy-paste at the center, with the same quantization matrix. Figure 15.8c, d are the image representation of detection results by the above steps (1) and (2). The x-axis and y-axis show the subregion indices. Figure 15.8e, f are the final results after the error-reduction or noise-removal that is described in the above step (3).

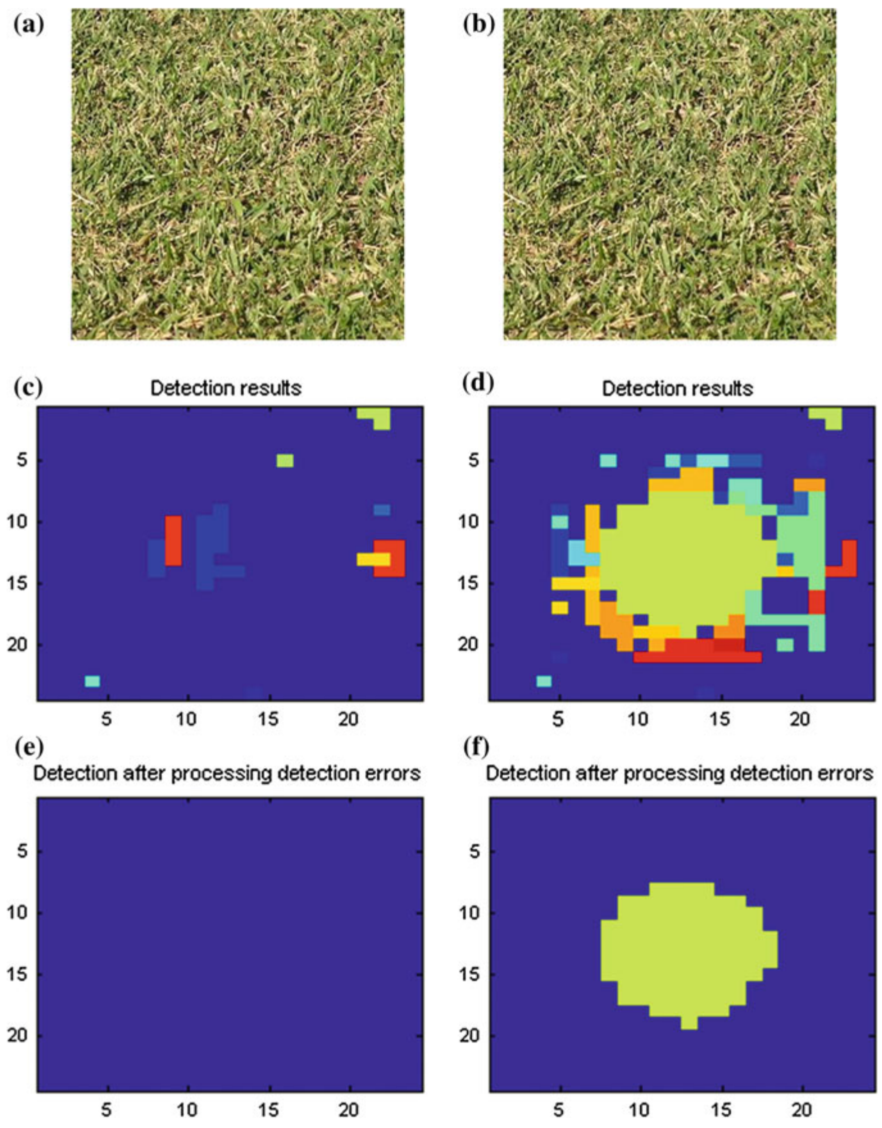


Fig. 15.8 An illustration of forgery detection using SRSC features and logistic regression classifier [24]. **a** An original image. **b** A copy-paste at the central region. **c** Classification results of (**a**). **d** Classification results of (**b**). Error-reduction to **e** (**c**) . Error-reduction to **f** (**d**)

15.4 Conclusions

We have proposed a shift-recompression-based spatial approach to detection of misaligned cropping and recompression with the same quantization matrix and relevant forgery in JPEG images. Experimental results show that the proposed approaches are very effective in exposing the forgery that is encoded by using the same quantization matrix in JPEG images.

We should note that the proposed method does not perform well in detecting seam-carving-based image forgery [1, 11, 41] with the same quantization matrix. The detection of such a forgery has been well investigated separately and the results can be found in [27].

Acknowledgments The study was supported from the National Institute of Justice, Office of Justice Programs, U.S. Department of Justice, under the Award No. 2010-DN-BX-K223 and from the Sam Houston State University under two enhanced research grants (ERG, 2011–2013).

References

1. Avidan S, Shamir A (2007) Seam carving for content-aware image resizing. ACM Trans Graph 26(3), Article 10
2. Bayram S, Sencar HT, Memon N (2009) An efficient and robust method for detecting copy-move forgery. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, April 2009
3. Chen Y, Hsu C (2011) Detecting recompression of JPEG images via periodicity analysis of compression artifacts for tampering detection. IEEE Trans Inf Forensics Secur 6(2):396–406
4. Chen C, Shi YQ (2008) JPEG image steganalysis utilizing both intrablock and interblock correlations. In: Proceedings of the 2008 IEEE international symposium on circuits and systems, pp 3029–3032
5. Chen W, Shi YQ, Su W (2007) Image splicing detection using 2-D phase congruency and statistical moments of characteristic function. Proc SPIE 6505: 65050R. doi:10.1117/12.704321
6. Dirik AE, Memon N (2009) Image tamper detection based on demosaicing artifacts. In: Proceedings of the IEEE ICIP'09, November 2009
7. Farid H (1999) Detecting digital forgeries using bispectral analysis. AI Lab, Massachusetts Institute of Technology, Technical report AIM-1657
8. Farid H (2006) Digital image ballistics from JPEG quantization. Department Computer Science, Dartmouth College, Technical report TR2006-583
9. Farid H (2009) Image forgery detection, a survey. In: IEEE Signal Processing Magazine, pp 16–25, March 2009
10. Farid H (2009) Exposing digital forgeries from JPEG ghosts. IEEE Trans Inf Forensics Secur 4(1):154–160
11. Fillion C, Sharma G (2010) Detecting content adaptive scaling of images for forensics applications. Proc SPIE, Media forensics Secur II 7541:2010. doi:10.1117/12.838647
12. Fridrich J, Kodovsky J (2012) Rich models for steganalysis of digital images. IEEE Trans Inf Forensics Secur 7(3):868–882
13. Fridrich J, Soukal D, Lukás J (2003) Detection of copy move forgery in digital images. In: Proceedings of the digital forensic research workshop, August 2003
14. Hilbe JM (2009) Logistic regression models. Chapman and Hall/CRC Press, London. ISBN: 978-1-4200-7575-5

15. http://www.csie.ntu.edu.tw/~cjlin/libsvm/
16. http://www.npr.org/blogs/thetwo-way/2010/09/17/129938169/doctored-photograph-hosni-mubarak-al-ahram-white-house-obama-mideast-peace-talks
17. Huang F, Huang J, Shi Y (2010) Detecting double JPEG compression with the same quantization matrix. IEEE Trans Inf Forensics Secur 5(4):848–856
18. Joachims T (2000) Estimating the generalization performance of a SVM efficiently. In: Proceedings of the international conference on machine learning, Morgan Kaufmann
19. Kodovsky J, Fridrich J (2009) Calibration revisited. In: Proceedings of the 11th ACM multimedia and security workshop, pp 63–74
20. Kodovsky J, Fridrich J (2011) Steganalysis in high dimensions: fusing classifiers built on random subspaces. Proc SPIE 7880: 78800L. doi:10.1117/12.872279
21. Kodovsky J, Fridrich J (2011). Steganalysis in high dimensions: fusing classifiers built on random subspaces. Proc SPIE 7880: 78800L. doi:10.1117/12.872279
22. Kodovsky J, Fridrich J (2012) Steganalysis of JPEG images using rich models. Proc. SPIE 8303, Media Watermarking Secur forensics 2012, 83030A, February 2012. doi:10.1117/12.907495
23. Krishnapuram B, Carin L, Figueiredo MAT, Hartemink AJ (2005) Sparse multinomial logistic regression: fast algorithms and generalization bounds. IEEE Trans Pattern Anal Mach Intell 27(6):957–968
24. Liu Q (2011) Detection of misaligned cropping and recompression with the same quantization matrix and relevant forgery. In: Proceedings of the 3rd ACM multimedia workshop on multimedia in forensics, pp 25–30
25. Liu Q (2011) Steganalysis of DCT-embedding-based adaptive steganography and YASS. In: Proceedings of the 13th ACM workshop on multimedia and security, September 2011, Buffalo
26. Liu Q, Sung AH, Qiao M (2009) Temporal derivative based spectrum and mel-cepstrum audio steganalysis. IEEE Trans Inf Forensics Secur 4(3):359–368
27. Liu Q, Chen Z (2014) Improved approaches with calibrated neighboring joint density to steganalysis and seam carved forgery detection in JPEG images, ACM Trans Intell Syst Technol 5(4), article 63
28. Liu Q, Sung AH, Qiao M (2009) Improved detection and evaluation for JPEG steganalysis. In: Proceedings of the ACM multimedia, pp 873–876
29. Liu Q, Sung AH, Qiao M (2009) Novel stream mining for audio steganalysis. In: Proceedings of the 17th ACM multimedia, pp 95–104
30. Liu Q, Sung AH, Qiao M (2011) A method to detect JPEG-based double compression. In: Proceedings of the 8th international symposium on neural networks, pp 466–476
31. Liu Q, Sung AH, Qiao M (2011) Derivative based audio steganalysis. ACM Trans. Multimed. Comput. Commun. Appl. 7(3), 1–19, article 18
32. Liu Q, Sung AH, Qiao M (2011) Neighboring joint density based JPEG steganalysis. ACM Trans Intell Syst Technol 2(2), 1–16, Article 16
33. Luo W, Qu Z, Huang J, Qiu G (2007) A novel method for detecting cropped and recompressed image block. In: Proceedings of the IEEE conference acoustics, speech and signal processing, pp 217–220
34. Pan X, Lyu S (2010) Region duplication detection using image feature matching. IEEE Trans Inf Forensics Secur 5(4):857–867
35. Pevny T, Fridrich J (2007) Merging Markov and DCT features for multi-class JPEG steganalysis. Proc SPIE, 6505: 650503. doi:10.1117/12.696774
36. Pevny T, Fridrich J (2008) Detection of double-compression in JPEG images for applications in steganography. IEEE Trans Inf Forensics Secur 3(2):247–258
37. Popescu AC, Farid H (2004) Statistical tools for digital forensics. In: Proceedings of the 6th international workshop on information hiding, pp 128–147
38. Popescu AC, Farid H (2005) Exposing digital forgeries by detecting traces of re-sampling. IEEE Trans Signal Process 53(2):758–767
39. Popescu AC, Farid H (2005) Exposing digital forgeries in color filter array interpolated images. IEEE Trans. Signal Process. 53(10):3948–3959

40. Prasad S, Ramakrishnan KR (2006) On resampling detection and its application to image tampering. In: Proceedings of the IEEE international conference multimedia and exposition, pp 1325–1328
41. Sarkar A, Nataraj L, Manjunath BS (2009) Detection of seam carving and localization of seam insertions in digital images. In: Proceedings of the 11th ACM workshop on multimedia and security, pp 107–116
42. Shi YQ, Chen C, Chen W (2007) A Markov process based approach to effective attacking JPEG steganography. Lecture Notes in Computer Science, vol 4437. Springer, Berlin. pp 249–264
43. Vapnik V (1998) Statistical learning theory. Wiley, New York

Part V
Algorithms for Multimedia Data Presentation, Processing and Visualization

Chapter 16
Fast Binary Embedding for High-Dimensional Data

Felix X. Yu, Yunchao Gong and Sanjiv Kumar

Abstract Binary embedding of high-dimensional data requires long codes to preserve the discriminative power of the input space. Traditional binary coding methods often suffer from very high computation and storage costs in such a scenario. To address this problem, we propose two solutions which improve over existing approaches. The first method, Bilinear Binary Embedding (BBE), converts high-dimensional data to compact similarity-preserving binary codes using compact bilinear projections. Compared to methods that use unstructured matrices for projection, it improves the time complexity from $\mathcal{O}(d^2)$ to $\mathcal{O}(d^{1.5})$, and the space complexity from $\mathcal{O}(d^2)$ to $\mathcal{O}(d)$ where d is the input dimensionality. The second method, Circulant Binary Embedding (CBE), generates binary codes by projecting the data with a circulant matrix. The circulant structure enables the use of Fast Fourier Transformation to speed up the computation. This further improves the time complexity to $\mathcal{O}(d \log d)$. For both BBE and CBE, we propose to learn the projections in a data-dependent fashion. We show by extensive experiments that the proposed approaches give much better performance than the state of the arts for fixed time, and provides much faster computation with no performance degradation for fixed number of bits. The BBE and CBE methods were previously presented in [6, 38]. In this book chapter, we present the two approaches in a unified framework, covering randomized binary embedding, learning-based binary embedding, and learning with dimension reductions. We also discuss the choice of algorithms.

F.X. Yu (✉)
Department of Electrical Engineering, Columbia University,
116th Street & Broadway, New York, NY 10025, USA
e-mail: yuxinnan@ee.columbia.edu

Y. Gong
Facebook AI Research, 1601 Willow Rd, Menlo Park, CA 94025, USA
e-mail: ycgong@facebook.com

S. Kumar
Google Research, 111 8th Ave, New York, NY 10011, USA
e-mail: sanjivk@google.com

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_16

16.1 Introduction

Embedding input data in binary spaces is becoming popular for efficient retrieval and learning on massive datasets [5, 6, 10, 11, 13, 19–22, 24, 30]. Moreover, in a large number of application domains such as computer vision, biology, and finance, data is typically high-dimensional. Taking image retrieval and classification as an example, it has been shown recently that in order to achieve high accuracy on large-scale datasets, it is advantageous to use very high-dimensional descriptors such as Fisher Vectors (FV) [26, 27, 31], Vector of Locally Aggregated Descriptors (VLAD) [15], Locally Constraint Linear Code (LLC) [35], or a large set of weak attributes [37]. When representing such high-dimensional data by binary codes, it has been shown that long codes are required in order to achieve good performance. In fact, the required number of bits is $\mathcal{O}(d)$, where d is the input dimensionality [6, 19, 31].

The goal of binary embedding is to well approximate the input distance as Hamming distance so that efficient learning and retrieval can happen directly in the binary space. It is important to note that another related area called *hashing* is a special case with slightly different goal: creating hash tables such that points that are similar fall in the same (or nearby) bucket with high probability. In fact, even in hashing, if high accuracy is desired, one typically needs to use hundreds of hash tables involving tens of thousands of bits.

Most of the existing linear binary coding approaches generate the binary code by applying a "full" (unstructured) projection matrix, followed by a binarization step. Formally, given a data point, $\mathbf{x} \in \mathbb{R}^d$, the k-bit binary code, $h(\mathbf{x}) \in \{+1, -1\}^k$ is generated simply as

$$h(\mathbf{x}) = \text{sgn}(\mathbf{R}\mathbf{x}), \tag{16.1}$$

where $\mathbf{R} \in \mathbb{R}^{k\times d}$, and $\text{sgn}(\cdot)$ is a binary map which returns element-wise sign.[1] Several techniques have been proposed to generate the projection matrix randomly without taking into account the input data [2, 30]. These methods are very popular due to their simplicity but often fail to give the best performance due to their inability to adapt the codes with respect to the input data. Thus, a number of data-dependent techniques have been proposed with different optimization criteria such as reconstruction error [17], data dissimilarity [23, 36], ranking loss [24], quantization error after PCA [7], and pairwise misclassification [34]. These methods are shown to be effective for learning compact codes for relatively low-dimensional data. However, the $\mathcal{O}(d^2)$ computational and space costs prohibit them from being applied to learning long codes for high-dimensional data. For instance, to generate $\mathcal{O}(d)$-bit binary codes for data with $d \sim 1$M, a huge projection matrix will be required needing TBs of memory, which is not practical.[2]

[1] A few methods transform the linear projection via a nonlinear map before taking the sign [30, 36].

[2] In principle, one can generate the random entries of the matrix on-the-fly (with fixed seeds) without needing to store the matrix. But this will increase the computational time even further.

Table 16.1 Comparison of the proposed methods (BBE and CBE) with the full projection-based methods for generating long codes (code dimension k comparable to input dimension d)

Method	Time	Space	Time (Learning)
Full projection	$\mathcal{O}(d^2)$	$\mathcal{O}(d^2)$	$\mathcal{O}(nd^3)$
BBE	$\mathcal{O}(d^{1.5})$	$\mathcal{O}(d)$	$\mathcal{O}(nd^{1.5})$
CBE	$\mathcal{O}(d \log d)$	$\mathcal{O}(d)$	$\mathcal{O}(nd \log d)$

n is the number of instances used for learning data-dependent projection matrices. The $\mathcal{O}(d^{1.5})$ computational complexity of BBE can be achieved when the input vector is reshaped as a $\sqrt{d} \times \sqrt{d}$ matrix. The $\mathcal{O}(d \log d)$ computational complexity of CBE is achieved by using FFT to speed up the computation.

In order to overcome the computational challenges for the full projection-based methods, we propose two approaches reducing both the computational cost and storage cost. The first method, Bilinear Binary Embedding (BBE), reshapes the input vector $\mathbf{x}$ into a matrix $\mathbf{Z}$, and applies a bilinear projection to get the binary code:

$$h(\mathbf{x}) = \text{vec}(\text{sgn}(\mathbf{R}_1^T \mathbf{Z} \mathbf{R}_2)). \tag{16.2}$$

We use vec$(\cdot)$ to denote an operator which reshapes a matrix to a vector. It is easy to show that when the shapes of $\mathbf{Z}, \mathbf{R}_1, and\, \mathbf{R}_2$ are $\mathcal{O}(\sqrt{d}) \times \mathcal{O}(\sqrt{d})$, the method has time and space complexity of $\mathcal{O}(d^{1.5})$ and $\mathcal{O}(d)$, respectively. The BBE method was originally presented in [6].

The second method, Circulant Binary Embedding (CBE), is even faster than BBE. This is achieved by imposing a *circulant structure* on the projection matrix $\mathbf{R}$ in (16.1).

$$h(\mathbf{x}) = \text{sgn}(\mathbf{R}\mathbf{x}), \quad \mathbf{R} \text{ is a circulant matrix.} \tag{16.3}$$

This special structure allows us to use Fast Fourier Transformation (FFT) based techniques, which have been extensively used in signal processing. The proposed method further reduces the time complexity to $\mathcal{O}(d \log d)$, enabling efficient binary embedding for very high-dimensional data.[3] The CBE method was originally presented in [38].

Table 16.1 compares the time and space complexity for different methods. This book chapter along with [6, 38] make the following contributions:

- We propose the bilinear binary embedding (BBE) and circulant binary embedding (CBE) methods, which reduce both the computational cost and storage cost of binary embedding for high-dimensional data.

[3] One could in principal use other structured matrices like Hadamard matrix along with a sparse random Gaussian matrix to achieve fast projection as was done in fast Johnson–Lindenstrauss transform [1, 3], but it is still slower than circulant projection and needs more space.

- For both methods, in addition to the randomized versions, we propose to learn the data-dependent projections. This helps to further improve the coding quality by considering the data distributions.
- Extensive experiments show that, compared to the state of the art, the proposed methods improve the result dramatically for a fixed time cost, and provide much faster computation with no performance degradation for a fixed number of bits.

16.2 Bilinear Binary Embedding (BBE)

Most high-dimensional descriptors have a natural matrix or tensor structure. For example, a HOG descriptor is a two-dimensional grid of histograms, and this structure has been exploited for object detection [29]. A Fisher Vector [26, 27, 31] can be represented as a $k \times 2l$ matrix, where k is the visual vocabulary size and l is the dimensionality of the local image features (the most common choice is SIFT with $l = 128$). VLAD [15], which can be seen as a simplified version of FV, can be represented as a $k \times l$ matrix. Finally, an LLC [35] descriptor with s spatial bins can be represented as a $k \times s$ matrix.

Let $\boldsymbol{x} \in \mathbb{R}^d$ denote our descriptor vector. Based on the structure and interpretation of the descriptor, we reorganize it into a $d_1 \times d_2$ matrix with $d = d_1 d_2$:

$$\boldsymbol{x} \in \mathbb{R}^{d_1 d_2 \times 1} \quad \mapsto \quad \mathbf{Z} \in \mathbb{R}^{d_1 \times d_2}. \tag{16.4}$$

We assume that each vector $\boldsymbol{x} \in \mathbb{R}^d$ is zero-centered and has unit norm, as L_2 normalization is a widely used preprocessing step that usually improves performance [28].

We will first introduce a randomized method to obtain d-bit bilinear codes in Sect. 16.2.1 and then explain how to learn data-dependent codes in Sect. 16.2.2. Learning of reduced-dimension codes will be discussed in Sect. 16.2.3.

16.2.1 Randomized Bilinear Binary Embedding (Bilinear-rand)

To convert a descriptor $\boldsymbol{x} \in \mathbb{R}^d$ to a d-dimensional binary string, we first consider the framework of [2, 7] that applies a random rotation $\mathbf{R} \in \mathbb{R}^{d \times d}$ to $\boldsymbol{x}$:

$$h(\boldsymbol{x}) = \mathrm{sgn}(\mathbf{R}\boldsymbol{x}). \tag{16.5}$$

Since $\boldsymbol{x}$ can be represented as a matrix $\mathbf{Z} \in \mathbb{R}^{d_1 \times d_2}$, to make rotation more efficient, we propose a bilinear formulation using two random orthogonal matrices $\mathbf{R}_1 \in \mathbb{R}^{d_1 \times d_1}$ and $\mathbf{R}_2 \in \mathbb{R}^{d_2 \times d_2}$:

$$h(\mathbf{x}) = \mathrm{vec}\left(\mathrm{sgn}(\mathbf{R}_1^T \mathbf{Z} \mathbf{R}_2)\right), \tag{16.6}$$

where vec(·) denotes column-wise concatenation.

It is easy to show that applying a bilinear rotation to $\mathbf{Z} \in \mathbb{R}^{d_1 \times d_2}$ is equivalent to applying a $d_1 d_2 \times d_1 d_2$ rotation to vec($\mathbf{Z}$). This rotation is given by $\hat{\mathbf{R}} = \mathbf{R}_2 \otimes \mathbf{R}_1$, where $\otimes$ denotes the Kronecker product:

$$\mathrm{vec}(\mathbf{R}_1^T \mathbf{Z} \mathbf{R}_2) = (\mathbf{R}_2^T \otimes \mathbf{R}_1^T)\,\mathrm{vec}(\mathbf{Z}) = \hat{\mathbf{R}}^T\,\mathrm{vec}(\mathbf{Z})$$

follows from the properties of the Kronecker product [18]. Another basic property of the Kronecker product is that if $\mathbf{R}_1$ and $\mathbf{R}_2$ are orthogonal, then $\mathbf{R}_2 \otimes \mathbf{R}_1$ is orthogonal as well [18]. Thus, a bilinear rotation is simply a special case of a full rotation, such that the full rotation matrix $\hat{\mathbf{R}}$ can be reconstructed from two smaller matrices $\mathbf{R}_1$ and $\mathbf{R}_2$.

While the degree of freedom of our bilinear rotation is more restricted than a full rotation, the projection matrices are much smaller, and the projection speed is much faster. In terms of time complexity, performing a full rotation on $\boldsymbol{x}$ takes $\mathcal{O}((d_1 d_2)^2)$ time, while our approach is $\mathcal{O}(d_1^2 d_2 + d_1 d_2^2)$. In terms of space for projections, full rotation takes $\mathcal{O}((d_1 d_2)^2)$, and our approach only takes $\mathcal{O}(d_1^2 + d_2^2)$. For example, for a 64 K-dimensional vector, a full rotation will take roughly 16 GB of RAM, while the bilinear rotations only take 1 MB of RAM. The projection time for a full rotation is more than a second, versus only 3 ms for bilinear rotations.

Note that when d_1 and d_2 are set as $d_1 = d_2 = d^{1/2}$, the BBE method has the lowest computational complexity $\mathcal{O}(d^{1.5})$, and lowest space complexity $\mathcal{O}(d)$. As computational efficiency is the main focus of this paper, we use such settings in the experiment section. Empirically, tuning d_1 and d_2, or setting them accordingly based on the structure of the descriptor may result in better retrieval performance, but it will lead to higher computational cost.

16.2.2 Learning Bilinear Binary Embedding (Bilinear-opt)

In this section, we present a method for learning the rotations $\mathbf{R}_1$ and $\mathbf{R}_2$ that is inspired by two-sided Procrustes analysis [32] and builds on our earlier work [5, 7].

Following [7], we want to find a rotation $\hat{\mathbf{R}}$ such that the angle θ_i between a rotated feature vector $\hat{\mathbf{R}}^T \boldsymbol{x}_i = \mathrm{vec}(\mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2)$ and its binary encoding (geometrically, the nearest vertex of the binary hypercube), $\mathrm{sgn}(\hat{\mathbf{R}}^T \boldsymbol{x}) = \mathrm{vec}(\mathrm{sgn}(\mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2))$, is minimized. Given N training points, we want to maximize

$$\sum_{i=1}^{N} \cos(\theta_i) = \sum_{i=1}^{N} \left(\frac{\mathrm{sgn}(\hat{\mathbf{R}}^T \boldsymbol{x}_i)^T}{\sqrt{d}} (\hat{\mathbf{R}}^T \boldsymbol{x}_i) \right) \tag{16.7}$$

$$= \sum_{i=1}^{N} \left(\frac{\text{vec}(\text{sgn}(\mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2))^T}{\sqrt{d}} \text{vec}\, \mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2) \right)$$

$$= \frac{1}{\sqrt{d}} \sum_{i=1}^{N} \left(\text{vec}(\mathbf{B}_i)^T \text{ vec}(\mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2) \right)$$

$$= \frac{1}{\sqrt{d}} \sum_{i=1}^{N} \text{tr}(\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1), \tag{16.8}$$

where $\mathbf{B}_i = \text{sgn}(\mathbf{R}_1^T \mathbf{Z}_i \mathbf{R}_2)$. Notice that (16.7) involves the large projection matrix $\hat{\mathbf{R}} \in \mathbb{R}^{d \times d}$, direct optimization of which is challenging. However, after reformulation into bilinear form (16.8), the expression only involves the two small matrices $\mathbf{R}_1$ and $\mathbf{R}_2$. Letting $\mathcal{B} = \{\mathbf{B}_1, \ldots, \mathbf{B}_N\}$, our objective function is as follows:

$$\mathcal{Q}(\mathcal{B}, \mathbf{R}_1, \mathbf{R}_2) = \max_{\mathcal{B}, \mathbf{R}_1, \mathbf{R}_2} \sum_{i=1}^{N} \text{tr}(\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1) \tag{16.9}$$
$$\text{s.t.} \quad \mathbf{B}_i \in \{-1, +1\}^{d_1 \times d_2}, \quad \mathbf{R}_1^T \mathbf{R}_1 = I, \quad \mathbf{R}_2^T \mathbf{R}_2 = \mathbf{I}.$$

This optimization problem can be solved by block coordinate ascent by alternating between the different variables $\{\mathbf{B}_1, \ldots, \mathbf{B}_N\}$, $\mathbf{R}_1$, and $\mathbf{R}_2$. We describe the update steps for each variable below, assuming the others are fixed.

(S1) Update $\mathbf{B}_i$. When $\mathbf{R}_1$ and $\mathbf{R}_2$ are fixed, we independently solve for each $\mathbf{B}_i$ by maximizing

$$\mathcal{Q}(\mathbf{B}_i) = \text{tr}(\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1) = \sum_{k=1}^{d_1} \sum_{l=1}^{d_2} \mathbf{B}_i^{kl} \widetilde{\mathbf{V}}_i^{lk},$$

where $\widetilde{\mathbf{V}}_i^{lk}$ denote the elements of $\widetilde{\mathbf{V}}_i = \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1$. $\mathcal{Q}(\mathbf{B}_i)$ is maximized by $\mathbf{B}_i = \text{sgn}(\widetilde{\mathbf{V}}_i^T)$.

(S2) Update $\mathbf{R}_1$. Expanding (16.9) with $\mathbf{R}_2$ and $\mathbf{B}_i$ fixed, we have the following:

$$\mathcal{Q}(\mathbf{R}_1) = \sum_{i=1}^{N} \text{tr}(\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1)$$
$$= \text{tr}\Big(\sum_{i=1}^{N} (\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T) \mathbf{R}_1 \Big) = \text{tr}(\mathbf{D}_1 \mathbf{R}_1),$$

where $\mathbf{D}_1 = \sum_{i=1}^{N} (\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T)$. The above expression is maximized with the help of polar decomposition: $\mathbf{R}_1 = \mathbf{V}_1 \mathbf{U}_1^T$, where $\mathbf{D}_1 = \mathbf{U}_1 \mathbf{S}_1 \mathbf{V}_1^T$ is the SVD of $\mathbf{D}_1$.

(S3) Update $\mathbf{R}_2$:

$$
\begin{aligned}
\mathcal{Q}(\mathbf{R}_2) &= \sum_{i=1}^{N} \operatorname{tr}(\mathbf{B}_i \mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1) \\
&= \sum_{i=1}^{N} \operatorname{tr}(\mathbf{R}_2^T \mathbf{Z}_i^T \mathbf{R}_1 \mathbf{B}_i) \\
&= \operatorname{tr}\left(\mathbf{R}_2^T \sum_{i=1}^{N} (\mathbf{Z}_i^T \mathbf{R}_1 \mathbf{B}_i)\right) = \operatorname{tr}(\mathbf{R}_2^T \mathbf{D}_2)\,,
\end{aligned}
$$

where $\mathbf{D}_2 = \sum_{i=1}^{N} (\mathbf{Z}_i^T \mathbf{R}_1 \mathbf{B}_i)$. Analogously to the update rule for $\mathbf{R}_1$, the update rule for $\mathbf{R}_2$ is $\mathbf{R}_2 = \mathbf{U}_2 \mathbf{V}_2^T$, where $\mathbf{D}_2 = \mathbf{U}_2 \mathbf{S}_2 \mathbf{V}_2^T$ is the SVD of $\mathbf{D}_2$.

We cycle between these updates for several iterations to obtain a local maximum. The convergence of the above program is guaranteed in finite number of iterations as the optimal solution of each step is exactly obtained, each step is guaranteed not to decrease the objective function value, and the objective function is bounded from above. In our implementation, we initialize $\mathbf{R}_1$ and $\mathbf{R}_2$ by random rotations and use three iterations. We have not found significant improvement of performance by using more iterations. The time complexity of this program is $\mathcal{O}(N(d_1^3 + d_2^3))$ where d_1 and d_2 are typically fairly small (e.g., $d_1 = 128,\ d_2 = 500$).

Figure 16.1 visualizes the structure of a VLAD descriptor and the corresponding binary code before and after a learned bilinear rotation.

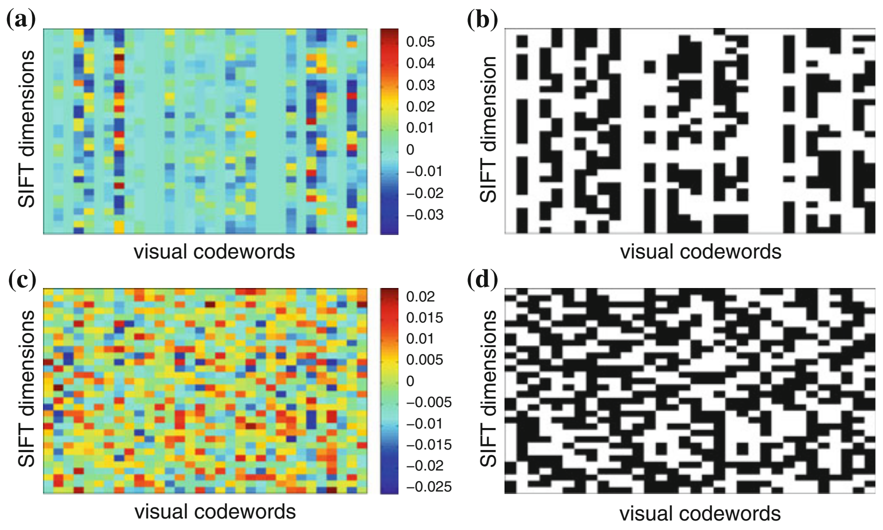


Fig. 16.1 Visualization of the VLAD descriptor and resulting binary code (given by the sign function) before and after learned bilinear rotation. We only show the first 32 SIFT dimensions and visual codewords [6]. Before the rotation, we can clearly see a block pattern, with many zero values. After the rotation, the descriptor and the binary code look more whitened. **a** Original VLAD descriptor. **b** Original binary code. **c** Bilinearly rotated VLAD. **d** Bilinearly rotated binary code

16.2.3 Learning with Dimensionality Reduction

The formulation of Sect. 16.2.2 is used to learn d-dimensional binary codes starting from d-dimensional descriptors. Now, to produce a code of size $c = c_1 \times c_2$, where $c_1 < d_1$ and $c_2 < d_2$, we need projection matrices $\mathbf{R}_1 \in \mathbb{R}^{d_1 \times c_1}$, $\mathbf{R}_2 \in \mathbb{R}^{d_2 \times c_2}$ such that $\mathbf{R}_1^T \mathbf{R}_1 = \mathbf{I}$ and $\mathbf{R}_2^T \mathbf{R}_2 = \mathbf{I}$. Each $\mathbf{B}_i$ is now a $c_1 \times c_2$ binary variable. Consider the cosine of the angle between a lower dimensional projected vector $\hat{\mathbf{R}}^T \boldsymbol{x}_i$ and its binary encoding $\mathrm{sgn}(\hat{\mathbf{R}}^T \boldsymbol{x})$:

$$\cos(\theta_i) = \frac{\mathrm{sgn}(\hat{\mathbf{R}}^T \boldsymbol{x}_i)^T}{\sqrt{c}} \frac{\hat{\mathbf{R}}^T \boldsymbol{x}_i}{\|\hat{\mathbf{R}}^T \boldsymbol{x}_i\|_2},$$

where $\hat{\mathbf{R}} \in \mathbf{R}^{d_1 d_2 \times c_1 c_2}$ and $\hat{\mathbf{R}}^T \hat{\mathbf{R}} = I$. This formulation differs from that of (16.7) in that it contains $\|\hat{\mathbf{R}}^T \boldsymbol{x}_i\|_2$ in the denominator, which makes the optimization difficult [5]. Instead, we follow [5] to define a relaxed objective function based on the sum of linear correlations

$$\mathcal{Q}(\mathcal{B}, \mathbf{R}_1, \mathbf{R}_2) = \sum_{i=1}^{N} \left(\frac{\mathrm{sgn}(\hat{\mathbf{R}}^T \boldsymbol{x}_i)^T}{\sqrt{c}} (\hat{\mathbf{R}}^T \boldsymbol{x}_i) \right).$$

The optimization framework for this objective is similar to that of Sect. 16.2.2. For the three alternating optimization steps, (S1) remains the same. For (S2) and (S3), we compute the SVD of $\mathbf{D}_1$ and $\mathbf{D}_2$ as $\mathbf{U}_1 \mathbf{S}_1 \mathbf{V}_1^T$ and $\mathbf{U}_2 \mathbf{S}_2 \mathbf{V}_2^T$, respectively, and set the two rotations to $\mathbf{R}_1 = \hat{\mathbf{V}}_1 \mathbf{U}_1^T$ and $\mathbf{R}_2 = \hat{\mathbf{U}}_2 \mathbf{V}_2^T$, where $\hat{\mathbf{V}}_1$ is the top c_1 singular vectors of $\mathbf{V}_1$ and $\hat{\mathbf{U}}_2$ is the top c_2 singular vectors of $\mathbf{U}_2$. To initialize the optimization, we generate random orthogonal directions.

16.3 Circulant Binary Embedding (CBE)

In the former sections, we have proposed the BBE method which can produce binary code with computational complexity $\mathcal{O}(d^{1.5})$. In this section, we propose the circulant binary embedding (CBE) method which is even faster than the BBE method.

A circulant matrix $\mathbf{R} \in \mathbb{R}^{d \times d}$ is a matrix defined by a vector $\mathbf{r} = (r_0, r_1, \ldots, r_{d-1})^T$ [9]. Note that the circulant matrix is sometimes equivalently defined by "circulating" the rows instead of the columns.

$$\mathbf{R} = \mathrm{circ}(\mathbf{r}) := \begin{bmatrix} r_0 & r_{d-1} & \ldots & r_2 & r_1 \\ r_1 & r_0 & r_{d-1} & & r_2 \\ \vdots & r_1 & r_0 & \ddots & \vdots \\ r_{d-2} & & \ddots & \ddots & r_{d-1} \\ r_{d-1} & r_{d-2} & \ldots & r_1 & r_0 \end{bmatrix}. \tag{16.10}$$

Let **D** be a diagonal matrix with each diagonal entry being a Bernoulli variable (± 1 with probability 1/2). For $\mathbf{x} \in \mathbb{R}^d$, its d-bit Circulant Binary Embedding (CBE) with $\mathbf{r} \in \mathbb{R}^d$ is defined as:

$$h(\mathbf{x}) = \operatorname{sgn}(\mathbf{RDx}), \tag{16.11}$$

where $\mathbf{R} = \operatorname{circ}(\mathbf{r})$. The k-bit ($k < d$) CBE is defined as the first k elements of $h(\mathbf{x})$. The need for such a **D** is discussed in Sect. 16.3.1. Note that applying **D** to **x** is equivalent to applying random sign flipping to each dimension of **x**. Since sign flipping can be carried out as a preprocessing step for each input **x**, here onwards for simplicity we will drop explicit mention of **D**. Hence the binary code is given as $h(\mathbf{x}) = \operatorname{sgn}(\mathbf{Rx})$.

The main advantage of circulant binary embedding is its ability to use Fast Fourier Transformation (FFT) to speed up the computation.

Proposition 1 *For d-dimensional data, CBE has space complexity $\mathcal{O}(d)$, and time complexity $\mathcal{O}(d \log d)$.*

Since a circulant matrix is defined by a single column/row, clearly the storage needed is $\mathcal{O}(d)$. Given a data point **x**, the d-bit CBE can be efficiently computed as follows. Denote $\circledast$ as operator of circulant convolution. Based on the definition of circulant matrix,

$$\mathbf{Rx} = \mathbf{r} \circledast \mathbf{x}. \tag{16.12}$$

The above can be computed based on Discrete Fourier Transformation (DFT), for which fast algorithm (FFT) is available. The DFT of a vector $\mathbf{t} \in \mathbb{C}^d$ is a d-dimensional vector with each element defined as

$$\mathcal{F}(\mathbf{t})_l = \sum_{m=0}^{d-1} t_n \cdot e^{-i2\pi lm/d}, \quad l = 0, \ldots, d-1. \tag{16.13}$$

The above can be expressed equivalently in a matrix form as

$$\mathcal{F}(\mathbf{t}) = \mathbf{F}_d \mathbf{t}, \tag{16.14}$$

where $\mathbf{F}_d$ is the d-dimensional DFT matrix. Let $\mathbf{F}_d^H$ be the conjugate transpose of $\mathbf{F}_d$. It is easy to show that $\mathbf{F}_d^{-1} = (1/d)\mathbf{F}_d^H$. Similarly, for any $\mathbf{t} \in \mathbb{C}^d$, the Inverse Discrete Fourier Transformation (IDFT) is defined as

$$\mathcal{F}^{-1}(\mathbf{t}) = (1/d)\mathbf{F}_d^H \mathbf{t}. \tag{16.15}$$

Since the convolution of two signals in their original domain is equivalent to the hadamard product in their frequency domain [25],

$$\mathcal{F}(\mathbf{Rx}) = \mathcal{F}(\mathbf{r}) \circ \mathcal{F}(\mathbf{x}). \tag{16.16}$$

Therefore,

$$h(\mathbf{x}) = \operatorname{sgn}\left(\mathcal{F}^{-1}(\mathcal{F}(\mathbf{r}) \circ \mathcal{F}(\mathbf{x}))\right). \tag{16.17}$$

For k-bit CBE, $k < d$, we only need to pick the first k bits of $h(\mathbf{x})$. As DFT and IDFT can be efficiently computed in $\mathcal{O}(d \log d)$ with FFT [25], generating CBE has time complexity $\mathcal{O}(d \log d)$.

16.3.1 Randomized Circulant Binary Embedding (CBE-rand)

A simple way to obtain CBE is by generating the elements of $\mathbf{r}$ in (16.10) independently from the standard normal distribution $\mathcal{N}(0, 1)$. We call this method randomized CBE (CBE-rand). A desirable property of any embedding method is its ability to approximate input distances in the embedded space. Suppose $\mathcal{H}_k(\mathbf{x}_1, \mathbf{x}_2)$ is the normalized Hamming distance between k-bit codes of a pair of points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$:

$$\mathcal{H}_k(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{k} \sum_{i=0}^{k-1} \left|\operatorname{sgn}(\mathbf{R}_{i\cdot}\mathbf{x}_1) - \operatorname{sgn}(\mathbf{R}_{i\cdot}\mathbf{x}_2)\right|/2, \tag{16.18}$$

and $\mathbf{R}_{i\cdot}$ is the ith row of $\mathbf{R}$, $\mathbf{R} = \operatorname{circ}(\mathbf{r})$. If $\mathbf{r}$ is sampled from $\mathcal{N}(0, 1)$, from [2],

$$\mathbf{Pr}\left(\operatorname{sgn}(\mathbf{r}^T\mathbf{x}_1) \neq \operatorname{sgn}(\mathbf{r}^T\mathbf{x}_2)\right) = \theta/\pi, \tag{16.19}$$

where θ is the angle between $\mathbf{x}_1$ and $\mathbf{x}_2$. Since all the vectors that are circulant variants of $\mathbf{r}$ also follow the same distribution, it is easy to see that

$$\mathbf{E}(\mathcal{H}_k(\mathbf{x}_1, \mathbf{x}_2)) = \theta/\pi. \tag{16.20}$$

For the sake of discussion, if k projections, i.e., first k rows of $\mathbf{R}$, were generated independently, it is easy to show that the variance of $\mathcal{H}_k(\mathbf{x}_1, \mathbf{x}_2)$ will be

$$\mathbf{Var}(\mathcal{H}_k(\mathbf{x}_1, \mathbf{x}_2)) = \theta(\pi - \theta)/k\pi^2. \tag{16.21}$$

Thus, with more bits (larger k), the normalized hamming distance will be close to the expected value, with lower variance. In other words, the normalized hamming distance approximately preserves the angle.[4] Unfortunately in CBE, the projections are the rows of $\mathbf{R} = \text{circ}(\mathbf{r})$, which are not independent. This makes it hard to derive the variance analytically. To better understand CBE-rand, we run simulations to compare the analytical variance of normalized hamming distance of independent projections (16.21), and the *sample* variance of normalized hamming distance of circulant projections in Fig. 16.2. For each θ and k, we randomly generate $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ such that their angle is θ.[5] We then generate k-dimensional code with CBE-rand, and compute the hamming distance. The variance is estimated by applying CBE-rand 1,000 times. We repeat the whole process 1,000 times, and compute the averaged variance. Surprisingly, the curves of "Independent" and "Circulant" variances are almost indistinguishable. This means that bits generated by CBE-rand are generally as good as the independent bits for angle preservation. An intuitive explanation is that for the circulant matrix, though all the rows are dependent, circulant shifting one or multiple elements will in fact result in very different projections in most cases. We will later show in experiments on real-world data that CBE-rand and Locality Sensitive Hashing (LSH)[6] have almost identical performance (yet CBE-rand is significantly faster) (Sect. 16.4).

Note that the distortion in input distances after circulant binary embedding comes from two sources: circulant projection, and binarization. For the circulant projection step, recent works have shown that the Johnson–Lindenstrauss-type lemma holds with a slightly worse bound on the number of projections needed to preserve the input distances with high probability [12, 16, 33, 39]. These works also show that before applying the circulant projection, an additional step of randomly flipping the signs of input dimensions is necessary.[7] To show why such a step is required, let us consider the special case when $\mathbf{x}$ is an all-one vector, $\mathbf{1}$. The circulant projection with $\mathbf{R} = \text{circ}(\mathbf{r})$ will result in a vector with all elements to be $\mathbf{r}^T\mathbf{1}$. When $\mathbf{r}$ is independently drawn from $\mathcal{N}(0, 1)$, this will be close to 0, and the norm cannot be preserved. Unfortunately, the Johnson–Lindenstrauss-type results do not generalize to the distortion caused by the binarization step.

[4] In this paper, we consider the case that the data points are ℓ_2 normalized. Therefore the cosine distance, i.e., $1 - \cos(\theta)$, is equivalent to the l_2 distance.

[5] This can be achieved by extending the 2D points $(1, 0)$, $(\cos\theta, \sin\theta)$ to d-dimension, and performing a random orthonormal rotation, which can be formed by the Gram–Schmidt process on random vectors.

[6] Here, by LSH we imply the binary embedding using $\mathbf{R}$ such that all the rows of $\mathbf{R}$ are sampled iid. With slight abuse of notation, we still call it "hashing" following [2].

[7] For each dimension, whether the sign needs to be flipped is predetermined by a $(p = 0.5)$ Bernoulli variable.

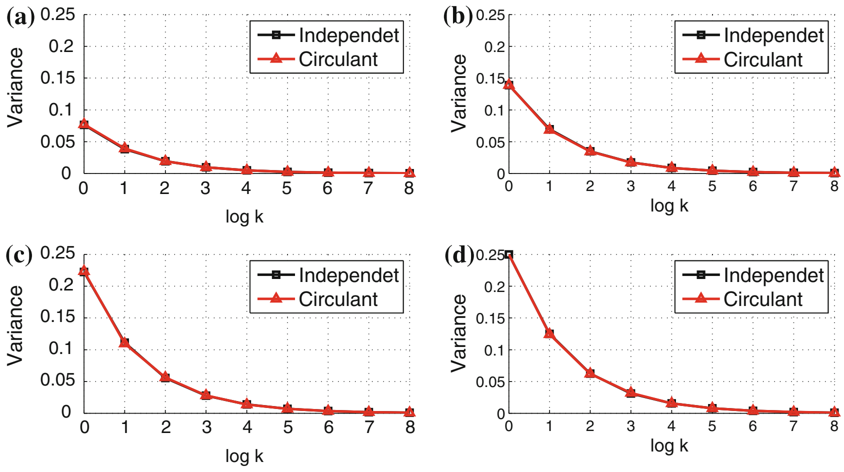


Fig. 16.2 The *analytical* variance of normalized hamming distance of independent bits as in (16.21), and the *sample* variance of normalized hamming distance of circulant bits, as a function of angle between points (θ) and number of bits (k). The two curves overlap [38]. **a** $\theta = \pi/12$. **b** $\theta = \pi/6$. **c** $\theta = \pi/3$. **d** $\theta = \pi/2$

One problem with the randomized CBE method is that it does not utilize the underlying data distribution while generating the matrix $\mathbf{R}$. In the next section, we propose to learn $\mathbf{R}$ in a data-dependent fashion, to minimize the distortions due to circulant projection and binarization.

16.3.2 Learning Circulant Binary Embedding (CBE-opt)

We propose data-dependent CBE (CBE-opt), by optimizing the projection matrix with a novel time–frequency alternating optimization. We consider the following objective function in learning the d-bit CBE. The extension of learning $k < d$ bits will be shown in Sect. 16.3.3.

$$\begin{aligned} \underset{\mathbf{B},\mathbf{r}}{\operatorname{argmin}} \quad & ||\mathbf{B} - \mathbf{Z}\mathbf{R}^T||_F^2 + \lambda ||\mathbf{R}\mathbf{R}^T - \mathbf{I}||_F^2 \\ \text{s.t.} \quad & \mathbf{R} = \operatorname{circ}(\mathbf{r}), \end{aligned} \tag{16.22}$$

where $\mathbf{Z} \in \mathbb{R}^{n \times d}$ is the data matrix containing n training points: $\mathbf{Z} = [\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}]^T$, and $\mathbf{B} \in \{-1, 1\}^{n \times d}$ is the corresponding binary code matrix.[8]

In the above optimization, the first term minimizes distortion due to binarization. The second term tries to make the projections (rows of $\mathbf{R}$, and hence the corresponding bits) as uncorrelated as possible. In other words, this helps to reduce the redundancy in the learned code. If $\mathbf{R}$ were to be an orthogonal matrix, the second term will vanish and the optimization would find the best rotation such that the distortion due to binarization is minimized. However, when $\mathbf{R}$ is a circulant matrix, $\mathbf{R}$, in general, will not be orthogonal. Similar objective has been used in previous works including [6, 7] and [34].

The above is a combinatorial optimization problem, for which an optimal solution is hard to find. In this section we propose a novel approach to efficiently find a local solution. The idea is to alternatively optimize the objective by fixing $\mathbf{r}$, and $\mathbf{B}$, respectively. For a fixed $\mathbf{r}$, optimizing $\mathbf{B}$ can be easily performed in the input domain ("time" as opposed to "frequency"). For a fixed $\mathbf{B}$, the circulant structure of $\mathbf{R}$ makes it difficult to optimize the objective in the input domain. Hence we propose a novel method, by optimizing $\mathbf{r}$ in the frequency domain based on DFT. This leads to a very efficient procedure.

For a fixed r. The objective is independent on each element of $\mathbf{B}$. Denote B_{ij} as the element of the ith row and jth column of $\mathbf{B}$. It is easy to show that $\mathbf{B}$ can be updated as:

$$B_{ij} = \begin{cases} 1 & \text{if } \mathbf{R}_{j\cdot}\mathbf{x}_i \geq 0 \\ -1 & \text{if } \mathbf{R}_{j\cdot}\mathbf{x}_i < 0 \end{cases}, \qquad (16.23)$$
$$i = 0, \ldots, n-1. \quad j = 0, \ldots, d-1.$$

For a fixed B. Define $\tilde{\mathbf{r}}$ as the DFT of the circulant vector $\tilde{\mathbf{r}} := \mathcal{F}(\mathbf{r})$. Instead of solving $\mathbf{r}$ directly, we propose to solve $\tilde{\mathbf{r}}$, from which $\mathbf{r}$ can be recovered by IDFT.

Key to our derivation is the fact that DFT projects the signal to a set of orthogonal basis. Therefore, the ℓ_2 norm can be preserved. Formally, according to Parseval's theorem, for any $\mathbf{t} \in \mathbb{C}^d$ [25],

$$||\mathbf{t}||_2^2 = (1/d)||\mathcal{F}(\mathbf{t})||_2^2.$$

Denote diag$(\cdot)$ as the diagonal matrix formed by a vector. Denote $\Re(\cdot)$ and $\Im(\cdot)$ as the real and imaginary parts, respectively. We use $\mathbf{B}_{i\cdot}$ to denote the ith row of $\mathbf{B}$. With complex arithmetic, the first term in (16.22) can be expressed in the frequency domain as:

[8] If the data is ℓ_2 normalized, we can set $\mathbf{B} \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^{n \times d}$ to make $\mathbf{B}$ and $\mathbf{Z}\mathbf{R}^T$ more comparable. This does not empirically influence the performance.

$$\begin{aligned}||\mathbf{B}-\mathbf{X}\mathbf{R}^T||_F^2 &= \frac{1}{d}\sum_{i=0}^{n-1}||\mathcal{F}(\mathbf{B}_{i\cdot}^T-\mathbf{R}\mathbf{x}_i)||_2^2\\ &= \frac{1}{d}\sum_{i=0}^{n-1}||\mathcal{F}(\mathbf{B}_{i\cdot}^T)-\tilde{\mathbf{r}}\circ\mathcal{F}(\mathbf{x}_i)||_2^2\\ &= \frac{1}{d}\sum_{i=0}^{n-1}||\mathcal{F}(\mathbf{B}_{i\cdot}^T)-\mathrm{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}||_2^2\\ &= \frac{1}{d}\sum_{i=0}^{n-1}\Big(\mathcal{F}(\mathbf{B}_{i\cdot}^T)-\mathrm{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}\Big)^T\Big(\mathcal{F}(\mathbf{B}_{i\cdot}^T)-\mathrm{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}\Big)\\ &= \frac{1}{d}\Big[\Re(\tilde{\mathbf{r}})^T\mathbf{M}\Re(\tilde{\mathbf{r}})+\Im(\tilde{\mathbf{r}})^T\mathbf{M}\Im(\tilde{\mathbf{r}})+\Re(\tilde{\mathbf{r}})^T\mathbf{h}+\Im(\tilde{\mathbf{r}})^T\mathbf{g}\Big]+||\mathbf{B}||_F^2,\end{aligned} \tag{16.24}$$

where,

$$\begin{aligned}\mathbf{M} &= \mathrm{diag}\Big(\sum_{i=0}^{n-1}\Re(\mathcal{F}(\mathbf{x}_i))\circ\Re(\mathcal{F}(\mathbf{x}_i))+\Im(\mathcal{F}(\mathbf{x}_i))\circ\Im(\mathcal{F}(\mathbf{x}_i))\Big)\\ \mathbf{h} &= -2\sum_{i=0}^{n-1}\Re(\mathcal{F}(\mathbf{x}_i))\circ\Re(\mathcal{F}(\mathbf{B}_{i\cdot}^T))+\Im(\mathcal{F}(\mathbf{x}_i))\circ\Im(\mathcal{F}(\mathbf{B}_{i\cdot}^T))\\ \mathbf{g} &= 2\sum_{i=0}^{n-1}\Im(\mathcal{F}(\mathbf{x}_i))\circ\Re(\mathcal{F}(\mathbf{B}_{i\cdot}^T))-\Re(\mathcal{F}(\mathbf{x}_i))\circ\Im(\mathcal{F}(\mathbf{B}_{i\cdot}^T)).\end{aligned}$$

The above can be derived based on the following fact. For any $\mathbf{Q}\in\mathbb{C}^{d\times d}$, $\mathbf{s}$, $\mathbf{t}\in\mathbb{C}^d$,

$$\begin{aligned}||\mathbf{s}-\mathbf{Q}\mathbf{t}||_2^2 &= (\mathbf{s}-\mathbf{Q}\mathbf{t})^H(\mathbf{s}-\mathbf{Q}\mathbf{t}) \qquad (16.25)\\ &= \mathbf{s}^H\mathbf{s}-\mathbf{s}^H\mathbf{Q}\mathbf{t}-\mathbf{t}^H\mathbf{Q}^H\mathbf{s}+\mathbf{t}^H\mathbf{Q}^H A\mathbf{t}\\ &= \Re(\mathbf{s})^T\Re(\mathbf{s})+\Im(\mathbf{s})^T\Im(\mathbf{s})\\ &\quad -2\Re(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{s})+\Im(\mathbf{Q})^T\Im(\mathbf{s}))\\ &\quad +2\Im(\mathbf{t})^T(\Im(\mathbf{Q})^T\Re(\mathbf{s})-\Re(\mathbf{Q})^T\Im(\mathbf{s}))\\ &\quad +\Re(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{Q})+\Im(\mathbf{Q})^T\Im(\mathbf{Q}))\Re(\mathbf{t})\\ &\quad +\Im(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{Q})+\Im(\mathbf{Q})^T\Im(\mathbf{Q}))\Im(\mathbf{t})\\ &\quad +2\Re(\mathbf{t})^T(\Im(\mathbf{Q})^T\Re(\mathbf{Q})-\Re(\mathbf{Q})^T\Im(\mathbf{Q}))\Im(\mathbf{t}).\end{aligned}$$

For the second term in (16.22), we note that the circulant matrix can be diagonalized by DFT matrix $\mathbf{F}_d$ and its conjugate transpose $\mathbf{F}_d^H$. Formally, for $\mathbf{R} = \text{circ}(\mathbf{r})$, $\mathbf{r} \in \mathbb{R}^d$,

$$\mathbf{R} = (1/d)\mathbf{F}_d^H \text{diag}(\mathcal{F}(\mathbf{r}))\mathbf{F}_d. \tag{16.26}$$

Let $\text{Tr}(\cdot)$ be the trace of a matrix. Therefore,

$$\begin{aligned}
||\mathbf{R}\mathbf{R}^T - \mathbf{I}||_F^2 &= ||\frac{1}{d}\mathbf{F}_d^H(\text{diag}(\tilde{\mathbf{r}})^H \text{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\mathbf{F}_d||_F^2 \\
&= \text{Tr}\left[\frac{1}{d}\mathbf{F}_d^H(\text{diag}(\tilde{\mathbf{r}})^H \text{diag}(\tilde{\mathbf{r}}) - \mathbf{I})^H(\text{diag}(\tilde{\mathbf{r}})^H \text{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\mathbf{F}_d\right] \\
&= \text{Tr}\left[(\text{diag}(\tilde{\mathbf{r}})^H \text{diag}(\tilde{\mathbf{r}}) - \mathbf{I})^H(\text{diag}(\tilde{\mathbf{r}})^H \text{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\right] \\
&= ||\tilde{\mathbf{r}}^H \circ \tilde{\mathbf{r}} - \mathbf{1}||_2^2 = ||\Re(\tilde{\mathbf{r}})^2 + \Im(\tilde{\mathbf{r}})^2 - \mathbf{1}||_2^2.
\end{aligned} \tag{16.27}$$

Furthermore, as $\mathbf{r}$ is real-valued, additional constraints on $\tilde{\mathbf{r}}$ are needed. For any $u \in \mathbb{C}$, denote $\overline{u}$ as the complex conjugate of u. We have the following result [25]: For any real-valued vector $\mathbf{t} \in \mathbb{C}^d$, $\mathcal{F}(\mathbf{t})_0$ is real-valued, and

$$\mathcal{F}(\mathbf{t})_{d-i} = \overline{\mathcal{F}(\mathbf{t})_i}, \quad i = 1, \ldots, \lfloor d/2 \rfloor.$$

From (16.24) to (16.27), the problem of optimizing $\tilde{\mathbf{r}}$ becomes

$$\begin{aligned}
\underset{\tilde{\mathbf{r}}}{\text{argmin}} \quad & \Re(\tilde{\mathbf{r}})^T\mathbf{M}\Re(\tilde{\mathbf{r}}) + \Im(\tilde{\mathbf{r}})^T\mathbf{M}\Im(\tilde{\mathbf{r}}) + \Re(\tilde{\mathbf{r}})^T\mathbf{h} \\
& + \Im(\tilde{\mathbf{r}})^T\mathbf{g} + \lambda d||\Re(\tilde{\mathbf{r}})^2 + \Im(\tilde{\mathbf{r}})^2 - \mathbf{1}||_2^2 \\
\text{s.t.} \quad & \Im(\tilde{r}_0) = 0 \\
& \Re(\tilde{r}_i) = \Re(\tilde{r}_{d-i}), \quad i = 1, \ldots, \lfloor d/2 \rfloor \\
& \Im(\tilde{r}_i) = -\Im(\tilde{r}_{d-i}), \quad i = 1, \ldots, \lfloor d/2 \rfloor.
\end{aligned} \tag{16.28}$$

The above is nonconvex. Fortunately, the objective function can be decomposed, such that we can solve two variables at a time. Denote the diagonal vector of the diagonal matrix $\mathbf{M}$ as $\mathbf{m}$. The above optimization can then be decomposed to the following sets of optimizations.

$$\underset{\tilde{r}_0}{\text{argmin}} \quad m_0\tilde{r}_0^2 + h_0\tilde{r}_0 + \lambda d\left(\tilde{r}_0^2 - 1\right)^2, \quad \text{s.t. } \tilde{r}_0 = \overline{\tilde{r}_0}. \tag{16.29}$$

$$\begin{aligned}
\underset{\tilde{r}_i}{\text{argmin}} \quad & (m_i + m_{d-i})(\Re(\tilde{r}_i)^2 + \Im(\tilde{r}_i)^2) \\
& + 2\lambda d\left(\Re(\tilde{r}_i)^2 + \Im(\tilde{r}_i)^2 - 1\right)^2 \\
& + (h_i + h_{d-i})\Re(\tilde{r}_i) + (g_i - g_{d-i})\Im(\tilde{r}_i), \\
& i = 1, \ldots, \lfloor d/2 \rfloor.
\end{aligned} \tag{16.30}$$

In (16.29), we need to minimize a 4th order polynomial with one variable, with the closed form solution readily available. In (16.30), we need to minimize a 4th order polynomial with two variables. Though the closed form solution is hard (requiring solution of a cubic bivariate system), we can find local minima by gradient descent, which can be considered as having constant running time for such small-scale problems. The overall objective is guaranteed to be nonincreasing in each step. In practice, we can get a good solution with just 5–10 iterations. In summary, the proposed time–frequency alternating optimization procedure has running time $\mathcal{O}(nd\log d)$.

16.3.3 Learning with Dimension Reduction

In the case of learning $k < d$ bits, we need to solve the following optimization problem:

$$\begin{aligned} \underset{\mathbf{B},\mathbf{r}}{\text{argmin}} \quad & ||\mathbf{B}\mathbf{P}_k - \mathbf{X}\mathbf{P}_k^T\mathbf{R}^T||_F^2 + \lambda||\mathbf{R}\mathbf{P}_k\mathbf{P}_k^T\mathbf{R}^T - \mathbf{I}||_F^2 \\ \text{s.t.} \quad & \mathbf{R} = \text{circ}(\mathbf{r}), \end{aligned} \tag{16.31}$$

in which $\mathbf{P}_k = \begin{bmatrix} \mathbf{I}_k & \mathbf{O} \\ \mathbf{O} & \mathbf{O}_{d-k} \end{bmatrix}$, $\mathbf{I}_k$ is a $k \times k$ identity matrix. $\mathbf{O}_{d-k}$ is a $(d-k) \times (d-k)$ all-zero matrix, and $\mathbf{O}$ is a $k \times (d-k)$ all-zero matrix.

In fact, the right multiplication of $\mathbf{P}_k$ can be understood as a "temporal cut-off", which is equivalent to a frequency domain convolution. This makes the optimization difficult, as the objective in frequency domain can no longer be decomposed. To address this issue, we propose a simple solution in which $B_{ij} = 0$, $i = 0, \ldots, n-1$, $j = k, \ldots, d-1$ in (16.22). Thus, the optimization procedure remains the same, and the cost is also $\mathcal{O}(nd\log d)$. We will show in experiments that this heuristic provides good performance in practice.

16.4 Experiments

To demonstrate the performance of the proposed binary embedding methods, we conducted experiments on three real-world high-dimensional datasets used by the current state-of-the-art method for generating binary codes. The Flickr-25600 dataset contains 100 K images sampled from a noisy Internet image collection. Each image is represented by a 25,600 dimensional vector. The ImageNet-51200 contains 100 K images sampled from 100 random classes from ImageNet [4], each represented by a 51,200 dimensional vector. The third dataset (ImageNet-25600) is another random subset of ImageNet containing 100 K images in 25,600 dimensional space. All the vectors are normalized to be of unit norm. Most of the experiment results in this section have been presented in our former work [38].

We compared the performance of the randomized (bilinear-rand, CBE-rand) and learned (bilinear-opt, CBE-opt) versions of our embedding methods with the widely used method for high-dimensional data, i.e., LSH. Note that bilinear binary embeddings have been shown to perform similar or better than another promising technique called Product Quantization [14]. We also show an experiment with relatively low-dimensional data in 2,048 dimensional space using Flickr data to compare against techniques that perform well for low-dimensional data but do not scale to high-dimensional scenario. Example techniques include ITQ [7], SH [36], SKLSH [30], and AQBC [5].

Following [6, 8, 23], we use 10,000 randomly sampled instances for training. We then randomly sample 500 instances, different from the training set as queries. The performance (recall@1–100) is evaluated by averaging the recalls of the query instances. The ground truth of each query instance is defined as its 10 nearest neighbors based on ℓ_2 distance. For each dataset, we conduct two sets of experiments: *fixed time* where code generation time is fixed and *fixed bits* where the number of bits is fixed across all techniques. We also show an experiment where the binary codes are used for classification. For the bilinear method, in order to get fast computation, the feature vector is reshaped to a near-square matrix, and the dimension of the two bilinear projection matrices are also chosen to be close to square. Parameters for other techniques are tuned to give the best results on these datasets.

Computational Time. When generating k-bit code for d-dimensional data, the full projection, bilinear projection, and circulant projection methods have time complexity $O(kd)$, $O(\sqrt{k}d)$, and $O(d \log d)$, respectively. We compare the computational time in Table 16.2 on a fixed hardware. Based on our implementation, the computational time of the above three methods can be roughly characterized as $d^2 : d\sqrt{d} : 5d \log d$. Note that faster implementation of FFT algorithms will lead to better computational time for CBE by further reducing the constant factor. Due to the small storage requirement $\mathcal{O}(d)$, and the wide availability of highly optimized FFT libraries, CBE is also suitable for implementation on GPU. Our preliminary

Table 16.2 Computational time (ms) of full projection (LSH, ITQ, SH etc.), bilinear projection (BBE), and circulant projection (CBE)

d	Full projection	Bilinear projection	Circulant projection
2^{15}	5.44×10^2	2.85	1.11
2^{17}	–	1.91×10^1	4.23
2^{20} (1M)	–	3.76×10^2	3.77×10^1
2^{24}	–	1.22×10^4	8.10×10^2
2^{27} (100M)	–	2.68×10^5	8.15×10^3

The time is based on a single 2.9 GHz CPU core. The error is within 10 %. An empty cell indicates that the memory needed for that method is larger than the machine limit of 24 GB tests based on GPU shows up to 20 times speedup compared to CPU. In this paper, for fair comparison, we use same CPU-based implementation for all the methods.

In addition, the optimizations of learning-based CBE (Sect. 16.3.2) can be easily solved in a parallel fashion. The small footprints of both BBE and CBE also make them suitable to be implemented on mobile devices, which have strict memory requirement.

Retrieval. The recall for different methods is compared on the three datasets in Figs. 16.3, 16.5, and 16.7 shows the performance for different methods when the code generation time for all the methods is kept the same as that of CBE. For a fixed time, both CBE and BBE significantly outperform LSH. And CBE outperforms BBE in such high-dimensional settings. Even CBE-rand outperforms LSH and Bilinear code by a large margin.

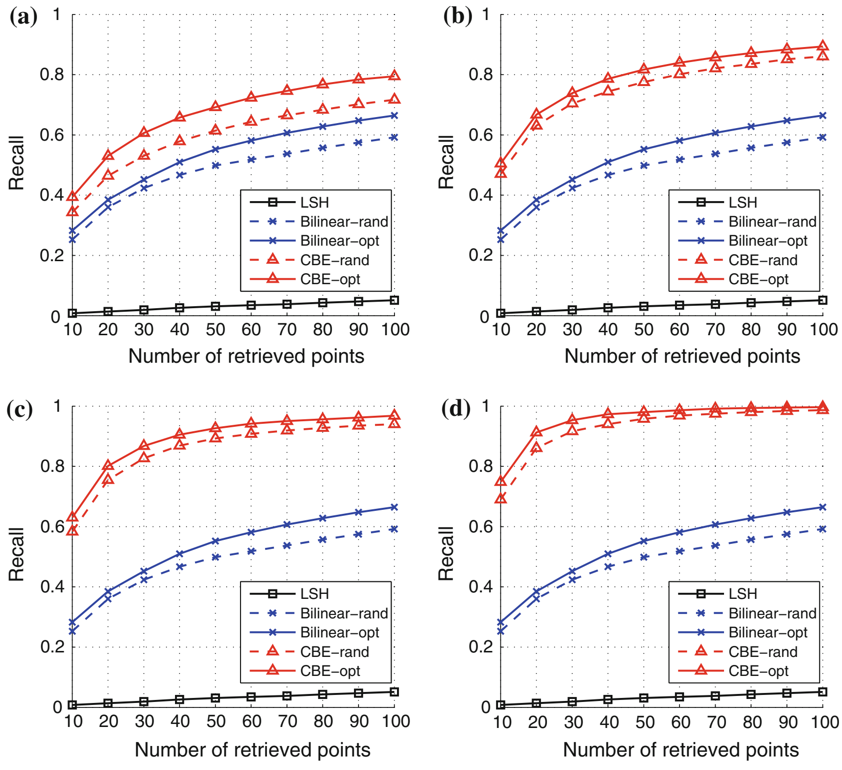


Fig. 16.3 Recall on Flickr-25600 with fixed time. "# bits" is the number of bits of CBE. Other methods are using less bits to make their computational time identical to CBE. The standard deviation is within 1 %. **a** # bits (CBE) = 3,200. **b** # bits (CBE) = 6,400. **c** # bits (CBE) = 12,800. **d** # bits (CBE) = 25,600

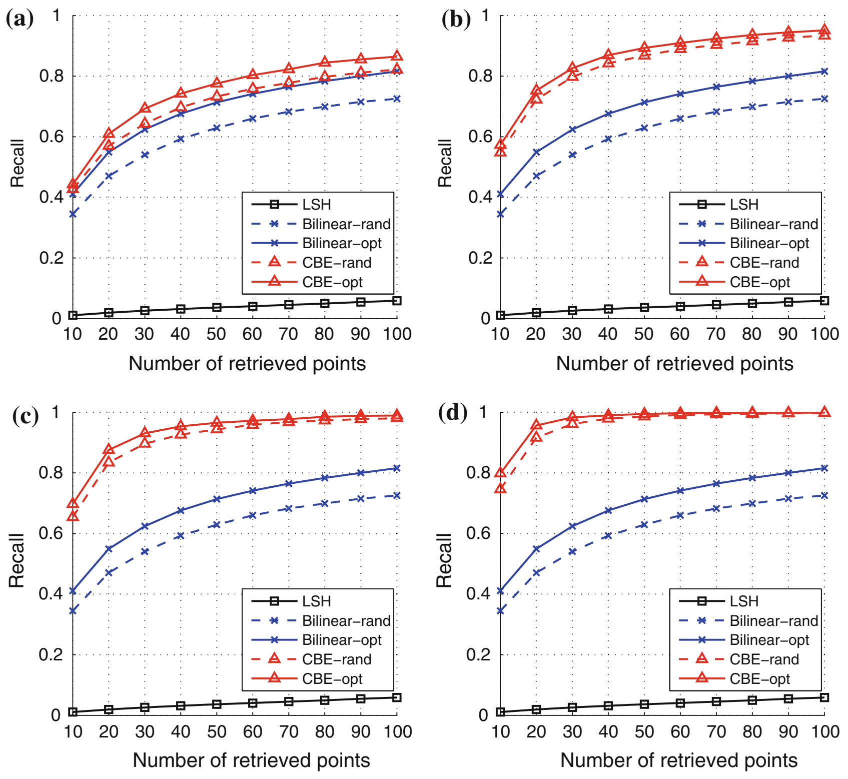


Fig. 16.4 Recall on Flickr-25600 with fixed number of bits. CBE-opt/CBE-rand are 2–3 times faster than Bilinear-opt/Bilinear-rand. Both CBE and BBE(Bilinear) are hundreds of times faster than LSH. The standard deviation is within 1 %. **a** # bits (all) = 3,200. **b** # bits (all) = 6,400. **c** # bits (all) = 12,800. **d** # bits (all) = 25,600

Figures 16.4, 16.6, and 16.8 compare the performance of different techniques with codes of same length. In this case, the performance of CBE-rand is almost identical to LSH even though it is hundreds of times faster. This is consistent with our analysis in Sect. 16.3.1. Bilinear-rand is also very competitive to LSH. In addition, CBE-opt/CBE-rand outperform the Bilinear-opt/Bilinear-rand in addition to being 2–3 times faster.

Classification. Besides retrieval, we also test the binary codes for classification. The advantage is to save on storage allowing even large-scale datasets to fit in memory [19, 31]. We follow the asymmetric setting of [31] by training linear SVM on binary code sgn($\mathbf{Rx}$), and testing on the original $\mathbf{Rx}$. This empirically has been shown to give better accuracy than the symmetric procedure. We use ImageNet-25600, with randomly sampled 100 images per category for training, 50 for validation, and 50 for

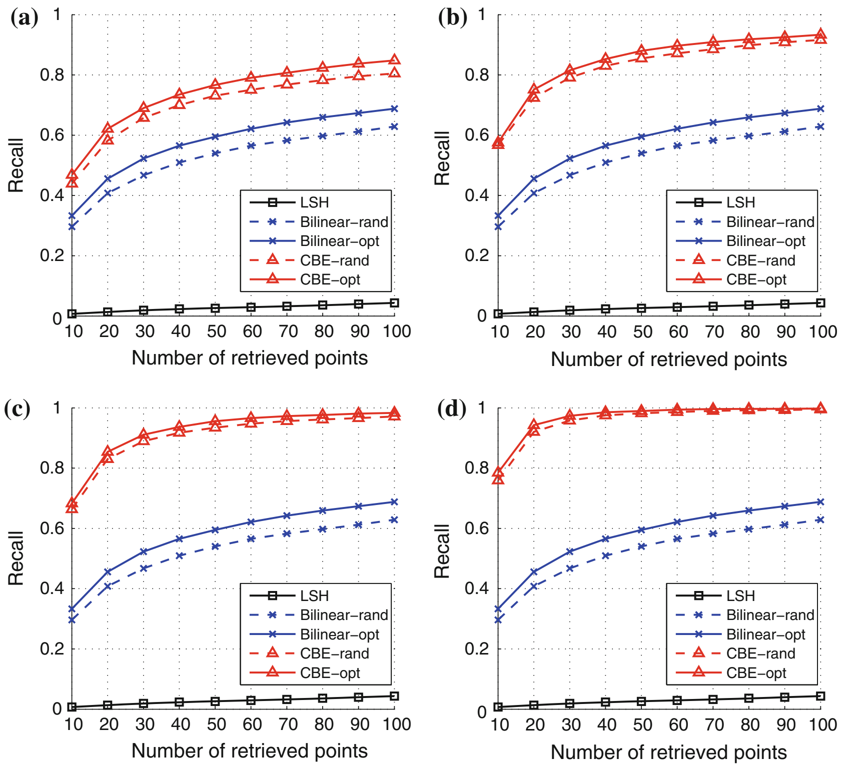


Fig. 16.5 Recall on ImageNet-25600 with fixed time. "# bits" is the number of bits of CBE. Other methods are using less bits to make their computational time identical to CBE. The standard deviation is within 1 %. **a** # bits (CBE) = 3,200. **b** # bits (CBE) = 6,400. **c** # bits (CBE) = 12,800. **d** # bits (CBE) = 25,600

testing. The code dimension is set as 25,600. As shown in Table 16.3, our methods, which have much faster computation, does not show any performance degradation compared to LSH in classification task as well.

Low-Dimensional Experiment. There exist several techniques that do not scale to high-dimensional case. To compare our method with those, we conducted experiments with fixed number of bits on a relatively low-dimensional dataset (Flickr-2048), constructed by randomly sampling 2,048 dimensions of Flickr-25600. As shown in Fig. 16.9, though BBE and CBE are not designed for such scenario, they perform better or equivalent to other techniques except ITQ which scales very poorly with $d(\mathcal{O}(d^3))$. Moreover, as the number of bits increases, the gap between ITQ and our methods becomes much smaller suggesting that the performance of ITQ isnot

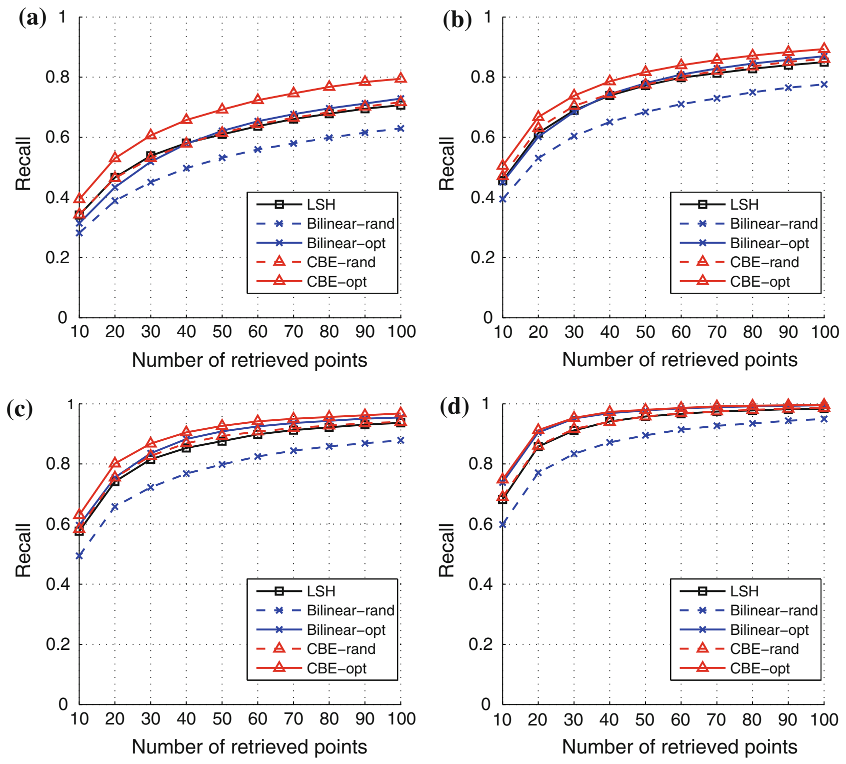


Fig. 16.6 Recall on ImageNet-25600 with fixed number of bits. CBE-opt/CBE-rand are 2–3 times faster than Bilinear-opt/Bilinear-rand. Both CBE and BBE(Bilinear) are hundreds of times faster than LSH. The standard deviation is within 1 %. **a** # bits (all) = 3,200. **b** # bits (all) = 6,400. **c** # bits (all) = 12,800. **d** # bits (all) = 25,600

expected to be better if one could run ITQ on high-dimensional data. Note that in such small-scale scenario, BBE is faster than CBE due to the computational overhead of FFT.

16.5 Choice of Algorithms

CBE has better computational complexity compared to BBE. In addition, according to the experimental results, with fixed bits, CBE outperforms BBE in general. This suggests that the circulant projection is more powerful than the bilinear projection in generating the binary code. This resonates with the works using circulant projections for Johnson-Lindenstrauss transformations [1, 3].

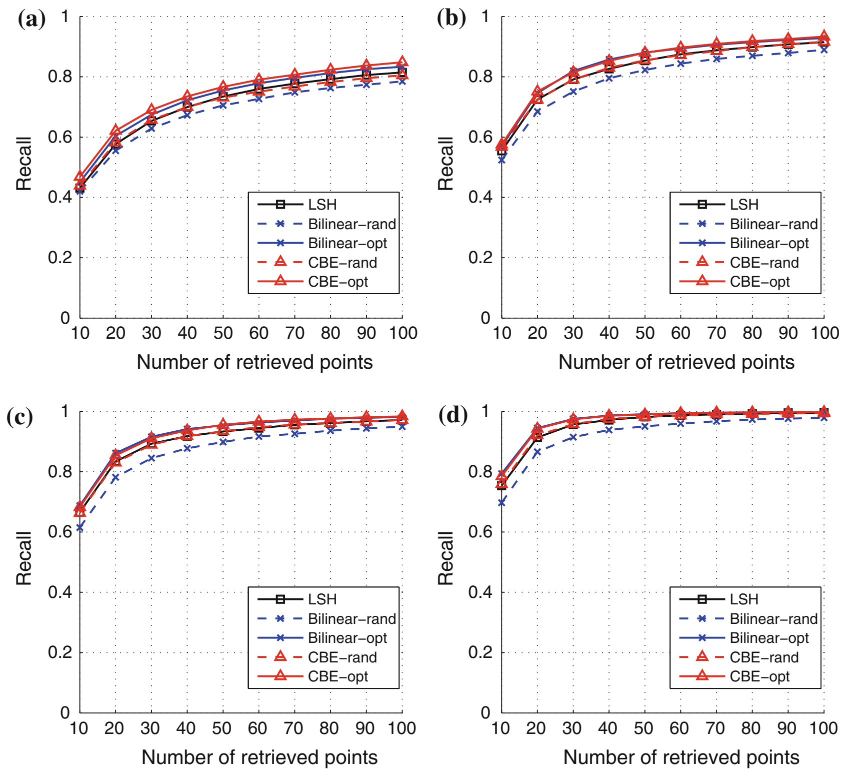


Fig. 16.7 Recall on ImageNet-51200, with fixed time. "# bits" is the number of bits of CBE. Other methods are using less bits to make their computational time identical to CBE. The standard deviation is within 1 %. **a** # bits (CBE) = 6,400. **b** # bits (CBE) = 12,800. **c** # bits (CBE) = 25,600. **d** # bits (CBE) = 51,200

Table 16.3 Felix et al. [38] multiclass classification accuracy on binary-coded ImageNet-25600. The binary codes of same dimensionality are 32 times more space-efficient than the original features (single-float)

Original	LSH	Bilinear-opt	CBE-opt
25.59 ± 0.33	23.49 ± 0.24	24.02 ± 0.35	24.55 ± 0.30

The disadvantage of CBE is the computational overhead of FFT. Based on our implementation, BBE is faster for moderate to high-dimensional data (10–30 k), and CBE is faster on very high-dimensional data (30–100 M). Therefore, the final choice of the algorithm should depend on the evaluation metric, and the actual computational cost based on implementation.

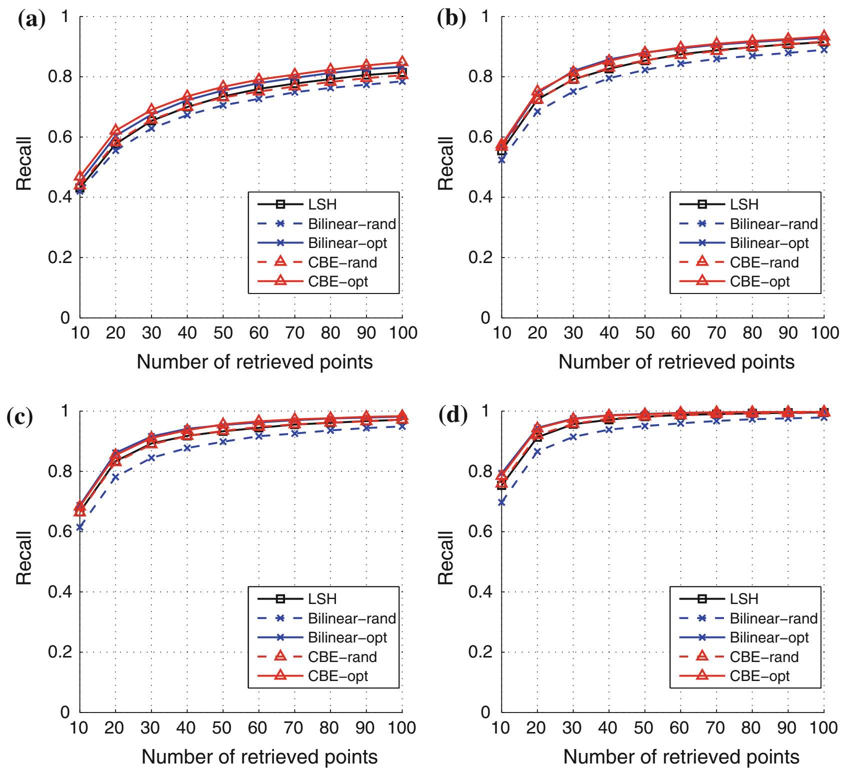


Fig. 16.8 Recall on ImageNet-51200 with fixed number of bits. CBE-opt/CBE-rand are 2–3 times faster than Bilinear-opt/Bilinear-rand. Both CBE and BBE(Bilinear) are hundreds of times faster than LSH. The standard deviation is within 1 %. **a** # bits (all) = 6,400. **b** # bits (all) = 12,800. **c** # bits (all) = 25,600. **d** # bits (all) = 51,200

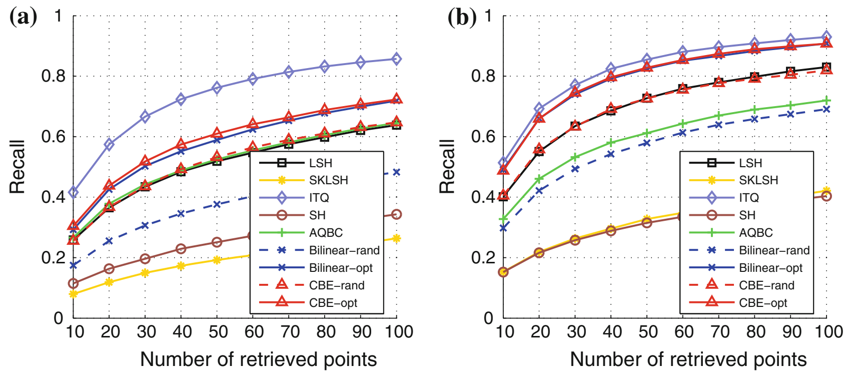


Fig. 16.9 Performance comparison on relatively low-dimensional data (Flickr-2048) with fixed number of bits. CBE gives comparable performance to the state of the art even on low-dimensional data as the number of bits is increased. However, note that these other methods do not scale to very high-dimensional datasetting which is the main focus of this work. **a** # bits = 1,024. **b** # bits = 2,048

16.6 Conclusion

This book chapter introduces two fast binary embedding methods for high-dimensional data [6, 38] with unified notations and framework. The Bilinear Binary Embedding (BBE) has time complexity $\mathcal{O}(d^{1.5})$. The Circulant Binary Embedding (CBE) has time complexity $\mathcal{O}(d \log d)$. Both algorithms have space complexity $\mathcal{O}(d)$. We have also proposed methods for learning the projection matrices in a data-dependent fashion to further improve the performance. The proposed methods show no performance degradation on real-world data compared to the expensive full projection methods, which has computational complexity $\mathcal{O}(d^2)$. On the contrary, with the fixed time, our methods showed significant accuracy gains.

Both proposed methods use highly structured projections to speed up the computation. Our future work is to study more generalized structured projections for binary embedding. This requires both theoretical analysis on the randomized projections, and novel optimization algorithms for learning data-dependent projections.

References

1. Ailon N, Chazelle B (2006) Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In: ACM symposium on theory of computing
2. Charikar MS (2002) Similarity estimation techniques from rounding algorithms. In: ACM symposium on theory of computing
3. Dasgupta A, Kumar R, Sarlós T (2011) Fast locality-sensitive hashing. In: ACM SIGKDD conference on knowledge discovery and data mining
4. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: Computer vision and pattern recognition
5. Gong Y, Kumar S, Verma V, Lazebnik S (2012) Angular quantization-based binary codes for fast similarity search. In: Advances in neural information processing systems
6. Gong Y, Kumar S, Rowley HA, Lazebnik S (2013) Learning binary codes for high-dimensional data using bilinear projections. In: Computer vision and pattern recognition
7. Gong Y, Lazebnik S, Gordo A, Perronnin F (2013) Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. IEEE Trans Pattern Anal Mach Intell 35(12):2916–2929
8. Gordo A, Perronnin F (2011) Asymmetric distances for binary embeddings. In: Computer vision and pattern recognition
9. Gray RM (2006) Toeplitz and circulant matrices: a review. Now Publishers, Boston
10. He J, Radhakrishnan R, Chang S-F, Bauer C (2011) Compact hashing with joint optimization of search accuracy and time. In: Computer vision and pattern recognition
11. Heo J-P, Lee Y, He J, Chang S-F, Yoon S-E (2012) Spherical hashing. In: Computer vision and pattern recognition
12. Hinrichs A, Vybíral J (2011) Johnson-Lindenstrauss lemma for circulant matrices. Random Struct Algorithms 39(3):391–398
13. Jain P, Kulis B, Grauman K (2008) Fast image search for learned metrics. In: Computer vision and pattern recognition
14. Jégou H, Douze M, Schmid C (2011) Product quantization for nearest neighbor search. IEEE Trans Pattern Anal Mach Intell 33(1):117–128
15. Jégou H, Douze M, Schmid C, Patrick P (2010) Aggregating local descriptors into a compact image representation. In: Computer vision and pattern recognition

16. Krahmer F, Ward R (2011) New and improved Johnson-Lindenstrauss embeddings via the restricted isometry property. SIAM J Math Anal 43(3):1269–1281
17. Kulis B, Darrell T (2009) Learning to hash with binary reconstructive embeddings. In: Advances in neural information processing systems
18. Laub AJ (2004) Matrix analysis for scientists and engineers. SIAM
19. Li P, Shrivastava A, Moore J, Konig AC (2011) Hashing algorithms for large-scale learning. In: Advances in neural information processing systems
20. Liu W, Wang J, Kumar S, Chang S-F (2011) Hashing with graphs. In: International conference on machine learning
21. Liu W, Wang J, Ji R, Jiang Y-G, Chang S-F (2012) Supervised hashing with kernels. In: Computer vision and pattern recognition
22. Liu W, Wang J, Mu Y, Kumar S, Chang S-F (2012) Compact hyperplane hashing with bilinear functions. In: International conference on machine learning
23. Norouzi M, Fleet D (2012) Minimal loss hashing for compact binary codes. In: International conference on machine learning
24. Norouzi M, Fleet D, Salakhutdinov R (2012) Hamming distance metric learning. In: Advances in neural information processing systems
25. Oppenheim AV, Schafer RW, Buck JR et al (1999) Discrete-time signal processing, vol 5. Prentice Hall, Upper Saddle River
26. Perronnin F, Dance CR (2007) Fisher kernels on visual vocabularies for image categorization. In: Computer vision and pattern recognition
27. Perronnin F, Liu Y, Sánchez J, Poirier H (2010) Large-scale image retrieval with compressed fisher vectors. In: Computer vision and pattern recognition
28. Perronnin F, Sánchez J, Mensink T (2010) Improving the Fisher kernel for large-scale image classification
29. Pirsiavash H, Ramanan D, Fowlkes C (2009) Bilinear classifiers for visual recognition. In: Advances in neural information processing systems
30. Raginsky M, Lazebnik S (2009) Locality-sensitive binary codes from shift-invariant kernels. In: Advances in neural information processing systems
31. Sánchez J, Perronnin F (2011) High-dimensional signature compression for large-scale image classification. In: Computer vision and pattern recognition
32. Schönemann PH (1968) On two-sided orthogonal Procrustes problems. Psychometrika
33. Vybíral J (2011) A variant of the Johnson-Lindenstrauss lemma for circulant matrices. J Funct Anal 260(4):1096–1105
34. Wang J, Kumar S, Chang S-F (2010) Sequential projection learning for hashing with compact codes. In: International conference on machine learning
35. Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y (2010) Locality-constrained linear coding for image classification. In: Computer vision and pattern recognition
36. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing. In: Advances in neural information processing systems
37. Yu FX, Ji R, Tsai M-H, Ye G, Ye G, Chang S-F (2012) Weak attributes for large-scale image retrieval. In: Computer vision and pattern recognition
38. Yu FX, Kumar S, Gong Y, Chang S-F (2014) Circulant binary embedding. In: International conference on machine learning
39. Zhang H, Cheng L (2013) New bounds for circulant Johnson-Lindenstrauss embeddings. arXiv preprint arXiv:1308.6339

Chapter 17
Fast Approximate K-Means via Cluster Closures

Jingdong Wang, Jing Wang, Qifa Ke, Gang Zeng and Shipeng Li

Abstract K-means, a simple and effective clustering algorithm, is one of the most widely used algorithms in multimedia and computer vision community. Traditional k-means is an iterative algorithm—in each iteration new cluster centers are computed and each data point is re-assigned to its nearest center. The cluster re-assignment step becomes prohibitively expensive when the number of data points and cluster centers are large. In this chapter, we propose a novel approximate k-means algorithm to greatly reduce the computational complexity in the assignment step. Our approach is motivated by the observation that most active points changing their cluster assignments at each iteration are located on or near cluster boundaries. The idea is to efficiently identify those active points by pre-assembling the data into groups of neighboring points using multiple random spatial partition trees, and to use the neighborhood information to construct a closure for each cluster, in such a way only a small number of cluster candidates need to be considered when assigning a data point to its nearest cluster. Using complexity analysis, image data clustering, and applications to image retrieval, we show that our approach out-performs state-of-the-art approximate k-means algorithms in terms of clustering quality and efficiency.

J. Wang (✉) · S. Li
Microsoft, Building 2, No. 5 Dan Ling Street, Beijing 100080, Haidian District, China
e-mail: jingdw@microsoft.com

S. Li
e-mail: spli@microsoft.com

J. Wang
Key Laboratory on Machine Perception, Peking University,
Science Building No. 2, Beijing 1000871, China
e-mail: cis.wangjing@pku.edu.cn

Q. Ke
Microsoft, 1020 Enterprise Way, Sunnyvale, CA 94089, USA
e-mail: qke@microsoft.com

G. Zeng
Key Laboratory on Machine Perception, Peking University,
Room 2202, Science Building No. 2, Beijing 1000871, China
e-mail: g.zeng@ieee.org

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_17

17.1 Introduction

K-means [13] has been widely used in multimedia, computer vision and machine learning for clustering and vector quantization. In large-scale image retrieval, it is advantageous to learn a large codebook containing one million or more entries [17, 19, 24], which requires clustering tens or even hundreds of millions of high-dimensional feature descriptors into one million or more clusters. Another emerging application of large-scale clustering is to organize a large corpus of web images for various purposes such as web image browsing/exploring [27].

The standard k-means algorithm, Lloyd's algorithm [6, 12, 13], is an iterative refinement approach that greedily minimizes the sum of squared distances between each point and its assigned cluster center. It consists of two iterative steps, the assignment step and the update step. The assignment step aims to find the nearest cluster for each point by checking the distance between the point and each cluster center. The update step re-computes the cluster centers based on current assignments. When clustering n points into k clusters, the assignment step costs $O(nk)$. For applications with large nk, the assignment step in exact k-means becomes prohibitively expensive. Therefore many approximate solutions, such as hierarchial k-means (HKM) [17] and approximate k-means (AKM) [19], have been developed to improve the efficiency, but with a sacrifice of clustering accuracy.

In this chapter, we introduce a novel approximate k-means algorithm,[1] which makes a better tradeoff between the speed and the accuracy. Our approach is motivated by the observation that *active points*, defined as the points whose cluster assignments change in each iteration, often locate at or near boundaries of different clusters. The idea is to identify those active points at or near cluster boundaries to improve both the efficiency and accuracy in the assignment step of the k-means algorithm. We generate a neighborhood set for each data point by pre-assembling the data points using multiple random partition trees [26]. A *cluster closure* is then formed by expanding each point in the cluster into its neighborhood set, as illustrated in Fig. 17.2. When assigning a point $\mathbf{x}$ to its nearest cluster, we only need to consider those clusters that contain $\mathbf{x}$ in their closures. Typically a point belongs to a small number of cluster closures, thus the number of candidate clusters are greatly reduced in the assignment step.

We evaluate our algorithm by complexity analysis, the performance on clustering real data sets, and the performance of image retrieval applications with codebooks learned by clustering. Our proposed algorithm achieves significant improvements compared to the state-of-the-art, in both accuracy and running time. When clustering a real data set of 1 M 384-dimensional GIST features into 10 K clusters, our algorithm converges more than 2.5 faster than the state-of-the-art algorithms. In the image retrieval application on a standard dataset, our algorithm learns a codebook with

[1] A conference version appeared in [28].

750 K visual words that outperforms the codebooks with 1 M visual words learned by other state-of-the-art algorithms—even our codebook with 500 K visual words is superior over other codebooks with 1 M visual words.

17.2 Literature Review

Given a set of points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where each point is a d-dimensional vector, k-means clustering aims to partition these n points into k ($k \leqslant n$) groups, $\mathscr{G} = \{\mathscr{G}_1, \mathscr{G}_2, \ldots, \mathscr{G}_k\}$, by minimizing the within-cluster sum of squared distortions (WCSSD):

$$J(\mathscr{C}, \mathscr{G}) = \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in \mathscr{G}_j} \|\mathbf{x}_i - \mathbf{c}_j\|_2^2, \tag{17.1}$$

where $\mathbf{c}_j$ is the center of cluster $\mathscr{G}_j$, $\mathbf{c}_j = \frac{1}{|\mathscr{G}_j|} \sum_{\mathbf{x}_i \in \mathscr{G}_j} \mathbf{x}_i$, and $\mathscr{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$. In the following, we use *group* and *cluster* interchangeably.

17.2.1 Lloyd Algorithm

Minimizing the objective function in Eq. 17.1 is NP-hard in many cases [14]. Thus, various heuristic algorithms are used in practice, and k-means (or Lloyd's algorithm) [6, 12, 13] is the most commonly used algorithm. It starts from a set of k cluster centers (obtained from priors or random initialization) $\{\mathbf{c}_1^{(1)}, \ldots, \mathbf{c}_k^{(1)}\}$, and then proceeds by alternating the following two steps:

- **Assignment step**: Given the current set of k cluster centers, $\mathscr{C}^{(t)} = \{\mathbf{c}_1^{(t)}, \ldots, \mathbf{c}_k^{(t)}\}$, assign each point $\mathbf{x}_i$ to the cluster whose center is the closest to $\mathbf{x}_i$:

$$z_i^{(t+1)} = \arg\min_j \|\mathbf{x}_i - \mathbf{c}_j^{(t)}\|_2. \tag{17.2}$$

- **Update step**: Update the points in each cluster, $\mathscr{G}_j^{(t+1)} = \{\mathbf{x}_i | z_i^{(t+1)} = j\}$, and compute the new center for each cluster, $\mathbf{c}_j^{(t+1)} = \frac{1}{|\mathscr{G}_j^{(t+1)}|} \sum_{\mathbf{x}_i \in \mathscr{G}_j^{(t+1)}} \mathbf{x}_i$.

The computational complexity for the above assignment step and the update step is $O(ndk)$ and $O(nd)$, respectively. Various speedup algorithms have been developed by making the complexity of the assignment step less than the linear time (e.g., logarithmic time) with respect to n (the number of the data points), k (the number of clusters), and d (the dimension of the data pint). In the following, we present a short review mainly on handling large n and k.

17.2.2 Handling Large Data

Distance computation elimination. Various approaches have been proposed to speed up exact k-means. An accelerated algorithm is proposed by using the triangle inequality [3] and keeping track of lower and upper bounds for distances between points and centers to avoid unnecessary distance calculations but requires $O(k^2)$ extra storage, rendering it impractical for a large number of clusters.
Subsampling. An alternative solution to speed up k-means is based on sub-sampling the data points. One way is to run k-means over sub-sampled data points, and then to directly assign the remaining points to the clusters. An extension of the above solution is to optionally add the remaining points incrementally, and to rerun k-means to get a finer clustering. The former scheme is not applicable in many applications. As pointed in [19], it results in less accurate clustering and lower performance in image retrieval applications. The Coremeans algorithm [7] uses the latter scheme. It begins with a coreset and incrementally increases the size of the coreset. As pointed out in [7], Coremeans works well only for a small number of clusters. Consequently, those methods are not suitable for large-scale clustering problems, especially for problems with a large number of clusters.
Data organization. The approach in [9] presents a filtering algorithm. It begins by storing the data points in a k-d tree and maintains, for each node of the tree, a subset of candidate centers. The candidates for each node are pruned or filtered, as they propagate to the children, which eliminates the computation time by avoiding comparing each center with all the points. But as the paper points out, it works well only when the number of clusters is small.

In the community of document processing, Canopy clustering [15], which is closely related to our approach, first divides the data points into many overlapping subsets (called canopies), and clustering is performed by measuring exact distances only between points that occur within a common canopy. This eliminates a lot of unnecessary distance computations.

17.2.3 Handling Large Clusters

Hierarchical k-means. The hierarchical k-means (HKM) uses a clustering tree instead of flat k-means [17] to reduce the number of clusters in each assignment step. It first clusters the points into a small number (e.g., 10) of clusters, then recursively divides each cluster until a certain depth h is reached. The leaves in the resulted clustering tree are considered to be the final clusters (For $h = 6$, one obtains one million clusters).

Suppose that the data points associated with each node of the hierarchial tree are divided into a few (e.g., a constant number $\bar{k}$, much smaller than k) subsets (clusters). In each recursion, each point can only be assigned to one of the $\bar{k}$ clusters, and the depth of the recursions is $O(\log k)$. The computational cost is $O(n \log k)$ (ignoring the small constant number $\bar{k}$).

Approximate k-means. In [19] approximate nearest neighbor (ANN) search replaces the exact nearest neighbor (NN) search in the assignment step when searching for the nearest cluster center for each point. In particular, the current cluster centers in each k-means iteration are organized by a forest of k-d trees to perform an accelerated approximate NN search. The cost of the assignment step consists of two aspects: $O(k \log k)$ for constructing k-d trees, and $O(Mn \log k)$ for finding approximate nearest centers for each data point, where M is the number of accessed nearest cluster candidates in the k-d trees. Refined-AKM (RAKM) [18] further improves the convergence speed by enforcing constraints of non-increasing objective values during the iterations. Both AKM and RAKM require a considerable overhead of constructing k-d trees in each k-means iteration, thus a trade-off between the speed and the accuracy of the nearest neighbor search has to be made.

17.2.4 Others

There are some other complementary works in improving k-means clustering. In [22], the update step is speeded up by transforming a batch update to a mini-batch update. The high-dimensional issue has also been addressed by using dimension reduction, e.g., random projections [1, 5] and product quantization [8].

Object discovery and mining from spatially related images is one topic that is related to image clustering [2, 11, 20, 21, 23], which also aims to cluster the images so that each group contains the same object. This is a potential application of our scalable k-means algorithm. In [29, 30], we introduce an algorithm of clustering spatially-related images based on the neighborhood graph. The idea of constructing the neighborhood graph is to adopt multiple spatial partition trees, which is similar to the idea of the proposed approach in this chapter.

17.3 K-means with Cluster Closures

In this section, we first introduce the proposed approach, then give the analysis and discussions, and finally present the implementation details.

17.3.1 Approach

Active points. K-means clustering partitions the data space into Voronoi cells—each cell is a cluster and the cluster center is the center of the cell. In the assignment step, each point $\mathbf{x}$ is assigned to its nearest cluster center. We call points that change cluster assignments in an iteration *active points*. In other words, $\mathbf{x}$ changes cluster membership from the ith cluster to the jth cluster because $d(\mathbf{x}, \mathbf{c}_j) < d(\mathbf{x}, \mathbf{c}_i)$, where $d(\cdot)$ is the distance function.

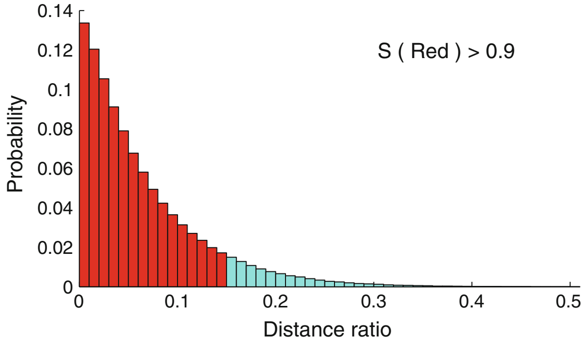


Fig. 17.1 The distribution of the distance ratio. It shows that most active points have smaller distance ratio and lie near some cluster boundaries

We observe that *active points are close to the boundary* between $\mathbf{c}_j$ and $\mathbf{c}_i$. To verify this, we define *distance ratio* for an active point $\mathbf{x}$ as: $r(\mathbf{x}) = 1 - \frac{d(\mathbf{x}, \mathbf{c}_j)}{d(\mathbf{x}, \mathbf{c}_i)}$. The distance ratio $r(\mathbf{x})$ is in the range of (0, 1], since we only compute distance ratio for active points. Smaller values of r mean closer to the cluster boundaries. Figure 17.1 shows the distribution of distance ratios when clustering 1 M GIST features from the Tiny image data set (described in 17.4.1) to 10 K clusters. We can see that most active points have small distance ratios, e.g. more than 90 % of the active points have a distance ratio less than 0.15 (shown in the red area), and thus lie near to cluster boundaries.

During the assignment step, we only need to identify the active points and change their cluster memberships. The above observation that active points lie close to cell boundaries suggests a novel approach to speed up the assignment step by identifying active points around cell boundaries.

Cluster closures. Assume for now that we have identified the neighborhood of a given point $\mathbf{x}$, a set of points containing $\mathbf{x}$'s neighboring points and itself, denoted by $\mathcal{N}_x$. We define the *closure* of a cluster $\mathcal{G}$ as:

$$\bar{\mathcal{G}} = \bigcup_{\mathbf{x} \in \mathcal{G}} \mathcal{N}_x. \tag{17.3}$$

Figure 17.2 illustrates the relationship between the cluster, the neighborhood points, and the closure.

If active points are on the cluster boundaries, as we have observed, then by increasing the neighborhood size $\mathcal{N}_x$, the group closure $\bar{\mathcal{G}}$ will be accordingly expanded to cover more active points that will be assigned to this group $\mathcal{G}$ in the assignment step. Figure 17.3 shows the recall (of an active point being covered by the closure of its newly assigned cluster) versus the neighborhood size of $\mathcal{N}_x$ over the Tinyimage data set describe in Sect. 17.4.1. Similar results are also observed in other data sets. As we can see, with a neighborhood size as small as 50, about 90 % of the active points are covered by the closures of the clusters to which these active points will be re-assigned.

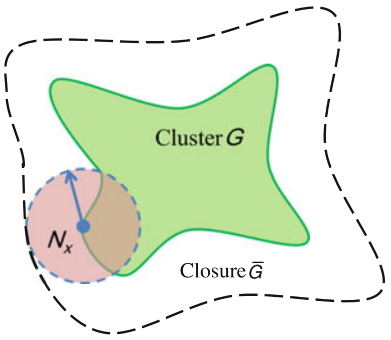


Fig. 17.2 Illustration of uniting neighborhoods to obtain the closure. The *black dash line* indicates the closure of cluster $\mathscr{G}$

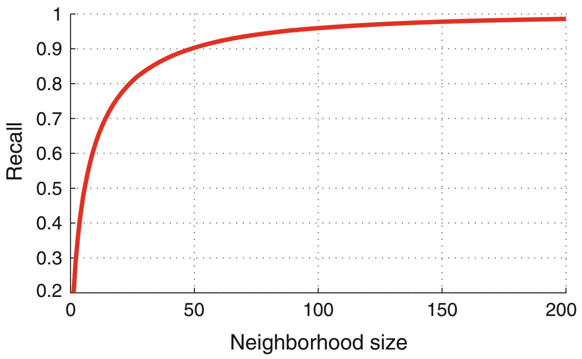


Fig. 17.3 The coverage of the active points by the closure w.r.t. the neighborhood size. A neighborhood of size 50 has about 90 % coverage

We now turn to the question of how to efficiently compute the neighborhood $\mathscr{N}_x$ of a given point $\mathbf{x}$ used in Eq. 17.3. We propose an ensemble approach using multiple random spatial partitions. A single approximate neighborhood for each point can be derived from a random partition (RP) tree [26], and the final neighborhood is assembled by combining the results from multiple random spatial partitions. Suppose that a leaf node of a single RP tree, contains a set of points $\mathscr{V} = \{\mathbf{x}_j\}$, we consider all the points in $\mathscr{V}$ to be mutually neighboring to each other. Thus the neighborhood of a point $\mathbf{x}$ in the set $\mathscr{V}$ can be straightforwardly computed by $\mathscr{N}_x = \mathscr{V}$.

Since RP trees are efficient to construct, the above neighborhood computation is also efficient. While the group closure from one single RP tree may miss some active points, using multiple RP trees effectively handles this problem. We simply unite the neighborhoods of $\mathbf{x}$ from all the RP trees:

$$\mathscr{N}_x = \bigcup_l \mathscr{V}_l.$$

Here $\mathscr{V}_l$ is a set of points in the leaf from the lth RP tree that contains $\mathbf{x}$. Note that a point $\mathbf{x}$ may belong to multiple group closures. Also note that the neighborhood of a given point is computed only once.

Fast assignment. With the group closures $\{\bar{\mathscr{G}_j}\}$ computed from Eq. 17.3, the assignment step can be done by verifying whether a point belonging to the closure $\bar{\mathscr{G}_j}$ should indeed be assigned to the cluster $\mathscr{G}_j$:

- **Initialization step**: Initialize the distance array $D[1:n]$ by assigning an positive infinity value to each entry.
- **Closure-based assignment**:
  For each cluster closure $\{\bar{\mathscr{G}_j}\}$:
  For each point $\mathbf{x}_i^s \in \bar{\mathscr{G}_j}$, $s = 1, 2, ..., |\bar{\mathscr{G}_j}|$:

$$\begin{aligned} \text{if:} \quad & \|\mathbf{x}_i^s - \mathbf{c}_j^{(t)}\|_2^2 < D[i], \\ \text{then:} \quad & z_i^{(t+1)} = j, \\ & D[i] = \|\mathbf{x}_i^s - \mathbf{c}_j^{(t)}\|_2^2. \end{aligned}$$

  Here $\mathbf{c}_j^{(t)}$ is the cluster center of $\mathscr{G}_j$ at the tth iteration, i is the global index for $\mathbf{x}$ and s is the index into $\bar{\mathscr{G}_j}$ for point $\mathbf{x}_i$.

In the assignment step, we only need to compute the distance from the center of a cluster to each point in the cluster closure. A point typically belongs to a small number of cluster closures. Thus, instead of computing the distances from a point $\mathbf{x}$ to all cluster centers in exact k-means, or constructing k-d trees of all cluster centers at each iteration to find the approximate nearest cluster center, we only need to compute the distance from $\mathbf{x}$ to a small number of cluster centers whose cluster closures contain $\mathbf{x}$, resulting in a significant reduction in computational cost. Moreover, the fact that active points are close to cluster boundaries is the worst case for k-d trees to find the nearest neighbor. On the contrary, such a fact is advantageous for our algorithm.

17.3.2 Analysis

Convergence. The following shows that our algorithm always converges. Since the objective function $J(\mathscr{C}, \mathscr{G})$ is lower-bounded, the convergence can be guaranteed if the objective value does not increase at each iterative step.

Theorem 1 (Non-increase) *The value of the objective function does not increase at each iterative step, i.e.,*

$$J(\mathscr{C}^{(t+1)}, \mathscr{G}^{(t+1)}) \leqslant J(\mathscr{C}^{(t)}, \mathscr{G}^{(t)}). \tag{17.4}$$

Proof In the assignment step for the $(t+1)$th iteration, $\{c_k^{(t)}\}$ computed from the tth iteration are cluster candidates. $\mathbf{x}_i$ would change its cluster membership only if it finds a closer cluster center, thus we have $\|\mathbf{x}_i - \mathbf{c}^{(t)}_{z_i^{(t+1)}}\|_2 \leqslant \|\mathbf{x}_i - \mathbf{c}^{(t)}_{z_i^{(t)}}\|_2$, and Eq. 17.4 holds for the assignment step.

In the update step, the cluster center will then be update based on the new point assignments. We now show that this update will not increase the within-cluster sum of squared distortions, or in a more general form:

$$\sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\bar{\mathbf{c}}_j\|_2^2 \leqslant \sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\mathbf{c}\|_2^2, \tag{17.5}$$

where $\bar{\mathbf{c}}_j$ is the jth updated cluster center $\bar{\mathbf{c}}_j = \frac{1}{|\mathscr{G}_j|}\sum_{\mathbf{x}\in\mathscr{G}_j}\mathbf{x}$, and $\mathbf{c}$ is an arbitrary point in the data space. Equation 17.5 can be verified by the following:

$$\begin{aligned}
&\sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\mathbf{c}\|_2^2 \\
&\quad= \sum_{\mathbf{x}\in\mathscr{G}_j} \|(\mathbf{x}-\bar{\mathbf{c}}_j)+(\bar{\mathbf{c}}_j-\mathbf{c})\|_2^2 \\
&\quad= \sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\bar{\mathbf{c}}_j\|_2^2 + 2(\bar{\mathbf{c}}_j-\mathbf{c})^T \sum_{\mathbf{x}\in\mathscr{G}_j}(\mathbf{x}-\bar{\mathbf{c}}_j) \\
&\qquad+ |\mathscr{G}_j|\|\bar{\mathbf{c}}_j-\mathbf{c}\|_2^2 \\
&\quad= \sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\bar{\mathbf{c}}_j\|_2^2 + |\mathscr{G}_j|\|\bar{\mathbf{c}}_j-\mathbf{c}\|_2^2 \\
&\quad\geqslant \sum_{\mathbf{x}\in\mathscr{G}_j} \|\mathbf{x}-\bar{\mathbf{c}}_j\|_2^2.
\end{aligned} \tag{17.6}$$

Thus Eq. 17.4 holds for the update step. □

Accuracy. Our algorithm obtains the same result as the exact Lloyd's algorithm if the closures of the clusters are large enough, in such a way all the points that would have been assigned to the jth cluster when using the Lloyd's algorithm belong to the cluster closure $\bar{\mathscr{G}}_j$. However, it should be noted that this condition is sufficient but not necessary. In practice, even with a small neighborhood, our approach often obtains results similar to using the exact Lloyd's algorithm. The reason is that the missing points, which should have been assigned to the current cluster at the current iteration but are missed, are close to the cluster boundary thus likely to appear in the closure of the new clusters updated by the current iteration. As a result, these missing points are very likely to be correctly[2] assigned in the next iteration.

Complexity. Consider a point $\mathbf{x}_i$ and its neighborhood $\mathscr{N}_{x_i}$, the possible groups that may absorb $\mathbf{x}_i$ are $\tilde{\mathscr{G}}_{x_i} = \{\mathscr{G}_j \mid \exists\, \mathbf{x}_j \text{ s.t. } \mathbf{x}_j \in \mathscr{G}_j \text{ and } \mathbf{x}_j \in \mathscr{N}_{x_i}\}$. As a result, we have $|\tilde{\mathscr{G}}_{x_i}| \leqslant |\mathscr{N}_{x_i}|$. In our implementation, we use balanced random bi-partition trees, with each leaf node containing c points (c is a small number). Suppose we use m random partition trees. Then the neighborhood size of a point will not be larger than $M = \text{cm}$. As a result, the complexity of the closure-based assignment step is $O(\text{nM})$.

[2] "Correctly" w.r.t. assignments if produced by Lloyd's algorithm.

For the complexity of constructing trees, our approach constructs a RP-tree in $O(n \log n)$ and AKM costs $O(k \log k)$ to build a k-d tree. However, our approach only needs a small number (typically 10 in our clustering experiments) of trees through all iterations, but AKM requires constructing a number (e.g., 8 in [19]) of trees in each iteration, which makes the total cost more expensive.

17.3.3 Discussion

We present the comparison of our approach with most relevant three algorithms, Canopy clustering, approximate k-means, and hierarchical k-means.
Versus Canopy clustering. Canopy clustering suffers from the canopy creation whose cost is high for visual features. More importantly, it is non-trivial (1) to define a meaningful and efficient approximate distance function for visual data, and (2) to tune the parameters for computing the canopy, both of which are crucial to the effectiveness and efficiency of Canopy clustering. In contrast, our approach is simpler and more efficient because random partitions can be created with a cost of only $O(n \log n)$. Moreover, our method can adaptively update cluster member candidates, in contrast to static canopies in [15].
Versus AKM. The advantages of the proposed approach over AKM are summarized as follows. First, the computational complexity of assigning a new cluster to a point in our approach is only $O(1)$, while the complexity is $O(\log k)$ for AKM or RAKM. The second advantage is that we only need to organize the data points once as the data points do not change during the iterations, in contrast to AKM or RAKM that needs to construct the k-d trees at each iteration as the cluster centers change from iteration to iteration. Last, it is shown that active points (points near cluster boundaries) present the worst case for ANN search (used in AKM) to return their accurate nearest neighbors. In contrast, our approach is able to identify active points efficiently and makes more accurate cluster assignment for active points without the shortcoming in AKM.
Versus HKM. As shown before, HKM takes less time cost than AKM and our approach. However, its cluster accuracy is not as good as HKM and our approach. This is because when assigning a point to a cluster (e.g., quantizing a feature descriptor) in HKM, it is possible that an error could be committed at a higher level of the tree, leading to a sub-optimal cluster assignment and thus sub-optimal quantization.

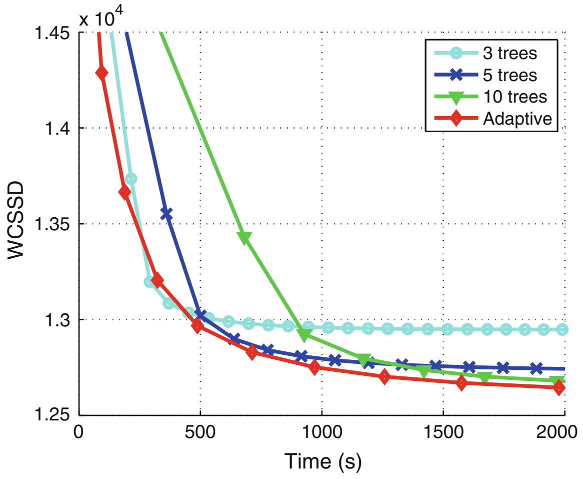
17.3.4 Implementation Details

Cluster Closure computation. The proposed algorithm adopts random partition trees for creating cluster closures. We first give a short introduction of random partition trees. The random partition tree is a binary tree structure that is formed by recursively splitting the space and aims to organize the data points in a hierarchical manner. Each node of the tree is associated with a region in the space, called a cell. These cells define a hierarchical decomposition of the space. The root node r is associated with the whole set of data points $\mathscr{X}$. Each internal node v is associated with a subset of data points $\mathscr{X}_v$ that lie in the cell of the node. It has two child nodes $\text{left}(v)$ and $\text{right}(v)$, which correspond to two disjoint subsets of data points $\mathscr{X}_{\text{left}(v)}$ and $\mathscr{X}_{\text{right}(v)}$. The leaf node l may be associated with a subset of data points or only contain a single point. Random partition trees have many applications, such as quantization and approximate nearest neighbor search. Our algorithm as described previously considers the data points lying in the same leaf node to be mutually neighboring to each other, to compute the approximate neighborhood for each data point.

In the implementation, we use a random principal direction to form the partition hyperplane to split the data points into two subsets. The principal directions are obtained by using principal component analysis (PCA). To generate random principal directions, rather than computing the principle direction from the whole subset of points, we compute the principal direction over the points randomly sampled from each subset. In our implementation, the principle direction is computed by the Lanczos algorithm [10].

We use an adaptive scheme that incrementally creates random partitions to automatically expand the group closures on demand. At the beginning of our algorithm, we only create one random partition tree. After each iteration, we compute the reduction rate of the within-cluster sum of squared distortions. If the reduction rate in successive iterations is smaller than a predefined threshold, a new random partition tree is added to expand points' neighborhood thus group closures. We compare the adaptive neighborhood scheme to a static one that computes the neighborhoods altogether at the beginning (called static neighborhoods). As shown in Fig. 17.4, we can see that the adaptive neighborhood scheme performs better in all the iterations and hence is adopted in the later comparison experiments.

Fig. 17.4 Clustering performance with adaptive versus static neighborhoods



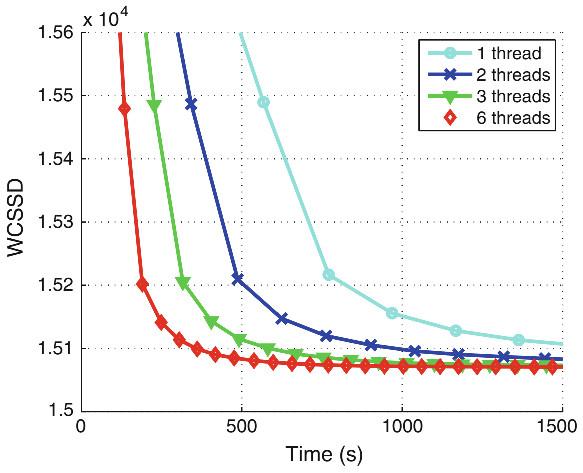


Fig. 17.5 Clustering performance with different numbers of threads

Assignment parallelization. The closure-based assignment step can be implemented in another equivalent way. For each point $\mathbf{x}$, we first identify the candidate centers by checking the cluster memberships $\mathscr{Z}_x$ of the points within the neighborhood of $\mathbf{x}$. Here $\mathscr{Z}_x = \{z(\mathbf{y}) \mid \mathbf{y} \in \mathscr{N}_x\}$, and $z(\mathbf{y})$ is the cluster membership of point $\mathbf{y}$. Then the best cluster candidate for $\mathbf{x}$ can be found by checking the clusters $\{\mathbf{c}_j \mid j \in \mathscr{Z}_x\}$. In this equivalent implementation, the assignments are computed independently and can be naturally parallelized. The update step computes the mean for each the cluster independently, which can be naturally parallelized as well. Thus, our algorithm can be easily parallelized. We show the clustering performance with the parallel implementation (using multiple threads on multi-core CPUs) in Fig. 17.5.

17.4 Experiments

17.4.1 Data Sets

SIFT. The SIFT features are collected from the Caltech 101 dataset [4]. We extract maximally stable extremal regions for each image, and compute a 128-dimensional SIFT feature for each region. We randomly sample 1 million features to form this data set.
Tiny images. We generate three datasets sampled from the tiny images [25]: 1 M tiny images, 200 K tiny images, and 500 K tiny images. The 1 M tiny images are randomly sampled without using category (tag) information. We sample 1 K (1.2 K) tags from the tiny images and sample about 200 (400) images for each tag, forming 200 K (500 K) images. We use a 384-dimensional GIST feature to represent each image.

Shopping images. We collect about 5 M shopping images from the Internet. Each image is associated with a tag to indicate its category. We sample 1 K tags and sample 200 images for each tag to form the 200 K image set. We use a 576-dimensional HOG feature to represent each image.
Oxford 5 K. This dataset [19] consists of 5,062 high resolution images of 11 Oxford landmarks. The collection has been manually annotated to generate a comprehensive ground truth for 11 different landmarks, each represented by 5 possible queries. This gives a set of 55 queries over which an object retrieval system can be evaluated. The images, the SIFT features, and the ground truth labeling of this dataset is publicly available.[3] This dataset and the next dataset will be used to demonstrate the application of our approach to object retrieval.
Ukbench 10 K. This dataset is from the Recognition Benchmark introduced in [17]. It consists of 10,200 images split into four-image groups, each of the same scene/object taken at different viewpoints. The dataset, the SIFT descriptors, and the ground truth is publicly available.[4]

17.4.2 Evaluation Metric

We use two metrics to evaluate the performance of various clustering algorithms, the within-cluster sum of squared distortions (WCSSD) which is the objective value defined by Eq. 17.1, and the normalized mutual information (NMI) which is widely used for clustering evaluation. NMI requires the ground truth of cluster assignments $\mathscr{G}$ for points in the dataset. Given a clustering result $\mathscr{X}$, NMI is defined by $\text{NMI}(\mathscr{G}, \mathscr{X}) = \frac{I(\mathscr{G},\mathscr{X})}{\sqrt{H(\mathscr{G})H(\mathscr{X})}}$, where $I(\mathscr{G}, \mathscr{X})$ is the mutual information of $\mathscr{G}$ and $\mathscr{X}$ and $H(\cdot)$ is the entropy.

In object retrieval, image feature descriptors are quantized into visual words using codebooks. A codebook of high quality will result in less quantization errors and more repeatable quantization results, thus leading to a better retrieval performance. We apply various clustering algorithms to constructing visual codebooks for object retrieval. By fixing all the other components and parameters in our retrieval system except the codebook, the retrieval performance is an indicator of the quality of the codebook. For the Oxford 5 K dataset, we follow [19] to use mean average precision (mAP) to evaluate the retrieved images. For the UKBench 10 K dataset, the retrieval performance is measured by the average number of relevant images in the top 4 retrieved images, ranging from 0 to 4.

[3] http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/index.html.

[4] http://www.vis.uky.edu/~stewe/ukbench/.

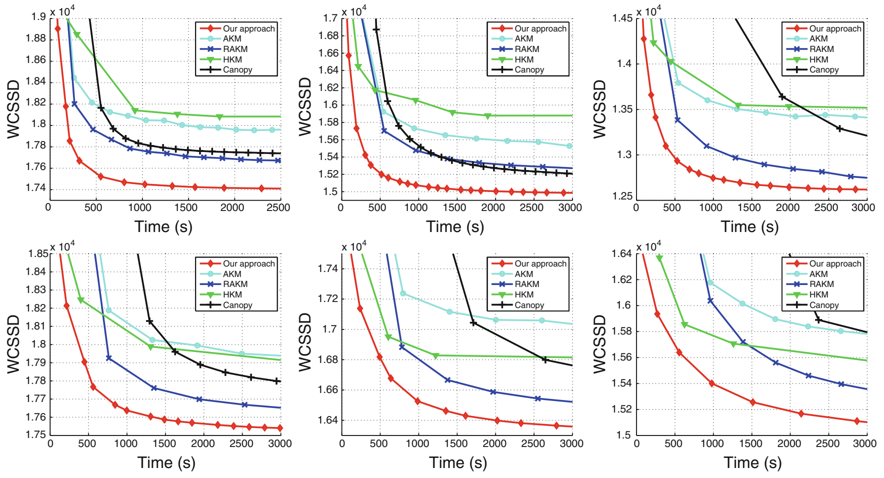


Fig. 17.6 Clustering performance in terms of within-cluster sum of squared distortions (WCSSD) versus time. The first row are the results of clustering 1 M SIFT dataset into 0.5, 2 and 10 K clusters, respectively. The second row are results on 1 M tiny image dataset

17.4.3 Clustering Performance Comparison

We compare our proposed clustering algorithm with four approximate k-means algorithms, namely hierarchial k-means (HKM), approximate k-means (AKM), refined approximate k-means (RAKM) and Canopy algorithm. The exact Lloyd's is much less efficient and prohibitively costly for large datasets, so we do not report its results. We use the implementation of HKM available from [16], and the public release of AKM.[5] The RAKM is modified from the above AKM release. For Canopy algorithm, we conduct principal component analysis over the features to project them to a lower-dimensional subspace to achieve a fast canopy construction. For a fair comparison, we initialize the cluster assignment by a random partition tree in all algorithms except HKM. The time costs for constructing trees or other initialization are all included in the comparisons. All algorithms are run on a 2.66 GHz desktop PC using a single thread.

Figure 17.6 shows the clustering performance in terms of WCSSD versus time. The experiments are performed on two datasets, the 1 M 128-dimensional SIFT datasets and the 1 M 384-dimensional tiny image dataset, respectively. The results are shown for different number of clusters, ranging from 500 to 10 K. Our approach consistently outperforms the other four approximate k-means algorithms—it converges faster to a smaller objective value.

[5] http://www.robots.ox.ac.uk/~vgg/software/fastcluster/.

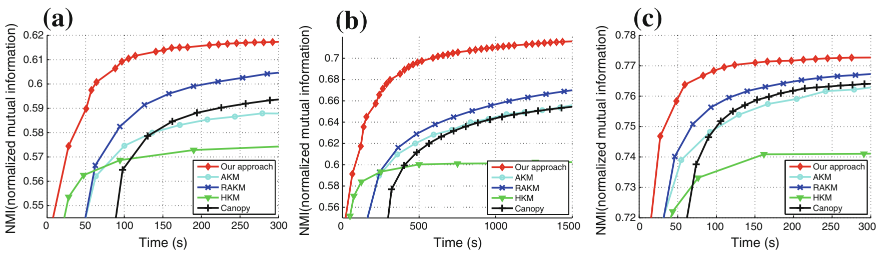


Fig. 17.7 Clustering performance in terms of normalized mutual information (NMI) versus time, on the dataset of **a** 200 K tiny images, **b** 500 K tiny images, and **c** 200 K shopping images

Figure 17.7 shows the clustering results in terms of NMI versus time. We use three labeled datasets, the 200 K tiny images, the 500 K tiny images and the 200 K shopping images. Consistent with the WCSSD comparison results, our proposed algorithm is superior to the other four clustering algorithms.

We also show the qualitative clustering results of our algorithm. Figure 17.8 shows some examples of the clustering results over the 200 K shopping images. Figure 17.9 shows some examples of the clustering results over the 500 K tiny images. The first 3 clusters are examples of similar objects, the second 3 clusters are examples of similar texture images, and the last cluster are an example of similar sceneries.

17.4.4 Empirical Analysis

We conduct empirical studies to understand why our proposed algorithm has superior performance. In particular we compare our proposed approach with AKM [19] and RAKM [18] in terms of the accuracy and the time cost of cluster assignment, using the task of clustering the 1 M tiny image dataset into 2,000 clusters. To be on the same ground, in the assignment step the number of candidate clusters for each point is set the same. For (R)AKM, the number of candidate clusters is simply the number of points accessed in k-d trees when searching for a nearest neighbor. For our proposed algorithm, we partition the data points with RP trees such that the average number of candidate clusters is the same as the number of accessed points in k-d trees. Figure 17.10a compares the accuracy of cluster assignment by varying the number of candidate clusters. We can see that our approach has a much higher accuracy in all cases, which has a positive impact on the iterative clustering algorithm to make it converge faster. Figure 17.10b compares the time of performing one iteration, by varying the number of candidate clusters M for each point. We can see that our algorithm is much faster than (R)AKM in all cases, e.g., taking only about half the time of (R)AKM when $M = 50$. This is as expected since finding the best cluster costs $O(1)$ for our algorithm but $O(\log k)$ for k-d trees used in (R)AKM.

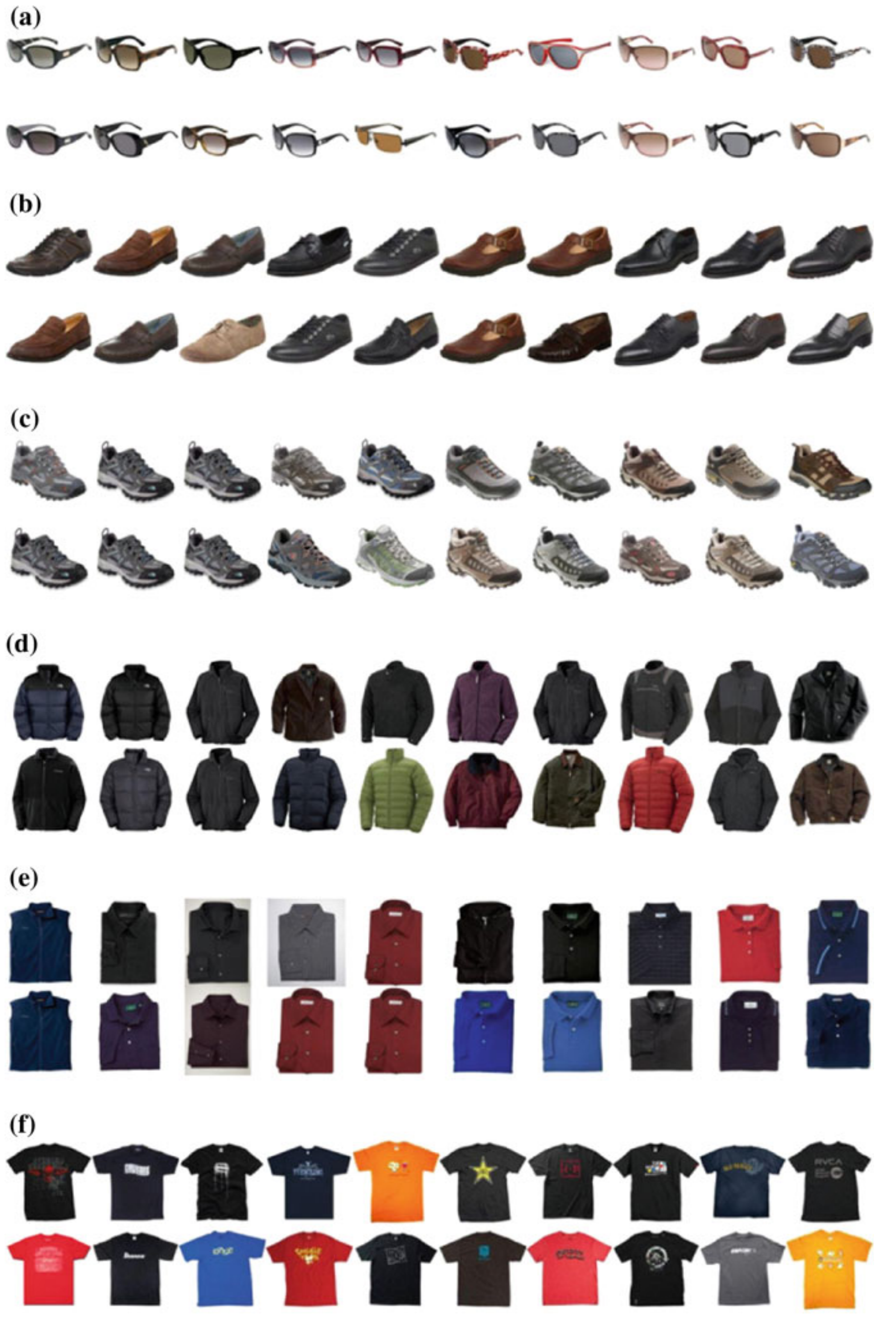


Fig. 17.8 Six representative clustering results shown from (**a**) to (**f**) over the 200 K shopping images: each cluster example is represented by two rows of images which are randomly picked out from the cluster

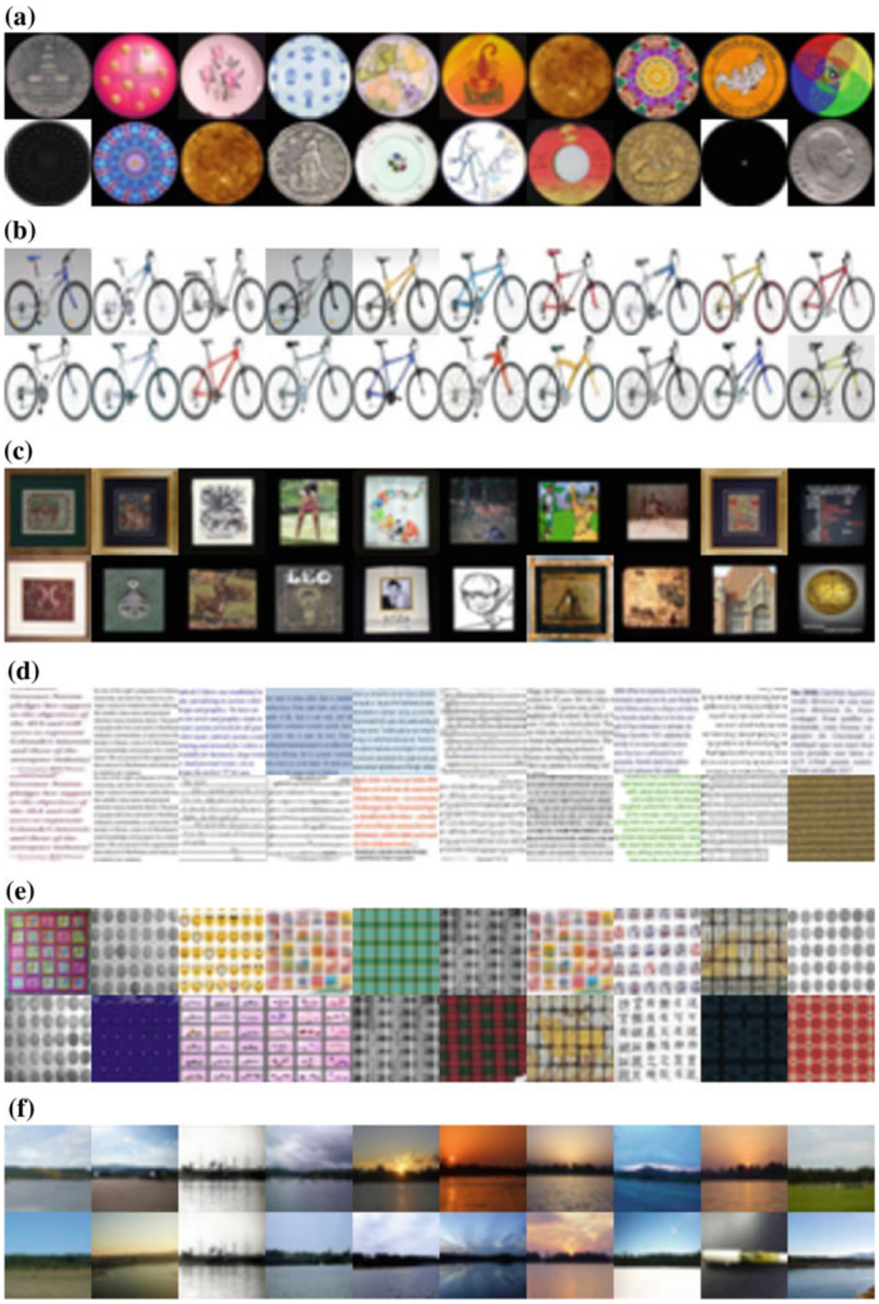


Fig. 17.9 Six representative clustering results shown from (**a**) to (**f**) over the 500 K tiny images: each cluster example is represented by two rows of images which are randomly picked out from the cluster

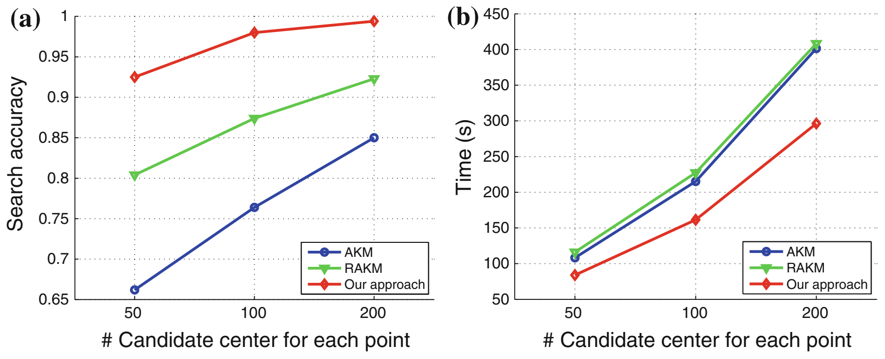


Fig. 17.10 Comparison of accuracy and time in the assignment step when clustering the 1 M tiny image data set into 2,000 clusters. **a** Accuracy versus the number of cluster candidates; **b** Time for one iteration versus the number of cluster candidates

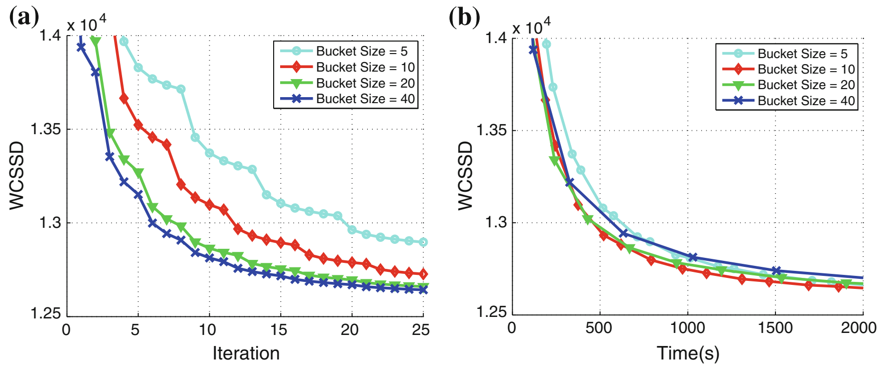


Fig. 17.11 Clustering performance versus the bucket size of a RP tree

We perform another empirical study to investigate the bucket size parameter in the RP tree, using the task of clustering the SIFT dataset into 10 K clusters. Figure 17.11a shows the results in terms of WCSSD versus the number of iterations, with bucket sizes set to 5, 10, 20, 40, respectively. A larger bucket size leads to a larger WCSSD reduction in each iteration, because it effectively increases the neighborhood size for each data point. Figure 17.11b shows the result in terms of WCSSD versus time. We observe that at the beginning, bucket sizes of 10 and 20 perform even better than the bucket size of 40. But eventually, the performance of various bucket sizes are similar. The difference between Figs. 17.11a and 17.11b is expected, as a larger bucket size leads to a better cluster assignment at each iteration, but increases the time cost for one iteration. In our comparison experiments, a bucket size of 10 is adopted.

17.4.5 Evaluation Using Object Retrieval

We compare the quality of codebooks built by HKM, (R)AKM, and our approach, using the performance of object retrieval. AKM and RAKM perform almost the same when the number of accessed candidate centers is large enough, so we only present results from AKM.

We perform the experiments on the UKBench 10 K dataset which has 7 M local features, and on the Oxford 5 K dataset which has 16 M local features. Following [19], we perform the clustering algorithms to build the codebooks, and test only the filtering stage of the retrieval system, i.e., retrieval is performed using the inverted file (including the tf-idf weighting).

The results over the UKbench 10 K dataset are obtained by constructing 1 M codebook, and use the L_1 distance metric. The results of HKM and AKM are taken from [17] and [19], respectively. From Table 17.1, we see that for the same codebook size, our method outperforms other approaches. Besides, we also conduct the experiment over subsets of various sizes, which means that we only consider the images in the subset as queries and the search range is also constrained within the subset. The performance comparison is given in Fig. 17.12, from which we can see our approach consistently gets superior performances.

The performance comparison using the Oxford 5 K dataset is shown in Table 17.2. We show the results of using the bag-of-words (BoW) representation with a 1 M codebook and using spatial re-ranking [19]. Our approach achieves the best performance, outperforming AKM in both the BoW representation and spatial re-ranking.

We also compare the performance of our approach to AKM and HKM using different codebook sizes, as shown in Table 17.3. Our approach is superior compared to other approaches with different codebook sizes. Different from AKM that gets the best performance with a 1 M-word codebook, our approach obtains the best performance with a 750 K-word codebook, indicating that our approach is producing a higher quality codebook.

Last, we show some visual examples of the retrieval results in Fig. 17.13. The first images in each row is the query, followed by the top results.

Table 17.1 A comparison of our approach to HKM and AKM on the UKbench 10 K data set using a 1 M-word codebook

Method	Scoring levels	Average top
HKM	1	3.16
HKM	2	3.07
HKM	3	3.29
HKM	4	3.29
AKM		3.45
Ours		**3.50**

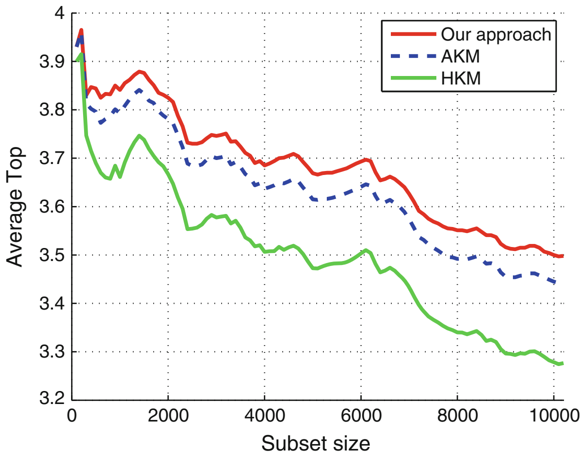


Fig. 17.12 A comparison of our approach to HKM and AKM on the UKbench 10 K data set with various subset sizes

Table 17.2 A comparison of our approach with HKM and AKM on the Oxford 5 K data set with a 1 M-word codebook

Method	Scoring level	mAP (BoW)	mAP (spatial)
HKM-1	1	0.439	0.469
HKM-2	2	0.418	
HKM-3	3	0.372	
HKM-4	4	0.353	
AKM		0.618	0.647
Our approach		**0.655**	**0.666**

Table 17.3 Performance comparison of our approach, HKM, and AKM using different codebook sizes on the Oxford 5 K data set

Vocabulary size	HKM	AKM	AKM spatial	Ours	Ours spatial
250 K	0.399	0.598	0.633	0.620	0.636
500 K	0.422	0.606	0.642	0.647	0.658
750 K	0.440	0.609	0.630	**0.664**	**0.674**
1 M	0.439	0.618	0.645	0.655	0.666
1.25 M	0.449	0.602	0.625	0.650	0.674
2 M	0.457	0.604	0.617	0.621	0.647

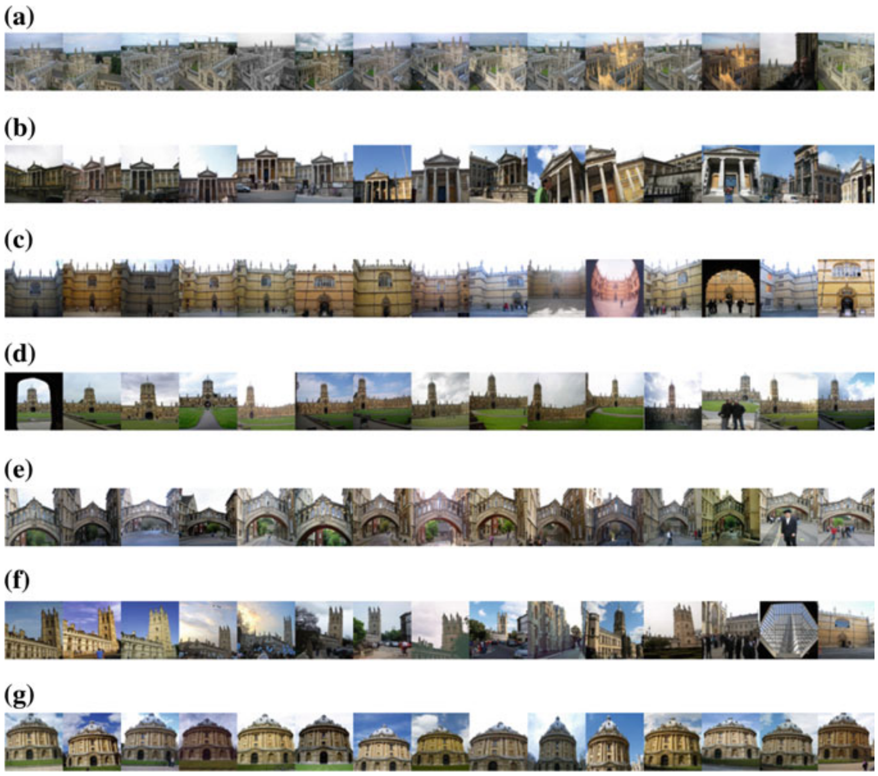


Fig. 17.13 Seven representative examples of the retrieval results shown from (**a**) to (**g**) over Oxford 5k dataset: the first image in each row is the query image and the following images are the top results

17.5 Conclusions

There are three factors that contribute to the superior performance of our proposed approach: (1) We only need to consider active points that change their cluster assignments in the assignment step of the k-means algorithm; (2) Most active points locate at or near cluster boundaries; (3) We can efficiently identify active points by pre-assembling data points using multiple random partition trees. The result is a simple, easily parallelizable, and surprisingly efficient k-means clustering algorithm. It outperforms state-of-the-art on clustering large-scale real datasets and learning codebooks for image retrieval.

Acknowledgments This work was partially supported by the National Basic Research Program of China (973 Program) under Grant 2014CB347600.

References

1. Boutsidis C, Zouzias A, Drineas P (2010) Random projections for k-means clustering. CoRR. abs/1011.4632
2. Chum O, Matas J (2010) Large-scale discovery of spatially related images. IEEE PAMI 32(2):371–377
3. Elkan C (2003) Using the triangle inequality to accelerate k-means. In: ICML
4. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: CVPR workshop on generative-model based vision
5. Fern XZ, Brodley CE (2003) Random projection for high dimensional data clustering: a cluster ensemble approach. In: ICML, pp 186–193
6. Forgy EW (1965) Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics 21:768–780
7. Frahling G, Sohler C (2008) A fast k-means implementation using coresets. Int J Comput Geom Appl 18(6):605–625
8. Jégou H, Douze M, Schmid C (2011) Product quantization for nearest neighbor search. IEEE PAMI 33(1):117–128
9. Kanungo T, Mount DM, Netanyahu NS, Piatko CD, Silverman R, Angela YW (2002) An efficient k-means clustering algorithm: analysis and implementation. IEEE PAMI 24(7): 881–892
10. Lanczos C (1950) An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. J Res Natl Bur Stand 45(4):255–282
11. Li X, Wu C, Zach C, Lazebnik S, Frahm J-M (2008) Modeling and recognition of landmark image collections using iconic scene graphs. In: ECCV
12. Lloyd Stuart P (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28(2): 129–137
13. MacQueen JB (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley symposium on mathematical statistics and probability, vol 1, pp 281–297
14. Mahajan M, Nimbhorkar P, Varadarajan KR (2009) The planar k-means problem is NP-hard. In: WALCOM, pp 274–285
15. McCallum A, Nigam K, Ungar LH (2000) Efficient clustering of high-dimensional data sets with application to reference matching. In: KDD
16. Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. VISSAPP 1:331–340
17. Nistér D, Stewénius H (2006) Scalable recognition with a vocabulary tree. In: CVPR
18. Philbin J (2010) Scalable object retrieval in very large image collections. PhD thesis, University of Oxford
19. Philbin J, Chum O, Isard M, Sivic J, Zisserman A (2007) Object retrieval with large vocabularies and fast spatial matching. In: CVPR
20. Philbin J, Zisserman A (2008) Object mining using a matching graph on very large image collections. In: ICVGIP, pp 738–745
21. Raguram R, Wu C, Frahm J-M, Lazebnik S (2011) Modeling and recognition of landmark image collections using iconic scene graphs. IJCV 95(3):213–239
22. Sculley D (2010) Web-scale k-means clustering. In: WWW
23. Simon I, Snavely N, Seitz SM (2007) Scene summarization for online image collections. In: ICCV, pp 1–8
24. Sivic J, Zisserman A (2003) Video Google: a text retrieval approach to object matching in videos. In: ICCV, pp 1470–1477
25. Torralba AB, Fergus R, Freeman WT (2008) 80 million tiny images: a large data set for nonparametric object and scene recognition. IEEE PAMI 30(11):1958–1970
26. Verma N, Kpotufe S, Dasgupta S (2009) Which spatial partition trees are adaptive to intrinsic dimension? In: Proceedings of 25th conference on uncertainty in artificial intelligence

27. Wang J, Jia L, Hua X-S (2011) Interactive browsing via diversified visual summarization for image search results. Multimed Syst 17(5):379–391
28. Wang J, Wang J, Ke Q, Zeng G, Li S (2012) Fast approximate k-means via cluster closures. In: CVPR, pp 3037–3044
29. Wang J, Wang J, Zeng G, Tu Z, Gan R, Li S (2012) Scalable k-NN graph construction for visual descriptors. In: CVPR, pp 1106–1113
30. Wang J, Wang J, Zeng G, Tu Z, Gan R, Li S (2013) Scalable k-NN graph construction. CoRR. abs/1307.7852

Chapter 18
Fast Neighborhood Graph Search Using Cartesian Concatenation

Jingdong Wang, Jing Wang, Gang Zeng, Rui Gan, Shipeng Li and Baining Guo

Abstract In this chapter, we propose a new data structure for approximate nearest neighbor search. This structure augments the neighborhood graph with a bridge graph. We propose to exploit Cartesian concatenation to produce a large set of vectors, called bridge vectors, from several small sets of subvectors. Each bridge vector is connected with a few reference vectors near to it, forming a bridge graph. Our approach finds nearest neighbors by simultaneously traversing the neighborhood graph and the bridge graph in the best-first strategy. The success of our approach stems from two factors: the exact nearest neighbor search over a large number of bridge vectors can be done quickly, and the reference vectors connected to a bridge (reference) vector near the query are also likely to be near the query. Experimental results on searching over large scale datasets (SIFT, GIST and HOG) show that our approach outperforms state-of-the-art ANN search algorithms in terms of efficiency and accuracy. The combination of our approach with the IVFADC system [1] also shows superior performance over the BIGANN dataset of 1 billion SIFT features compared with the best previously published result.

J. Wang (✉) · S. Li · B. Guo
Microsoft, Building 2, No. 5 Dan Ling Street, Beijing 100080, Haidian District, China
e-mail: jingdw@microsoft.com

S. Li
e-mail: spli@microsoft.com

B. Guo
e-mail: bainguo@microsoft.com

J. Wang
Key Laboratory on Machine Perception, Peking University, Science Building No. 2, Beijing 1000871, China
e-mail: cis.wangjing@pku.edu.cn

G. Zeng
Key Laboratory on Machine Perception, Peking University, Room 2202, Science Building No. 2, Beijing 1000871, China
e-mail: g.zeng@ieee.org

R. Gan
School of Mathematical Sciences, Peking University, Beijing 1000871, China
e-mail: rui_gan@ieee.org

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_18

18.1 Introduction

Similar image search, a.k.a., content-based image retrieval, has been attracted a lot of attention in multimedia research. The study before the year of 2005 mainly focuses on designing the low-level features describing the images. Recently, a lot of research focus has been switched to the nearest neighbor search algorithm as the size of the database has increased to millions and even billions because of the popularity of large scale and high-dimensional multimedia data in Internet and social media websites such as Flickr. In this chapter, we study the nearest neighbor (NN) search problem.

The simplest solution to NN search is linear scan, comparing each reference vector to the query vector. The search complexity is linear with respect to both the number of reference vectors and the data dimensionality. Apparently, it is too time-consuming and does not scale well in large scale and high-dimensional problems. Algorithms, including the KD tree [2–5], BD trees [2], cover tree [6], nonlinear embedding [7] and so on, have been proposed to improve the search efficiency. However, for high-dimensional cases, it turns out that such approaches are not much more efficient than linear scan and cannot satisfy the practical requirement. Therefore, a lot of efforts have been turned to approximate nearest neighbor (ANN) search, such as KD trees with its variants, hashing algorithms, neighborhood graph search, and inverted indices.

In this chapter, we propose a new data structure for approximate nearest neighbor search.[1] This structure augments the neighborhood graph with a bridge graph that is able to boost approximate nearest neighbor search performance. Inspired by the product quantization technology [1, 9], we adopt Cartesian concatenation (or Cartesian product), to generate a large set of vectors, which we call *bridge vectors*, from several small sets of subvectors to approximate the reference vectors. Each bridge vector is then connected to a few reference vectors that are near enough to it, forming a bridge graph. Combining the bridge graph with the neighborhood graph built over reference data vectors yields an augmented neighborhood graph. The ANN search procedure starts by finding the nearest bridge vector to the query vector, and discovers the first set of reference vectors connected to such a bridge vector. Then the search simultaneously traverses the bridge graph and the neighborhood graph in the best-first manner using a shared priority queue.

The advantages of adopting the bridge graph lie in two-fold. First, computing the distances from bridge vectors to the query is very efficient, for instance, the computation for $1{,}000{,}000$ bridge vectors that are formed by 3 sets of 100 subvectors takes almost the same time as that for 100 vectors. Second, the best bridge vector is most likely to be very close to true NNs, allowing the ANN search to quickly reach true NNs through bridge vectors.

We evaluate the proposed approach by the feature matching performance on SIFT and HOG features, and the performance of searching similar images over tiny images [10] with GIST features. We show that our approach achieves significant improvements compared with the state-of-the-art in terms of accuracy and search

[1] A conference version appeared in [8].

time. We also demonstrate that our approach in combination with the IVFADC system [1] outperforms the state-of-the-art over the BIGANN dataset of 1 billion SIFT vectors [11].

18.2 Literature Review

Nearest neighbor search in the d-dimensional metric space $\mathbb{R}^d$ is defined as follows: given a query $\mathbf{q}$, the goal is to find an element $\mathrm{NN}(\mathbf{q})$ from the database $\mathscr{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ so that $\mathrm{NN}(\mathbf{q}) = \arg\min_{\mathbf{x}\in\mathscr{X}} \mathrm{dist}(\mathbf{q}, \mathbf{x})$. In this chapter, we assume that $\mathbb{R}^d$ is an Euclidean space and $\mathrm{dist}(\mathbf{q}, \mathbf{x}) = \|\mathbf{q} - \mathbf{x}\|_2$, which is appropriate for most problems in multimedia search and computer vision.

There are two types of ANN search problems. One is error-constrained ANN search that terminates the search when the minimum distance found up to now lies in some scope around the true minimum (or desired) distance. The other one is time-constrained ANN search that terminates the search when the search reaches some prefixed time (or equivalently examines a fixed number of data points). The latter category is shown to be more practical and give better performance. Our proposed approach belongs to the latter category.

The ANN search algorithms can be roughly divided into four categories: partition trees, neighborhood graph, compact codes (hashing and source coding), and inverted index. The following presents a short review of the four categories.

18.2.1 Partition Trees

The partition tree based approaches recursively split the space into subspaces, and organize the subspaces via a tree structure. Most approaches select hyperplanes or hyperspheres according to the distribution of data points to divide the space, and accordingly data points are partitioned into subsets.

The KD trees [4, 5], using axis-aligned hyperplane to partition the space, have been modified to find ANNs. Other trees using different partition schemes, such as BD tress [2], metric trees [12–15], hierarchical k-means tree [16], and randomized KD trees [17–19], have been proposed. FLANN [20] aims to find the best configuration of the hierarchical k-means trees and randomized KD trees, and has been shown to work well in practice.

In the query stage, the branch-and-bound methodology [4] is usually adopted to search (approximate) nearest neighbors. This scheme needs to traverse the tree in the depth-first manner from the root to a leaf by evaluating the query at each internal node, and pruning some subtrees according to the evaluation and the currently-found nearest neighbors. The current state-of-the-art search strategy, priority search [2] or best-first search [3], maintains a priority queue to access subtrees in order so that the data points with large probabilities being true nearest neighbors are first accessed. It has been shown that best-first search (priority search) achieves the best performance for ANN search, while the performance might be worse for exact NN search than the algorithms without using best-first search.

18.2.2 Neighborhood Graph Search

The data structure of the neighborhood graph is a directed graph connecting each vector and its nearest neighbors. Usually a R-NN graph, that connects each vector to its R nearest neighbors, is used. Various algorithms based on neighborhood graph [3, 21–26] are developed for ANN search.

The basic procedure of neighborhood graph search starts from one or several seeding vectors, and puts them into a priority queue with the distance to the query being the key. Then the process proceeds by popping the top one in the queue, i.e., the nearest one to the query, and expanding its neighborhood vectors (from neighborhood graph), among which the vectors that have not been visited are pushed into the priority queue. This process iterates till a fixed number of vectors are accessed.

Using neighborhood vectors of a vector as candidates has two advantages. One is that extracting the candidates is very cheap and only takes $O(1)$ time. The other is that if one vector is close to the query, its neighborhood vectors are also likely to be close to the query. The main research effort consists of two aspects. One is to build an effective neighborhood graph [21, 24]. The other is to design efficient and effective ways to guide the search in the neighborhood graph, including presetting the seeds created via clustering [24, 25], picking the candidates from KD trees [22], iteratively searching between KD trees and the neighborhood graph [26]. In this chapter, we present a more effective way, combining the neighborhood graph with a bridge graph, to search for approximate nearest neighbors.

18.2.3 Compact Codes

The compact code approaches transform each data vector into a small code, using the hashing or source coding techniques. Usually the small code takes much less storage than the original vector, and particularly the distance in the small code space, e.g., hamming distance or using lookup table can be much more efficiently evaluated than in the original space.

Locality sensitive hashing (LSH) [27], originally used in a manner similar to inverted index, has been shown to achieve good theory guarantee in finding near neighbors with probability, but it is reported not as good as KD trees in practice [20]. Multi-probe LSH [28] adopts the search algorithm similar to priority search, achieving a significant improvement. Nowadays, the popular usage of hashing is to use the hamming distance between hash codes to approximate the distance in the original space and then adopt linear scan to conduct the search. To make the best of the data, recently, various data-dependent hashing algorithms are proposed by learning hash functions using metric learning-like techniques, including optimized kernel hashing [29], learned metrics [30], learnt binary reconstruction [31], kernelized LSH [32], and shift kernel hashing [33], semi-supervised hashing [34], (multidimensional) spectral hashing [35, 36], iterative quantization [37], complementary hashing [38] and order preserving hashing [39].

The source coding approach, product quantization [1], divides the vector into several (e.g., M) bands, and quantizes reference vectors for each band separately. Then each reference vector is approximated by the nearest center in each band, and the index for the center is used to represent the reference vector. Accordingly, the distance in the original space is approximated by the distance over the assigned centers in all bands, which can be quickly computed using precomputed lookup tables storing the distances between the quantization centers of each band separately.

18.2.4 Inverted Index

Inverted index is composed of a set of inverted lists each of which contains a subset of the reference vectors. The query stage selects a small number of inverted lists, regards the vectors contained in the selected inverted lists as the NN candidates, and reranks the candidates, using the distance computed from the original vector or using the distance computed from the small codes followed by a second-reranking step using the distance computed from the original vector, to find the best candidates.

The inverted index algorithms are widely used for very large datasets of vectors (hundreds of million to billions) due to its small memory cost. Such algorithms usually load the inverted index (and possibly extra codes) into the memory and store the raw features in the disk. A typical inverted index is built by clustering algorithms, e.g., [1, 9, 16, 40, 41], and is composed of a set of inverted lists, each of which corresponds to a cluster of reference vectors. Other inverted indices include hash tables [27], tree codebooks [4] and complementary tree codebooks [42].

18.3 Preliminaries

This section gives short introductions on several algorithms our approach depends on: neighborhood graph search, product quantization, and the multi-sequence search algorithm.

18.3.1 Neighborhood Graph Search

A neighborhood graph of a set of vectors $\mathscr{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is a directed graph that organizes data vectors by connecting each data point with its neighboring vectors. The neighborhood graph is denoted as $G = \{(v_i, Adj[v_i])\}_{i=1}^n$, where v_i corresponds to a vector $\mathbf{x}_i$ and $Adj[v_i]$ is a list of nodes that correspond to its neighbors.

The ANN search algorithm proposed in [22], we call local neighborhood graph search, is a procedure that starts from a set of seeding points as initial NN candidates and propagates the search by continuously accessing their neighbors from previously-discovered NN candidates to discover more NN candidates. The *best-first* strategy [22] is usually adopted for local neighborhood expansion.[2] To this end, a *priority queue* is used to maintain the previously-discovered NN candidates whose neighborhoods are not expanded yet, and initially contains only seeds. The best candidate in the priority queue is extracted out, and the points in its neighborhood are discovered as new NN candidates and then pushed into the priority queue. The resulting search path of discovering NN candidates may not be monotone, but always attempts to move closer to the query point without repeating points. As a local search that finds better solutions only from the neighborhood of the current solution, the local neighborhood graph search will be stuck at a locally optimal point and has to conduct exhaustive neighborhood expansions to find better solutions. Both the proposed approach and the iterated approach [26] aim to efficiently find solutions beyond local optima.

18.3.2 Product Quantization

The idea of product quantization is to decompose the space into a Cartesian product of M low-dimensional subspaces and to quantize each subspace separately (e.g., using the k-means algorithm). A vector $\mathbf{x}$ is then decomposed into M subvectors, $\mathbf{x}^1, \ldots, \mathbf{x}^M$, such that $\mathbf{x}^T = [(\mathbf{x}^1)^T \ (\mathbf{x}^2)^T \ldots (\mathbf{x}^M)^T]$. Let the quantization dictionaries over the M subspaces be $\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_M$ with $\mathscr{C}_m$ being a set of centers $\{\mathbf{c}_{m1}, \ldots, \mathbf{c}_{mK}\}$. A vector $\mathbf{x}$ is represented by a short code composed of its subspace quantization indices, $\{k_1, k_2, \ldots, k_M\}$. Equivalently,

$$\mathbf{x} = \begin{bmatrix} \mathbf{C}^{(1)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{(2)} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C}^{(M)} \end{bmatrix} \begin{bmatrix} \mathbf{b}^{(1)} \\ \mathbf{b}^{(2)} \\ \vdots \\ \mathbf{b}^{(M)} \end{bmatrix}, \tag{18.1}$$

[2] The depth-first search strategy can also be used. Our experiments show that the performance is much worse than the best-first search.

where $\mathbf{C}^{(m)} = [\mathbf{c}_{m1}\ \mathbf{c}_{m2} \ldots \mathbf{c}_{mK}]$, and $\mathbf{b}^{(m)}$ is a vector in which the k_mth entry is 1 and all others are 0.

Given a query $\mathbf{q}$, the asymmetric scheme divides $\mathbf{q}$ into M subvectors $\{\mathbf{q}^1, \ldots, \mathbf{q}^M\}$, and computes M distance arrays $\{\mathbf{d}_1, \ldots, \mathbf{d}_M\}$ (for computation efficiency, store the square of the Euclidean distance) with the centers of the M subspaces. For a database point encoded as $\{k_1, k_2, \ldots, k_M\}$, the square of the Euclidean distance is approximated as $\sum_{m=1}^{M} d_{mk_m}$, which is called asymmetric distance.

The application of product quantization in our approach is different from applications to fast distance computation [1] and code book construction [9], the goal of Cartesian product in this chapter is to build a bridge to connect the query and the reference vectors through bridge vectors.

18.3.3 Multi-sequence Search

Given several monotonically increasing sequences, $\{S_b\}_{b=1}^{m}$ where S_b is a sequence of real values, $s_b(1), s_b(2), \ldots, s_b(n_b)$, with $s_b(l) < s_b(l+1)$, the multi-sequence search algorithm [9] aims at traversing the set of m-tuples $\{(i_1, i_2, \ldots, i_m)|i_b = 1 \ldots n_b\}$ in order of increasing the sum $s_1(i_1) + s_2(i_2) + \cdots + s_m(i_m)$, or finding t m-tuples $\{(i_1^k, i_2^k, \ldots, i_m^k)|k = 1, 2, \ldots, t\}$ whose sums are the smallest among all the possible m-tuples.

The algorithm uses a min-priority queue of the tuples $(i_1, i_2, \ldots, i_m)$ with the key being the sum $s_1(i_1) + s_2(i_2) + \cdots + s_m(i_m)$. It starts by initializing the queue with a tuple $(1, 1, \ldots, 1)$. At step t, the tuple with top priority (the minimum sum), $(i_1^{(t)}, i_2^{(t)}, \ldots, i_m^{(t)})$, is popped from the queue and regarded as the tth best tuple whose sum is the tth smallest. At the same time, the tuple $(i_1, i_2, \ldots, i_m)$, if all its preceding tuples, $\{(i'_1, i'_2, \ldots, i'_m)|i'_b = i_b, i_b - 1\} - \{(i_1, i_2, \ldots, i_m)\}$ have already been pushed into the queue, is pushed into the queue. As a result, the multi-sequence algorithm produces a sequence of m-tuples in order of increasing the sum and can stop at step $t-1$ if the best t m-tuples are required. The time cost of extracting the best t m-tuples is $t(\log t + m)$. Here m corresponds to the cost of checking if the m successors of one tuple popped out from the queue are qualified to be pushed into the queue. $\log t$ corresponds to the cost of inserting a tuple into the queue, where t is the upper bound of the length of the queue for the tth step.

18.4 Approach

The database $\mathcal{X}$ contains N d-dimensional reference vectors, $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^d$. Our goal is to build an index structure using the bridge graph such that, given a query vector $\mathbf{q}$, its nearest neighbors can be quickly discovered. In this section,we first describe the index structure and then show the search algorithm.

18.4.1 Data Structure

Our index structure consists of two components: a bridge graph that connects bridge vectors and their nearest reference vectors, and a neighborhood graph that connects each reference vector to its nearest reference vectors.

Bridge vectors Cartesian concatenation is an operation that builds a new set out of a number of given sets. Given m sets, $\{\mathscr{S}_1, \mathscr{S}_2, \ldots, \mathscr{S}_m\}$, where each set, in our case, contains a set of d_i-dimensional subvectors such that $\sum_{i=1}^{m} d_i = d$, the Cartesian concatenation of those sets is defined as follows,

$$\mathscr{Y} = \times_{i=1}^{m} \mathscr{S}_i \triangleq \{\mathbf{y}_j = [\mathbf{y}_{j_1}^T \; \mathbf{y}_{j_2}^T \; \cdots \; \mathbf{y}_{j_m}^T]^T | \mathbf{y}_{j_i} \in \mathscr{S}_i\}.$$

Here $\mathbf{y}_j$ is a d-dimensional vector, and there exist $\prod_{i=1}^{m} n_i$ vectors ($n_i = |\mathscr{S}_i|$ is the number of elements in $\mathscr{S}_i$) in the Cartesian concatenation $\mathscr{Y}$. Without loss of generality, we assume that $n_1 = n_2 = \cdots = n_m = n$ for convenience. There is a nice property that identifying the nearest one from $\mathscr{Y}$ to a query only takes $O(dn)$ time rather than $O(dn^m)$, despite that the number of elements in $\mathscr{Y}$ is n^m. Inspired by this property, we use the Cartesian concatenation $\mathscr{Y}$, called bridge vectors, as bridges to connect the query vector with the reference vectors.

Computing bridge vectors We propose to use product quantization [1], which aims to minimize the distance of each vector to the nearest concatenated center derived from subquantizers, to compute bridge vectors. This ensures that the reference vectors discovered through one bridge vector are not far away from the query and hence the probability that those reference vectors are true NNs is high.

It is also expected that the number of reference vectors that are close enough to at least one bridge vector should be as large as possible (to make sure that enough good reference vectors can be discovered merely through bridge vectors) and that the average number of the reference vectors discovered through each bridge vector should be small (to make sure that the time cost to access them is low). To this end,we generate a large amount of bridge vectors. Such a requirement is similar to [1] for source coding and different from [9] for inverted indices.

Augmented neighborhood graph The augmented neighborhood graph is a combination of the neighborhood graph $\bar{G}$ over the reference database $\mathscr{X}$ and the bridge graph B between the bridge vectors $\mathscr{Y}$ and the reference vectors $\mathscr{X}$. The neighborhood graph $\bar{G}$ is a directed graph. Each node corresponds to a point $\mathbf{x}_i$, and is also denoted as $\mathbf{x}_i$ for convenience. Each node $\mathbf{x}_i$ is connected with a list of nodes that correspond to its neighbors, denoted by $Adj[\mathbf{x}_i]$.

The bridge graph B is constructed by connecting each bridge vector $\mathbf{y}_j$ in $\mathscr{Y}$ to its nearest vectors $Adj[\mathbf{y}_i]$ in $\mathscr{X}$. To avoid expensive computation cost, we build the bridge graph approximately by finding top t (typically 100 in our experiments) nearest bridge vectors for each reference vector and then keeping top b nearest (typically 5 in our experiments) reference vectors for each bridge vector.

The bridge graph is different from the inverted multi-index [9]. In the inverted multi-index, each bridge vector $\mathbf{y}$ contains a list of vectors that are closer to $\mathbf{y}$ than all other bridge vectors, while in our approach each bridge is associated with a list of vectors that are closer to $\mathbf{y}$ than all other reference data points.

18.4.2 Query the Augmented Neighborhood Graph

To make the description clear, without loss of generality, we assume there are two sets of n subvectors, $\mathcal{S}_1 = \{\mathbf{y}_1^1, \mathbf{y}_2^1, \ldots, \mathbf{y}_n^1\}$ and $\mathcal{S}_2 = \{\mathbf{y}_1^2, \mathbf{y}_2^2, \ldots, \mathbf{y}_n^2\}$. Given a query $\mathbf{q}$ consisting of two subvectors $\mathbf{q}^1$ and $\mathbf{q}^2$, the goal is to generate a list of T ($T \ll N$) candidate reference points from $\mathcal{X}$ where the true NNs of $\mathbf{q}$ are most likely to lie. This is achieved by traversing the augmented neighborhood graph in a best-first strategy.

We give a brief overview of the ANN search procedure over a neighborhood graph before describing how to make use of bridge vectors. The algorithm begins with a set of (one or several) vectors $\mathcal{P}_s = \{\mathbf{p}\}$ that are contained in the neighborhood graph. It maintains a set of nearest neighbor candidates (whose neighborhoods have not been expanded), using a min-priority queue, which we call the main queue, with the distance to the query as the key. The main queue initially contains the vectors in $\mathcal{P}_s$. The algorithm proceeds by iteratively expanding the neighborhoods in a best-first strategy. At each step, the vector $\mathbf{p}^*$ with top priority (the nearest one to $\mathbf{q}$) is popped from the queue. Then each neighborhood vector in $Adj[\mathbf{p}^*]$ is inserted to the queue if it is not visited, and at the same time it is added to the result set (maintained by a max-priority queue with a fixed length depending on how many nearest neighbors are expected).

To exploit the bridge vectors, we present an extraction-on-demand strategy, instead of fetching all the bridge vectors to the main queue, which leads to expensive cost in sorting them and maintaining the main queue. Our strategy is to maintain the main queue such that it consists of only one bridge vector if available. To be specific, if the top vector $\mathbf{p}^*$ in the main queue is a reference vector, the algorithm proceeds as usual, the same to the above procedure without using bridge vectors. If the top vector is a bridge vector, we first insert its neighbors $Adj[\mathbf{p}^*]$ into the main queue and the result set, and in addition we find the next nearest bridge vector (to the query $\mathbf{q}$) and insert it to the main queue. The pseudo code of the search algorithm is given in Algorithm 1 and an example process is illustrated in Fig. 18.1.

Before traversing the augmented neighborhood graph, we first process the bridge vectors, and compute the distances (the square of the Euclidean distance) from $\mathbf{q}^1$ to the subvectors in $\mathcal{S}_1$ and from $\mathbf{q}^2$ to the subvectors in $\mathcal{S}_2$, and then sort the subvectors in the order of increasing distances, respectively. We denote the sorted subvectors as $\{\mathbf{y}_{i_1}^1, \ldots, \mathbf{y}_{i_n}^1\}$ and $\{\mathbf{y}_{j_1}^2, \ldots, \mathbf{y}_{j_n}^2\}$. As the size n of $\mathcal{S}_1$ and $\mathcal{S}_2$ is typically not large (e.g., 100 in our case), the computation cost is very small (See details in Sect. 18.6).

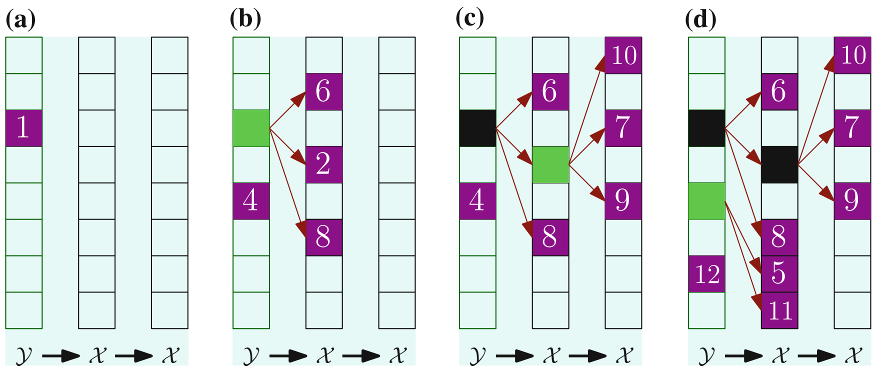

(a)
1
(b)
6
2
4
8
(c)
10
6
7
4
9
8
(d)
10
6
7
9
8
12
5
11
$\mathcal{Y} \rightarrow \mathcal{X} \rightarrow \mathcal{X}$

Fig. 18.1 An example illustrating the search process. $\mathscr{Y} \rightarrow \mathscr{X}$: the bridge graph, and $\mathscr{X} \rightarrow \mathscr{X}$: the neighborhood graph. The *white numbers* are the distances to the query. *Magenta* denotes the vectors in the main queue, *green* represents the vector being popped out from the main queue, and *black* indicates the vectors whose neighborhoods have already been expanded **a** Iteration 1, **b** Iteration 2, **c** Iteration 3, **d** Iteration 4

The extraction-on-demand strategy needs to visit the bridge vector one by one in the order of increasing distance from $\mathbf{q}$. It is easily shown that $\text{dist}^2(\mathbf{q}, \mathbf{y}) = \text{dist}^2(\mathbf{q}^1, \mathbf{y}^1) + \text{dist}^2(\mathbf{q}^2, \mathbf{y}^2)$, where $\mathbf{y}$ is consists of $\mathbf{y}^1$ and $\mathbf{y}^2$. Naturally, $\mathbf{y}_{i_1, j_1}$, composed of $\mathbf{y}^1_{i_1}$ and $\mathbf{y}^2_{i_1}$, is the nearest one to $\mathbf{q}$. The multi-sequence algorithm (corresponding to ExtractNextNearestBridgeVector() in Algorithm 1) is able to fast produce a sequence of pairs (i_k, j_l) so that the corresponding bridge vectors are visited in the order of increasing distances to the query $\mathbf{q}$. The algorithm is very efficient and producing the tth bridge vector only takes $O(\log(t))$ time. Slightly different from extracting a fixed number of nearest bridge vectors once [9], our algorithm automatically determines when to extract the next one, that is when there is no bridge vector in the main queue.

18.5 Experiments

18.5.1 Setup

We perform our experiments on three large datasets: the first one with local SIFT features, the second one with global GIST features, and the third one with HOG features, and a very large dataset, the BIGANN dataset of 1 billion SIFT features [11]. These three kinds of features are useful in real applications: Particular object retrieval includes a step of mapping a SIFT feature to a visual word, which is a problem of searching the nearest one from a vocabulary of SIFT features; Similar image search can use GIST or HOG to represent the images and then is transformed to a nearest neighbor search problem over the GIST or HOG vector space.

Algorithm 1 ANN search over the augmented neighborhood graph

```
    /* $\mathbf{q}$: the query; $\mathscr{X}$: the reference data vectors; $\mathscr{Y}$: the set of bridge vectors; $G$: the augmented neighborhood graph; $Q$: the main queue; $R$: the result set; $T$: the maximum number of discovered vectors; */
Procedure ANNSearch($\mathbf{q}$, $\mathscr{X}$, $\mathscr{Y}$, $G$, $Q$, $R$, $T$)
 1. /* Mark each reference vector undiscovered */
 2. for each $\mathbf{x} \in \mathscr{X}$ do
 3.    Color[$\mathbf{x}$] $\leftarrow$ white;
 4. end for
 5. /* Extract the nearest bridge vector */
 6. ($\mathbf{y}$, $D$) $\leftarrow$ ExtractNextNearestBridgeVector($\mathscr{Y}$);
 7. $Q \leftarrow (\mathbf{y}, D)$;
 8. $t \leftarrow 0$
 9. /* Start the search */
10. while ($Q \neq \emptyset$ &&$t \leqslant T$) do
11.    /* Pop out the best candidate vector and expand its neighbors */
12.    ($\mathbf{p}$, $D$) $\leftarrow$ $Q$.pop();
13.    for each $\mathbf{x} \in Adj[\mathbf{p}]$ do
14.       if Color[$\mathbf{x}$] = white then
15.          $D \leftarrow \text{dist}(\mathbf{q}, \mathbf{x})$;
16.          $Q \leftarrow (\mathbf{x}, D)$;
17.          Color[$\mathbf{x}$] $\leftarrow$ black; /* Mark it discovered */
18.          $R \leftarrow (\mathbf{x}, D)$; /* Update the result set */
19.          $t \leftarrow t + 1$;
20.       end if
21.    end for
22.    /* Extract the next nearest bridge vector if $\mathbf{p}$ is a bridge vector */
23.    if $\mathbf{p} \in \mathscr{Y}$ then
24.       ($\mathbf{y}$, $D$) $\leftarrow$ ExtractNextNearestBridgeVector($\mathscr{Y}$);
25.       $Q \leftarrow (\mathbf{y}, D)$;
26.    end if
27. end while
28. return $R$;
```

The SIFT features are collected from the Caltech 101 dataset [43]. We extract maximally stable extremal regions (MSERs) for each image, and compute a 128-dimensional byte-valued SIFT feature for each MSER. We randomly sample 1,000 K SIFT features and 100 K SIFT features, respectively as the reference and query set. The GIST features are extracted on the tiny image set [10]. The GIST descriptor is a 384-dimensional byte-valued vector. We sample 1,000 K images as the reference set and 100 K images as the queries. The HOG descriptors are extracted from Flickr images, and each HOG descriptor is a 512-dimensional byte-valued vector. We sample 10 M HOG descriptors as the reference set and 100 K as the queries. The BIGANN dataset [11] consists of 1 B 128-dimensional byte-valued vectors as the reference set and 10 K vectors as the queries.

We use the accuracy score to evaluate the search quality. For k-ANN search, the accuracy is computed as r/k, where r is the number of retrieved vectors that are contained in the true k nearest neighbors. The true nearest neighbors are computed by comparing each query with all the reference vectors in the data set. We compare different algorithms by calculating the search accuracy given the same search time, where the search time is computed as the average query cost when accessing a certain number of candidate vectors. We report the performance in terms of search time vs. search accuracy for the first three datasets. Those results are obtained with 64 bit programs on a 3.4 G Hz quad core Intel PC with 24 G memory.

18.5.2 Empirical Analysis

The index structure construction in our approach includes partitioning the vector into m subvectors and grouping the vectors of each partition into n clusters. We conduct experiments to study how they influence the search performance. The results over the 1 M SIFT and 1 M GIST datasets are shown in Fig. 18.2. Considering two partitions, it can be observed that the performance becomes better with more clusters for each partition. This is because more clusters produce more bridge vectors and thus more reference vectors are associated with bridge vectors and their distances are much smaller. The result with 4 partitions and 50 clusters per partition gets the best performance as in this case the properties desired for bridge vectors described in Sect. 18.4.1 are more likely to be satisfied.

18.5.3 Comparisons

We compare our approach with state-of-the-art algorithms, including iterative neighborhood graph search [26], original neighborhood graph search (AryaM93) [22], trinary projection (TP) trees [17], vantage point (VP) tree [15], Spill trees [13], FLANN [20], and inverted multi-index [9]. The results of all other methods are

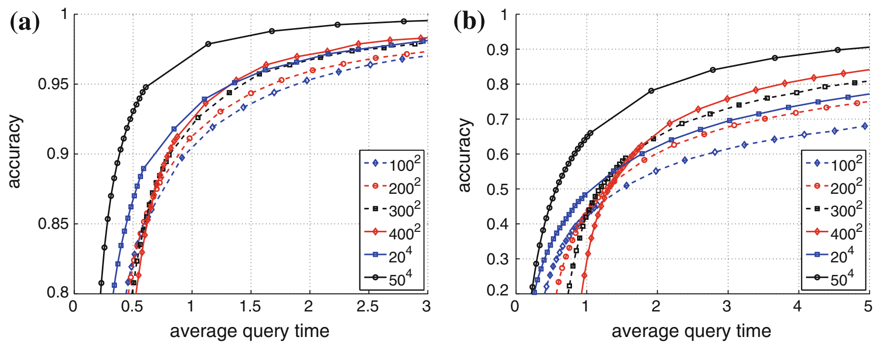


Fig. 18.2 Search performances with different number of partitions and clusters over **a** 1 M SIFT and **b** 1 M GIST. x^y: y means #partitions and x is #clusters

Table 18.1 The parameters of our approach and the statistics

	Size (M)	#partitions	#clusters	#reference (K)	α (%)
SIFT	1	4	50	715	11.4
GIST	1	4	50	599	9.59
HOG	10	4	100	5730	5.73

#reference means the number of reference vectors associated with the bridge vectors, and α means the average number of unique reference vectors associated with each bridge vector

obtained by well tuning parameters. We do not report the results from hashing algorithms as they are much worse than tree-based approach, which is also reported in [19, 20]. The neighborhood graphs of different algorithms are the same, and each vector is connected with 20 nearest vectors. We construct approximate neighborhood graphs using the algorithm [44]. Table 18.1 shows the parameters for our approach, together with some statistics.

The experimental comparisons are shown in Fig. 18.3. The horizontal axis corresponds to search time (milliseconds), and the vertical axis corresponds to search accuracy. From the results over the SIFT dataset shown in the first row of Fig. 18.3, our approach performs the best. We can see that, given the target accuracy 90 % 1-NN and 10-NN, our approach takes about $\frac{2}{3}$ time of the second best algorithm, iterative neighborhood graph search.

The second row of Fig. 18.3 shows the results over the GIST dataset. Compared with the SIFT feature (a 128-dimensional vector), the dimension of the GIST feature (384) is larger and the search is hence more challenging. It can be observed that our approach is still consistently better than other approaches. In particular, the improvement is more significant, and for the target precision 70 % our approach takes only half time of the second best approach, from 1 to 100 NNs. The third row of Fig. 18.3 shows the results over the HOG dataset. This dataset is the most difficult because it contains more (10 M) descriptors and its dimension is the largest (512). Again, our approach achieves the best results. For the target accuracy 70 %, the search time in the case of 1 NN is about $\frac{4}{7}$ of the time of the second best algorithm.

All the neighborhood graph search algorithms outperform the other algorithms, which shows that the neighborhood graph structure is good to index vectors. The superiority of our approach to previous neighborhood graph algorithms stems from that our approach exploits the bridge graph to help the search. Inverted multi-index does not produce competitive results because its advantage is small index structure size but its search performance is limited by an unfavorable trade-off between the search accuracy and the time overhead in quantization. It is shown in [9] that inverted multi-index works the best when using a second-order multi-index and a large codebook, but this results in high quantization cost. In contrast, our approach benefits from the neighborhood graph structure so that we can use a high-order product quantizer to save the quantization cost.

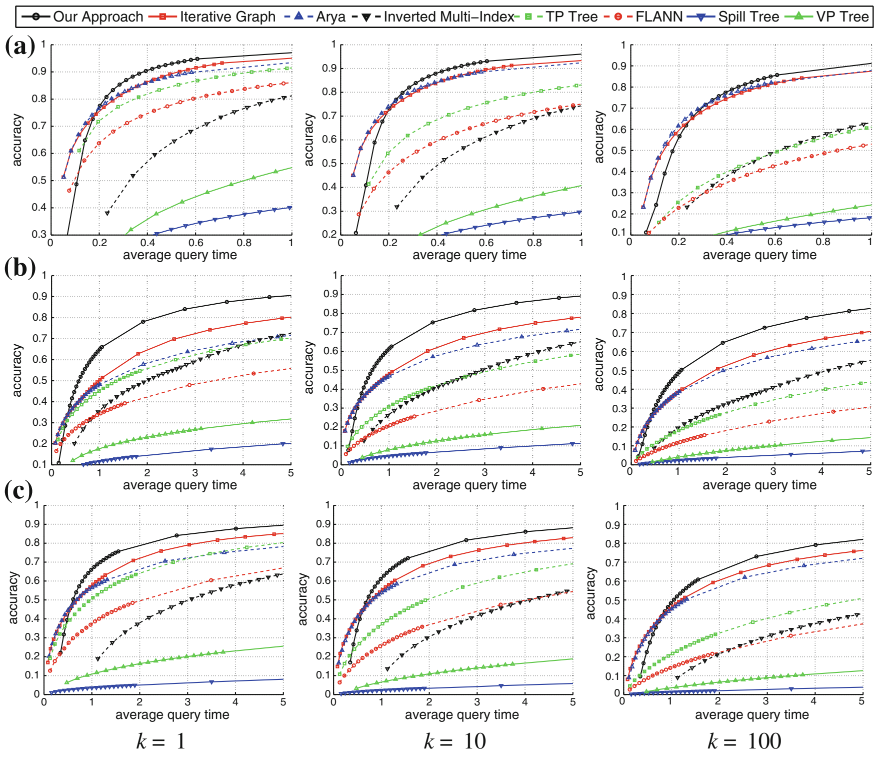


Fig. 18.3 Performance comparison on **a** 1 M 128-dimensional SIFT features, **b** 1 M 384-dimensional GIST features, and **c** 10 M 512-dimensional HOG features. k is the number of target nearest neighbors

In addition, we also conduct experiments to compare the source coding based ANN search algorithm [1]. This algorithm compresses each data vector into a short code using product quantization, resulting in the fast approximate distance computation between vectors. We report the results from the IVFADC system that performs the best as pointed in [1] over the 1M SIFT and GIST features. To compare IVFADC with our approach, we follow the scheme in [1] to add a verification stage to the IVFADC system. We cluster the data points into K inverted lists and use a 64-bits code to represent each vector as done in [1]. Given a query, we first find its M nearest inverted lists, then compute the approximate distance from the query to each of the candidates in the retrieved inverted lists. Finally we re-rank the top L candidates using Euclidean distance and compute the 1-recall [1] of the nearest neighbor (the same to the definition of the search accuracy for 1-NN). Experimental results show that $K = 2048$ gets superior performance. Figure 18.4 shows the results with respect to the parameters M and L. One can see that our approach gets superior performance.

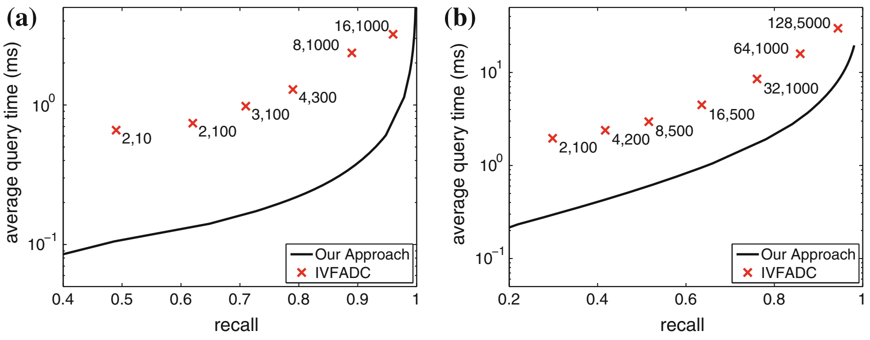


Fig. 18.4 Search performances comparison with IVFADC [1] over **a** 1 M SIFT and **b** 1 M GIST. The parameters M (the number of inverted lists visited), L (the number of candidates for re-ranking) are given beside each marker of IVFADC

18.5.4 Experiments over the BIGANN Dataset

We evaluate the performance of our approach when combining it with the IVFADC system [1] for searching very large scale datasets. The IVFADC system organizes the data using inverted indices built via a coarse quantizer and represents each vector by a short code produced by product quantization. During the search stage, the system visits the inverted lists in ascending order of the distances to the query and re-ranks the candidates according to the short codes. The original implementation only uses a small number of inverted lists to avoid the expensive time cost in finding the exact nearest inverted indices. The inverted multi-index [9] is used to replace the inverted indices in the IVFADC system, which is shown better than the original IVFADC implementation [1].

We propose to replace the nearest inverted list identification using our approach. The good search quality of our approach in terms of both accuracy and efficiency makes it feasible to handle a large number of inverted lists. We quantize the 1 B features into millions (6 M in our implementation) of groups using a fast approximate k-means clustering algorithm [45], and compute the centers of all the groups forming the vocabulary. Then we use our approach to assign each vector to the inverted list corresponding to the nearest center, producing the inverted indices. The residual displacement between each vector and its center is quantized using product quantization to obtain extra bytes for re-ranking. During the search stage, we find the nearest inverted lists to the query using our approach and then do the same reranking procedure as in [1, 9].

Following [1, 9] we calculate the recall@T scores of the nearest neighbor with respect to different length of the visited candidate list L and different numbers of extra bytes, $m = 8, 16$. The recall@T score is equivalent to the accuracy for the nearest neighbor if a short list of T vectors is verified using exact Euclidean distances [11]. The performance is summarized in Table 18.2. It can be seen that our approach

Table 18.2 The performance (recall for the top-1, top-10, and top-100 candidates after reranking and average search time in milliseconds) comparison between IVFADC [11], Multi-D-ADC [9] and Our approach (Graph-D-ADC)

System	List len	R@1	R@10	R@100	Time
BIGANN, 1 billion SIFTs, 8 bytes per vector					
IVFADC	4 million	0.100	0.280	0.600	960
Multi-D-ADC	10,000	0.165	0.492	0.726	29
Multi-D-ADC	30,000	0.172	0.526	0.824	44
Multi-D-ADC	100,000	0.173	0.536	0.870	98
Graph-D-ADC	10,000	0.199	0.562	0.802	24
Graph-D-ADC	30,000	0.201	0.584	0.873	39
Graph-D-ADC	100,000	0.201	0.589	0.896	90
BIGANN, 1 billion SIFTs, 16 bytes per vector					
IVFADC	4 million	0.220	0.610	0.890	1135
Multi-D-ADC	10,000	0.324	0.685	0.755	30
Multi-D-ADC	30,000	0.347	0.777	0.891	47
Multi-D-ADC	100,000	0.354	0.813	0.959	109
Graph-D-ADC	10,000	0.374	0.764	0.831	24
Graph-D-ADC	30,000	0.391	0.829	0.924	39
Graph-D-ADC	100,000	0.395	0.851	0.964	92

IVFADC uses inverted lists with $K = 1024$, Multi-D-ADC uses the second-order multi-index with $K = 2^{14}$ and our approach use inverted lists with $K = 6\,\text{M}$

consistently outperforms Multi-D-ADC [9] and IVFADC [1] in terms of both recall and time cost when retrieving the same number of visited candidates. The superiority over IVFADC stems from that our approach significantly increases the number of inverted indices and produces space partitions with smaller (coarse) quantization errors and that our system accesses a few coarse centers while guarantees relatively accurate inverted lists. For inverted multi-index approach, although the total number of centers is quite large the data vectors are not evenly divided into inverted lists. As reported in the supplementary material of [9], 61 % of the inverted lists are empty. Thus the quantization quality is not as good as ours. Consequently, it performs worse than our approach.

18.6 Analysis and Discussion

Index structure size In addition to the neighborhood graph and the reference vectors, the index structure of our approach includes a bridge graph and the bridge vectors. The number of bridge vectors in our implementation is $O(N)$, with N being the number of the reference vectors. The storage cost of the bridge vectors are then $O(\sqrt[m]{N})$, and the cost of the bridge graph is also $O(N)$. In the case of 1 M 384-dimensional

GIST byte-valued features, without optimization, the storage complexity (125 M bytes) of the bridge graph is smaller than the reference vectors (384 M bytes) and the neighborhood graph (160 M bytes). The cost of KD trees, VP trees, and TP trees are $\sim$180 M, $\sim$180 M, and $\sim$560 M bytes. In summary, the storage cost of our index structure is comparable with those neighborhood graph and tree-based structures.

In comparison to source coding [1, 11] and hashing without using the original features, and inverted indices (e.g. [9]), our approach takes more storage cost. However, the search quality of our approach in terms of accuracy and time is much better, which leaves users for algorithm selection according to their preferences to less memory or less time. Moreover the storage costs for 1 M GIST and SIFT features ($<$1 G bytes) and even 10 M HOG features ($<$8 G bytes) are acceptable in most today's machines. When applying our approach to the BIGANN dataset of 1 B SIFT features, the index structure size for our approach is about 14 G for $m = 8$ and 22 G for $m = 16$, which is similar with Multi-D-ADC [9] (13 G for $m = 8$ and 21 G for $m = 16$) and IVFADC [1] (12 G for $m = 8$ and 20 G for $m = 16$).

Construction complexity The index construction process consists of bridge vector computation via product quantization, bridge graph construction, and neighborhood graph construction. Bridge vector computation includes the quantization process over m groups of N $\frac{d}{m}$-dimensional vectors, whose total complexity is $O(NndQ)$ with Q being the number of iterations in the k-means algorithm. Bridge graph construction consists of 1) building m ordered sequences, which takes $O(N(nd + mn\log n))$ (nd corresponds to the distance computation of a reference point to mn $\frac{d}{m}$ product quantization centers and $mn\log n$ corresponds to sorting m sequences with the length of each sequence is n), 2) finding top t nearest bridge vectors for the reference points, which takes $O(Nt(\log t + m))$, and 3) finding top b reference vectors for a bridge vector from the reference vectors connecting to this bridge vector, which takes $O(Ntb)$. Here is the explanation for the third step. Suppose that t_j is the number of reference vectors connecting the bridge vector $\mathbf{y}_j$, finding top b reference vectors takes $O(t_j b)$. Then the total complexity for all bridge vectors is $O(\sum_{n=j}^{J} t_j b) = O(b\sum_{n=j}^{J} t_j) = O(Ntb)$, where J is the number of bridge vectors and the second equality comes from that the number of all the connections is $\sum_{n=j}^{J} t_j = Nt$. Approximate neighborhood graph construction using the algorithm [44] takes $O(N\log N)$ time. In our experiments, using a 3.4 G Hz quad core Intel PC, the index structures of the 1 M SIFT data, the 1 M GIST data, and the 10 M HOG data can be built within half an hour, an hour, and 10 hours, respectively. These time costs are relatively large but acceptable as they are offline processes.

The algorithm of combining our approach with the IVFADC system [1] over the BIGANN dataset of size 1 billion requires the similar construction cost with the state-of-the-art algorithm [9]. Because the number of data vectors is very large (1 B), the most time-consuming stage is to assign each vector to the inverted lists and both take about 2 days. The structure of our approach organizing the 6 M centers takes only a few hours, which is relatively small. These construction stages are all run with 48 threads on a server with 12 AMD Opteron 1.9 GHz quad core processors.

Search complexity The search procedure of our approach consists of the distance computation over the subvectors, the traversal over the bridge graph and the neighborhood graph. The distance computation over the subvectors is very cheap and takes small constant time (about the distance computation cost with 100 vectors in our experiments). Compared with the number of reference vectors that are required to reach an acceptable accuracy (e.g., the number is about 4,800 for accuracy 90 % in the 1 M 384-dimensional GIST feature data set), such time cost is negligible.

Besides the computation of the distances between the query vector and the visited reference vectors, the additional time overhead comes from maintaining the priority queue and querying the bridge vectors using the multi-sequence algorithm. Given there are T reference vectors that have been discovered, it can be easily shown that the main queue is no longer than T. Consider the worst case that all the T reference vectors come from the bridge graph, where each bridge vector is associated with α unique reference vectors on average (the statistics for α in our experiments is presented in Table 18.1), we have that $\frac{T}{\alpha}$ bridge vectors are visited. Thus, the maintenance of the main queue takes $O((1+\frac{1}{\alpha})T\log T)$ time. Extracting $\frac{T}{\alpha}$ bridge vectors using the multi-sequence algorithm [9] takes $O(\frac{T}{\alpha}\log(\frac{T}{\alpha}))$. Consequently the time overhead on average is $O((1+\frac{2}{\alpha})T\log T - \frac{T}{\alpha}\log\alpha) = O(T\log T)$.

Figure 18.5 shows the time cost of visiting 10 K reference vectors in different algorithms on two datasets. Linear scan represents the time cost of computing the distances between a query and all reference vectors. The overhead of a method is the difference between the time cost of this method and that of linear scan. We can see that the inverted multi-index takes the minimum overhead and our approach is the second minimum. This is because our approach includes extra operations over the main queue.

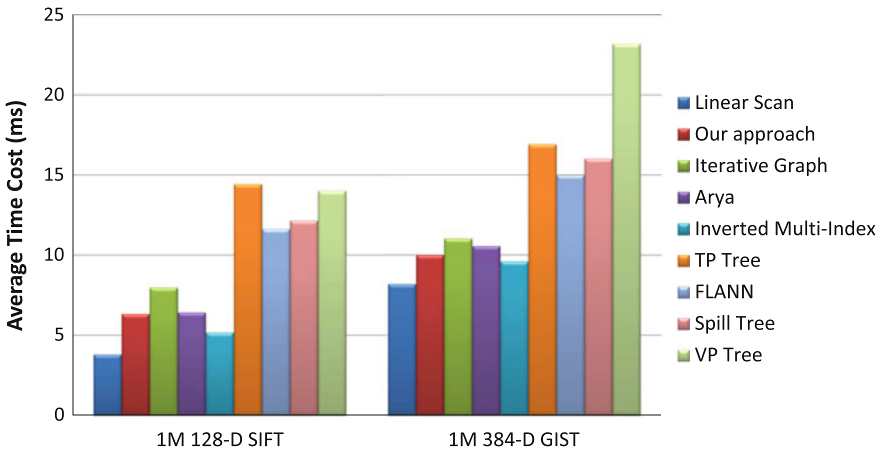


Fig. 18.5 Average time cost of visiting 10 K reference vectors. The time overhead (difference between the average time cost and the cost of linear scan) of our approach is comparably small

Relations to source coding [1] *and inverted multi-index* [9] Product quantization (or generally Cartesian concatenation) has two attractive properties. One property is that it is able to produce a large set of concatenated vectors from several small sets of subvectors. The other property is that the exact nearest vectors to a query vector from such a large set of concatenated vectors can be quickly found using the multi-sequence algorithm. The application to source coding [1] exploits the first property, thus results in fast distance approximation. The application to inverted multi-index [9] makes use of the second property to fast retrieve concatenated quantizers. In contrast, our approach exploits both the two properties: the first property guarantees that the approximation error of the concatenated vectors to the reference vectors is small with small sets of subvectors, and the second property guarantees that the retrieval from the concatenated vectors is very efficient and hence the time overhead is small.

18.7 Conclusions

The key factors contribute to the superior performance of our proposed approach include: (1) Discovering NN candidates from the neighborhood of both bridge vectors and reference vectors is very cheap; (2) The NN candidates from the neighborhood of the bridge vector have high probability to be true NNs because there are a large number of effective bridge vectors generated by Cartesian concatenation; (3) Retrieving nearest bridge vectors is very efficient. The algorithm is very simple and is easily implemented. The power of our algorithm is demonstrated by the superior ANN search performance over large scale SIFT, HOG, and GIST datasets, as well as over a very large scale dataset, the BIGANN dataset of 1 billion SIFT features through the combination of our approach with the IVFADC system.

Acknowledgments This work was partially supported by the National Basic Research Program of China (973 Program) under Grant 2014CB347600.

References

1. Jégou H, Douze M, Schmid C (2011) Product quantization for nearest neighbor search. IEEE Trans. Pattern Anal. Mach. Intell. 33(1):117–128
2. Arya S, Mount DM, Netanyahu NS, Silverman R, Wu AY (1998) An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. J ACM 45(6):891–923
3. Beis JS, Lowe DG (1997) Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: Proceedings of CVPR, pp 1000–1006
4. Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517
5. Friedman JH, Bentley JL, Finkel RA (1977) An algorithm for finding best matches in logarithmic expected time. ACM Trans Math Softw 3(3):209–226
6. Beygelzimer A, Kakade S, Langford J (2006) Cover trees for nearest neighbor. In: Proceedings of ICML, pp 97–104

7. Hwang Y, Han B, Ahn HK (2012) A fast nearest neighbor search algorithm by nonlinear embedding. In: Proceedings of CVPR, pp 3053–3060
8. Wang J, Wang J, Zeng G, Gan R, Li S, Guo B (2013) Fast neighborhood graph search using Cartesian concatenation. In: Proceedings of ICCV, pp 2128–2135
9. Babenko A, Lempitsky VS (2012) The inverted multi-index. In: Proceedings of CVPR, pp 3069–3076
10. Torralba AB, Fergus R, Freeman WT (2008) 80 million tiny images: a large data set for nonparametric object and scene recognition. IEEE Trans. Pattern Anal. Mach. Intell. 30(11):1958–1970
11. Jégou H, Tavenard R, Douze M, Amsaleg L (2011) Searching in one billion vectors: re-rank with source coding. In: Proceedings of ICASSP, pp 861–864
12. Dasgupta S, Freund Y (2008) Random projection trees and low dimensional manifolds. In: Proceedings of STOC, pp 537–546
13. Liu T, Moore AW, Gray AG, Yang K (2004) An investigation of practical approximate nearest neighbor algorithms. In: Proceedings of NIPS, pp 825–832
14. Moore AW (2000) The anchors hierarchy: using the triangle inequality to survive high dimensional data. In: Proceedings of UAI, pp 397–405
15. Yianilos PN (1993) Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proceedings of SODA, pp 311–321
16. Nistér D, Stewénius H (2006) Scalable recognition with a vocabulary tree. In: Proceedings of CVPR, vol 2, pp 2161–2168
17. Jia Y, Wang J, Zeng G, Zha H, Hua XS (2010) Optimizing kd-trees for scalable visual descriptor indexing. In: Proceedings of CVPR, pp 3392–3399
18. Silpa-Anan C, Hartley R (2008) Optimised kd-trees for fast image descriptor matching. In: Proceedings of CVPR, pp 1–8
19. Wang J, Wang N, Jia Y, Li J, Zeng G, Zha H, Hua XS (2013) Trinary-projection trees for approximate nearest neighbor search. IEEE Trans. Pattern Anal. Mach. Intell. 36(2):388–403
20. Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. VISSAPP 1:331–340
21. Aoyama K, Saito K, Sawada H, Ueda N (2011) Fast approximate similarity search based on degree-reduced neighborhood graphs. In: Proceedings of KDD, pp 1055–1063
22. Arya S, Mount DM (1993) Approximate nearest neighbor queries in fixed dimensions. In: Proceedings of SODA, pp 271–280
23. Hajebi K, Abbasi-Yadkori Y, Shahbazi H, Zhang H (2011) Fast approximate nearest-neighbor search with k-nearest neighbor graph. In: Proceedings of IJCAI, pp 1312–1317
24. Samet H (2006) Foundations of multidimensional and metric data structures. Elsevier, Amsterdam
25. Sebastian TB, Kimia BB (2002) Metric-based shape retrieval in large databases. In: Proceedings of ICPR, vol 3, pp 291–296
26. Wang J, Li S (2012) Query-driven iterated neighborhood graph search for large scale indexing. In: Proceedings of ACM multimedia, pp 179–188
27. Datar M, Immorlica N, Indyk P, Mirrokni VS (2004) Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of symposium on computational geometry, pp 253–262
28. Lv Q, Josephson W, Wang Z, Charikar M, Li K (2007) Multi-probe lSH: Efficient indexing for high-dimensional similarity search. In: Proceedings of VLDB, pp 950–961
29. He J, Liu W, Chang SF (2010) Scalable similarity search with optimized kernel hashing. In: Proceedings of KDD, pp 1129–1138
30. Jain P, Kulis B, Grauman K (2008) Fast image search for learned metrics. In: Proceedings of CVPR
31. Kulis B, Darrells T (2009) Learning to hash with binary reconstructive embeddings. In: Proceedigs of NIPS, pp 577–584
32. Kulis B, Grauman K (2009) Kernelized locality-sensitive hashing for scalable image search. In: Proceedings of ICCV, pp 2130–2137

33. Raginsky M, Lazebnik S (2009) Locality sensitive binary codes from shift-invariant kernels. In: Proceedings of NIPS
34. Wang J, Kumar S, Chang SF (2010) Semi-supervised hashing for scalable image retrieval. In: Proceedings of CVPR, pp 3424–3431
35. Weiss Y, Fergus R, Torralba A (2012) Multidimensional spectral hashing. ECCV 5:340–353
36. Weiss Y, Torralba AB, Fergus R (2008) Spectral hashing. In: Proceedings of NIPS, pp 1753–1760
37. Gong Y, Lazebnik S (2011) Iterative quantization: a procrustean approach to learning binary codes. In: Proceedings of CVPR, pp 817–824
38. Xu H, Wang J, Li Z, Zeng G, Li S, Yu N (2011) Complementary hashing for approximate nearest neighbor search. In: Proceedings of ICCV, pp 1631–1638
39. Wang J, Wang J, Yu N, Li S (2013) Order preserving hashing for approximate nearest neighbor search. In: Proceedings of ACM multimedia, pp 133–142
40. Sivic J, Zisserman A (2009) Efficient visual search of videos cast as text retrieval. IEEE Trans. Pattern Anal. Mach. Intell. 31(4):591–606
41. Wang J, Wang J, Hua XS, Li S (2012) Scalable similar image search by joint indices. In: Proceedings of ACM multimedia, pp 1325–1326
42. Tu W, Pan R, Wang J (2012) Similar image search with a tiny bag-of-delegates representation. In: Proceedings of ACM multimedia, pp 885–888
43. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: CVPR 2004 workshop on generative-model based vision
44. Wang J, Wang J, Zeng G, Tu Z, Gan R, Li S (2012) Scalable k-NN graph construction for visual descriptors. In: Proceedings of CVPR, pp 1106–1113
45. Wang J, Wang J, Ke Q, Zeng G, Li S (2012) Fast approximate k-means via cluster closures. In: Proceedings of CVPR, pp 3037–3044

Chapter 19
Listen to the Sound of Data

Mark Last and Anna Usyskin (Gorelik)

Abstract Timestamped observations, generally known as time series, may contain valuable information about a variety of natural and man-made phenomena ranging from weather changes to stock markets. Our capability to collect such data has increased dramatically due to advances in computing and sensory technologies. Visualization is known as a very effective tool for interactive data exploration tasks. In this research, we have tested the hypothesis that musical sonification (the use of musical audio) can serve as a viable alternative to visualization of time-series data whenever the visual representation is unavailable or impossible to use. We have developed a time-series sonification technique, which utilizes some important features of Western tonal music to convert univariate and multivariate time series into a musical equivalent. The technique was used to conduct two online user studies, where the subjects were asked questions about the data behavior by listening to a musical display of time series rather than viewing their visual representation. The results of both studies indicate that our methodology for musical representation of time-dependent data allows most users, including people with low musical hearing ability, to successfully perform a variety of common data exploration tasks.

19.1 Introduction

A time series is a sequence of time-dependent values or events, usually recorded at regular intervals. Meteorological measurements, manufacturing process monitoring, and stock markets are just a few examples of a wide variety of sources for time-series data. Traditional time series analysis [8, 13] includes the following tasks:

- *Indexing and retrieval* (given a time series of interest, find the nearest matching time series)

M. Last (✉) · A. Usyskin
Department of Information Systems Engineering, Ben-Gurion University of the Negev, Marcus Family Campus, 84105 Beersheva, Israel
e-mail: mlast@bgu.ac.il

A. Usyskin
e-mail: anna.usyskin@gmail.com

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1_19

- *Clustering* (find natural groupings of the time series)
- *Classification* (assign a time series to one of a set of predefined classes)
- *Segmentation* (approximate a time series by a sequence of piecewise segments)
- *Trend discovery* (identify long-term trends representing seasonal, cyclic or irregular variations)
- *Correlation analysis* (discovering similar/opposite trends between a pair of time series),
- *Event detection* (detect events of interest, especially abnormal events, in a time series).
- *Forecasting* (prediction of future time series values)

All tasks above apply to both univariate and multivariate time series.

A common perceptual tool for interactive data exploration tasks is visualization, which exploits the phenomenal abilities of the human eye to detect complex structures in images. As shown in [18], visualization methods for time-dependent data can answer a variety of questions on the behavior of univariate and multivariate time series. However, a visual display of data cannot be used by everybody. Thus, the eyes of some people (like pilots, drivers, physicians, etc.) may already be overloaded with visual information, whereas certain professionals may need to take urgent decisions based on time-series data while being away from their offices and without having access to a sufficiently large computer monitor. Besides, blind and visually impaired persons may find even the simplest visual displays completely useless. In these and other situations, another human sense, the sense of hearing, can come to the rescue.

According to [11], sonification is defined as "the use of non-speech audio to convey information." Sonification is also referred to as an auditory display as opposed to the visual display. Auditory user interfaces are routinely used by sonar devices, sound alarm systems, Geiger counters, metal detectors, and other tools. Since time series and audio sequences have the same sequential characteristics, a musical sound should provide a particularly good means to represent time series [20]. Another important advantage of time-series sonification is its bandwidth, since the sound is now believed to be largely processed in the right brain, which is responsible for synergistic thinking [24]. Thus, a driver can quickly respond to the horn sounds made by other drivers while listening to the music from his car audio system.

In this chapter, we study the capability of musical sonification to serve as a viable alternative to visualization of time-series data whenever the visual representation is unavailable or impossible to use. We present a general-purpose time-series sonification technique [12], which utilizes some important features of Western tonal music, like melody, rhythm, dynamics, and timbre, to convert univariate and multivariate time series into a musical equivalent. We start with choosing the period of interest (e.g., a day, a quarter, a year, etc.) and the series (time-dependent attributes) to be sonified. According to the approach proposed in [12], each attribute is translated separately into its musical equivalent before playing the entire set of attributes as one "melody". Though the total number of concurrently measured time series in a given data stream may be quite large, we have to take into consideration the natural limitations of the human auditory perception. Consequently, we do not recommend applying this procedure to more than ten attributes simultaneously. Our goal is to obtain a musical sequence with several voices so that altogether we will be able to monitor the time-dependent behavior of all attributes by listening to their sounds.

This chapter is organized as follows. Section 19.2 discusses the previous use of sonification for representation of data. Section 19.3 describes the proposed time-series sonification technique. In Sects. 19.4 and 19.5, we present the results of two online user studies aimed at evaluating the human ability to explore sonified data. Section 19.6 concludes the chapter with a detailed discussion of experimental results and an outline of proposed directions for future research in auditory mining of time series databases.

19.2 Related Work

19.2.1 Overview of Sonification Techniques

Hermann [5] summarizes the requirements for a technique to be called sonification in the following definition:

“Definition: A technique that uses data as input, and generates sound signals (eventually in response to optional additional excitation or triggering) may be called sonification, if and only if

(C1) The sound reflects objective properties or relations in the input data.

(C2) The transformation is systematic. This means that there is a precise definition provided of how the data (and optional interactions) cause the sound to change.

(C3) The sonification is reproducible: given the same data and identical interactions (or triggers) the resulting sound has to be structurally identical.

(C4) The system can intentionally be used with different data, and also be used in repetition with the same data.”

According to the above definition, sonification is an accurate scientific method, which leads to reproducible results, addressing the ear rather than the eye (as visualization would do). As indicated by Hermann [5], subjectivity in human interpretation is common to all techniques that bridge the gap between data and the human sensory system.

The most common sonification techniques include audification, earcons, auditory icons, spearcons, parameter-mapping sonification, and model-based sonification. A brief description of each of these techniques is provided in the following paragraphs. In addition to those techniques, there is also a branch of sonification that puts the user into the loop; this is referred to as interactive sonification [6]. This form of sonification can be integrated with a varying degree of ease into the above listed techniques.

Audification is the most direct transformation of data values into sound: the sound samples (instantaneous sound pressure levels) are directly obtained from the data values. This means that ordered lists of numbers, e.g., seismic data are directly taken as PCM (pulse code modulation) data for a sound. As a result, sounds are played back without interruption. There are a couple of interesting transformations like resampling, time stretching, pitch scaling, dynamic compression, filtering, etc., which allow to adapt the resulting sound to the preferred frequency range of the ear. Audification is best suited to domains where the data itself is generated by a physical process (e.g., waves propagating through material) [1].

Earcons were developed to provide an immediate feedback to a user activity in a graphical user interface (GUI). They can be implemented as simple sounds (such as typing on a touchscreen keyboard sound) or constructed from a lexicon of simple sounds to represent more complex meanings, similarly as words can be combined to form phrases. The lexicon may have elements that vary in rhythm, pitch, timbre, register, and dynamics. The main disadvantage of earcons is the human difficulty to learn a "dictionary" of more than seven symbolic sounds [21].

Auditory icons were also originally designed to provide feedback about activities in a GUI. The auditory icon approach is to map objects and events in the interface onto everyday sounds that represent reminiscent or conceptually related objects and events [4]. The meaning of the sound shall be connected to the information by metaphorical association. For example, when dragging a file icon on the computer desktop to the trashcan icon, a crushing sound could represent the deletion action. If the sound level or complexity would depend on the file size being deleted, this would be a parameterized auditory icon. However, the number of software operations that can be intuitively associated with sounds from the physical world is quite limited.

Recently, an alternative to earcons and auditory icons has emerged which may be able to ameliorate some of the flexibility–learnability trade-off in interface sounds. Spearcons (Speech-Based Earcons) use temporally compressed speech to represent objects, items, or processes with sound. Spearcons have been shown to outperform both earcons and auditory icons and may be especially useful in the design of advanced auditory menus [30] or for representing a large number of items.

Parameter mapping is the most popular sonification technique for representing high-dimensional data as sound. Typically, a data dimension is mapped onto an auditory parameter such as onset, duration, pitch, pitch variation, loudness, position (spatial cues), reverberation, brilliance, etc. Different data variables can be mapped to different auditory parameters at the same time to produce a complex sound. Consequently, flexible, multi-dimensional data displays can be obtained. The resulting sound may become confusing when there are nonlinear relationships between the represented variables [10].

Model-based sonification has been proposed as an alternative framework for computing data-driven sound in [5–7]. The starting point is that sound in the real world is a by-product of physical processes and a complex sound encodes in a holistic way source properties in its temporal evolution. However, since the extraction of source-related information from the sound has been of high importance in the real world (e.g., to recognize arriving predators early) evolution has led to an optimization of these sound processing skills, including the processing hardware, the brain. In addition, one often learns about the world by interacting with the world and interpreting the acoustic feedback. To carry these concepts over to the domain of data exploration, a sonification model defines a kind of "virtual acoustic object," whose setup might be driven by the dataset under analysis. Laws of dynamics (corresponding to the laws of physics in the real world) determine the temporal evolution of a sonification model.

19.2.2 Sonification of Time Series

Important features of the musical sound are that it has a sequential nature, a particular duration, and evolves as a function of time. The order of sounds in the musical sequence is critical—they have to be heard in a given order. Time series depend on time as well and have the same sequential characteristics. Consequently, it can be argued, that the musical sound provides a good means to represent time series [20].

We could find only a few articles discussing the use of sonification for representation of time series, some of them concentrated on combining of visual and audio information [19, 20]. Several other articles described a toolkit or an algorithm that was designed to represent specific types of databases, like meteorological observations, geo-referenced data, EEG, DNA, etc. [3, 11, 28, 29, 31]. There are several works evaluating interactive data mining tasks, like classification [3, 5, 23] and others. All these studies argue that in classification tasks, where a human expert evaluates the data, sonification can improve the classification performance. Only very few works speak about similarity-related tasks. One of them is Pauletto and Hunt's research from 2004 [23], where authors introduce a toolkit for interactive sonification that is supposed to be generic enough to work on different databases (the empirical case studies were conducted for two time-series databases—physiotherapy movement data and helicopter flight data). Pauletto and Hunt define five categories of sonification algorithms that can produce perceptually distinguishable sounds. The identified categories are [23]:

1. Mapping to clearly recognizable perceptual parameters, such as pitch, loudness, and duration.
2. Mapping onto sound variables so that the overall timbre of the sound represents the evolution of the data.
3. Creating a mix of separately perceivable sounds, when wishing to portray many channels of data.
4. Mapping onto many channels of data. This can contribute to the evolution in time of a single sound with a complex timbre.
5. Utilizing parameter redundancy by mapping one channel of data onto more than one sound variable.

The Interactive Sonification Toolkit [23] was used in [22] to evaluate three different levels of interactive sonification. In the first experiment, subjects were asked to navigate and listen to a sonified dataset and then to recognize which data structures were present in the dataset and in which order. Five basic data structures (including random noise) were synthetically produced for this experiment. Each dataset (time series) contained a combination of various data structures. The sonification method used was mapping the datasets to the amplitude of a sine oscillator. In the second experiment, multivariate data from electromyography (EMG) sensors was used for real-time sonification of muscles activity. The EMG data was sonified using a sound synthesis technique called *amplitude modulation* (AM). Both studies have shown that user interaction can increase the usability of sonification systems.

19.3 The Sonification Procedure

19.3.1 The Procedure Overview

The proposed sonification procedure consists of three main steps, or sub-algorithms: the time-series segmentation algorithm (SWAB), the sonification algorithm (SEG->MSQ) translating a segmented database to the MSQ[1] format, and the MSQ->MIDI Converter translating a file in the MSQ format to a standard MIDI file. The input of the sonification procedure is a time series dataset, which can be a univariate or a multivariate one. The procedure output is a MIDI and/or an MP3 music file.

The algorithm interprets each time series as an ordered sequence of values related to a specific attribute. The number of selected attributes should not exceed 10, whereas we do not recommend using this mode for more than 3–4 continuous and another 1–3 nominal attributes due to the natural limitations of human auditory perception. Each procedure step runs separately for only one attribute at a time, including all the statistics calculations, the segmentation algorithm, and the translation to a musical equivalent (using the SEG->MSQ algorithm). We refer to this mode of sonification as "vertical" since we parse the input database by selected attributes (time series) and nearly all calculations are done separately for each attribute without any "horizontal" or "inter-series" relation. As a result of running the sonification algorithm in the vertical mode, we obtain a musical sequence with several voices each playing its own melody, the sounds start and stop only due to changes in a specific attribute, but altogether we hear what happens to all attributes during some period of time or at some specific moment.

Two different kinds of sonification may be used for *continuous data*. We represent continuous attributes (like temperature, price, height, number of items, etc.) by changes in music pitch[2] or by changes in volume (loudness). We have applied four main rules to sonification of continuous data:

- A direct correspondence exists between a numeric value and music pitch/volume. For example, if the value on the first day is greater than on the second one, then the pitch/volume of the first sound will be higher/louder than those of the second sound.

[1] MSQ format was designed by Siegfried Koepf and Bernd Haerpfer in 1998. The idea is to have a platform-independent, easily readable, and editable file format qualified for algorithmic manipulation and composition as well as for real-time controlling MIDI instruments. More details about the MSQ Project are available at [17].

[2] *Pitch* is related to the repetition rate of the waveform of a sound; for a pure tone this corresponds to its frequency.

- The order of numbers is important. For example, if 27 comes before 38 in the series, then the corresponding sound of 27 will be played before the corresponding sound of 38.
- The sound duration is important. If the values of a numerical sequence did not change significantly during a given period, then the pitch/volume of musical sound representing it would not change too and its duration would represent the duration of the sequence.
- It is important to clarify that representation by volume changes is used only for multivariate datasets, where one attribute is represented by music pitch and another attribute is represented by volume changes. As a result, we will obtain a "melody" where the pitch of music sound represents one attribute and volume (loudness) represents another attribute from the same dataset.

It is important to understand that volume sonification (sonification by changes of volume) can only be secondary to the regular, pitch sonification (sonification by changes of pitch). An attribute represented by volume sonification is always related to another attribute, represented by pitch sonification.

Besides sonification of continuous data, one may want to sonify *nominal data* as well. For representation of nominal data (e.g., yes/no, exists/does not exist, item categories, etc.) we use the following rules:

- To represent nominal data we use particular musical instruments (usually various drums or bells) and music sound.
- Each attribute is represented by a different instrument and attribute values are mapped to different musical pitches.

Let us introduce each of the mandatory sonification parameters, and then every optional parameter, as well. Besides input and output files, the only mandatory parameter of the vertical sonification mode is the list of attributes selected for sonification. The order of "sonification attributes" is important only if a default set of instruments is used for sonification. In this case, the serial number of a selected attribute determines the instrument representing it (if instruments are not specified). As indicated above, the number of sonification attributes should not be more than 10.

Besides the mandatory parameters, several groups of optional parameters exist. The first two optional parameters are *Percentage of data to be sonified* and *Use the same scale*. The first parameter determines the "compression ratio" (percentage) between the final number of segments to be created and the total number of records in the dataset (values in each time series). If the user is not interested in running any segmentation on the data, s/he should enter 100 for this parameter. If this parameter is not entered, the algorithm will use the default settings for the segmentation step (to be defined later). The second parameter allows using the same scale for representing different attributes. This means that if all selected attributes have the same range of values, it may be useful to keep the same scale, or base, for representing the values of these attributes. For example, if we have a database with closing prices of two stocks, we can use the same scale to represent these prices, making the stock prices comparable to each other. Another purpose of the *Use the same scale* parameter is to keep the range of each attribute and then run sonification and get representation with each attribute "melody" playing in its own "range". After this scaling operation, it is much easier to interpret what happens at each moment to every attribute.

Three optional sonification types for each attribute in the list of sonification attributes (mandatory parameter defined above) include pitch-based sonification (default for continuous attributes), volume-based sonification (another option for continuous attributes), and nominal sonification (for all nominal attributes). There can be maximum three nominal sonification attributes. By our definition, each attribute represented by volume should correspond to one of attributes represented by pitch. This results in a maximum of five pairs of pitch-based and volume-based attributes.

For nominal attributes, there are two additional parameters: *Default value* and *Sonify default value*. The first parameter indicates the user-specified "default" value of each nominal attribute, which is considered more significant than all other values. The second parameter determines the sonified values of the attribute (all values or just those that are different from the default value). For some binary-valued attributes like *yes/no* and *exists/does not exist* it makes sense to sonify only "positive" values (*yes*, *exists*), while "negative" values (*no*, *does not exist*) would not be represented at all. For attributes where all values are equally important (like various item categories), it is better to sonify each value by a unique musical sound played by the same instrument (per attribute).

The last, but not the least group of optional parameters defines the instruments used for sonification. We defined two different default sets for continuous and nominal attributes, respectively. If no specific instrument was entered as a parameter for an attribute, the default value is used according to the type of the attribute. It is obvious that no instrument should be used for the volume parameter, because it is only used for volume changes of its paired pitch attribute.

The pseudo-code of the vertical mode of the sonification algorithm for continuous attributes can be found in Fig. 19.1. The algorithm works as follows: it starts with partition of the original input file into the number of sub-files equal to the number of selected attributes and calculates statistics for each new input file. After this stage, each sub-file contains all values of one specific attribute. If the user decides to use a common scale for all selected attributes, the algorithm runs on two arrays, which contain information about minimal and maximal values for each attribute and calculates the minimum and the maximum values of all attributes.

In the next step, the segmentation algorithm runs separately on each selected attribute. If the *inSegPers* (percent of segmentation) parameter equals 100, there is no need to run segmentation and we just simulate the output of the segmentation algorithm by concatenating each input record with the number 2, where 2 is the minimal segment that could be found by a segmentation algorithm. Otherwise, by using statistics calculated for the current attribute, we start iterations of the SWAB (sliding window and bottom-up) segmentation algorithm [9]. The maximum of number of iterations is 20. If the best max_error is not found, the algorithm continues with the best error found after the 20th iteration. Once the best max_error is found, we stop the algorithm. After completing the segmentation step for all attributes, we go to the next step of our algorithm. For each attribute, we calculate the vector statistics for the created segmentation file and use the calculated statistics in the next step

```
Input:
        InFile  - a univariate or a multivariate time series dataset
        NumCols - number of attributes (time series) in the dataset
Output:
        OutFile - a MIDI music file
For $j$ = 1 to numCols do
   Run ComputeStatistics function to calculate statistics for the original
    values of $j$
   Run Segmentation function to perform segmentation of $j$
   Run ComputeStatistics function to compute vector statistics for the
    segmented values of $j$
   Run SEG->MSQ function to translate the segmented attribute $j$ to the
    MSQ file format
End for
Merge individual attribute MSQ files into one MSQ file
Convert the file in the MSQ format to a standard MIDI file format
```

Fig. 19.1 Algorithm 1—The vertical mode of sonification

of our procedure—the SEG->MSQ conversion algorithm, where each record of the segmented file is translated to its musical equivalent. The final step of the procedure merges all MSQ files, which were built for each attribute separately, to one complete MSQ file that contains MSQ data of all selected attributes altogether.

19.3.2 The Algorithm Details

Let us discuss the above procedure in more detail. In Step 1 (see Fig. 19.1), we build a separate input file for each selected attribute and calculate statistics on it. The order of files is determined by the order of input attribute parameters. This order is saved until the final step of the sonification procedure, where many output files of SEG->MSQ algorithm are merged together into one final output file. For each record of the original input time series database, we find the value of the current attribute (according to the attributes order), and calculate its minimal and maximal values using the *ComputeStatistics* algorithm. These values are saved in *Min* and *Max* arrays respectively, and besides that, we count the total number of elements (records) in the dataset (n). The *Range* variable of each attribute is calculated as the difference between its maximal and minimal values.

In Step 2 of the sonification procedure, we run the SWAB (sliding window and bottom-up) segmentation algorithm [9] separately for each attribute. The SWAB segmentation algorithm gets four parameters—the input file (time series data), the output file (segmented data), the maximal error, and the indication of nominal attributes. After running a number of experiments on time series of different sizes with different values for the number of segments, we chose the appropriate default number of segments as follows. 25–50 % of time series size for time series with less than 100 observations, 20–35 % for time series with 100–200 observations, and 15–25 % for time series with more than 200 observations. If the user is not interested to use the default value for any reason, he can enter his own number of segments as a parameter to the algorithm.

Starting with the default values for the minimum and the maximum error, we run the segmentation algorithm for the first time and get the minimum number of segments for a given time series (the higher the maximum error, the fewer segments will be found). Then we decrease the maximum error (and so increase the number of found segments) trying to narrow the upper and the lower bounds of error by dividing the base by powers of 2 (like in binary search). Every time after running the segmentation algorithm with the current maximal error, we test whether this value gives a better approximation for the optimal number of segments, and so is a better upper or lower bound for the optimal maximum error. If so, we advance the appropriate bound to this value. In the beginning, only the upper bound is affected. However, once we found the lower bound that provides more segments than the optimum, we continue to look for the optimal number of segments by smaller steps: the next maximum error is the mean between the current upper and lower bounds.

As follows from our experience with many different time series databases, the optimal maximal error is usually found within 3–4 iterations. The convergence rate depends on the input time series database itself. If the algorithm has not converged within 20 iterations, we stop searching and proceed with the next sonification steps using the segments found at the 20th iteration.

In Step 3, we calculate vector statistics on each segmented file found in the previous step, using again the *ComputeStatistics* algorithm. The statistics of this step are calculated in the same way as in Step 1 of the algorithm. It is important to do this once again because of possible changes that happened to the original values of each attribute due to the segmentation algorithm. We find the updated minimum and maximum values of each attribute and then calculate the *Range* variable as the difference between maximal and minimal values of this attribute. As a result, we get an updated array of minimal values and an array of *Range* values. The size of each array equals to the number of selected attributes.

In Step 4 (see Fig. 19.2), we translate the segmented file of each attribute to its musical equivalent using the SEG->MSQ sonification algorithm. This step deals with all three possible kinds of sonification—sonification of continuous attributes by pitch changes, sonification of continuous attributes by volume changes, and sonification of nominal attributes. For pitch attributes, the algorithm translates each value of every

```
Input:    $S_j$ – number of segments in the attribute $j$
          $Count_{jk}$ – number of observations in a segment $k$ of the attribute
          $j$ ($k = 1,\ldots, S_j$)
          $Value_{jk}$ – the value representing a segment $k$ of the attribute $j$
          (default: mean value) ($k = 1,\ldots, S_j$)
          $Min_j$ - the minimum value representing the segments of $j$
          $Max_j$ - the maximum value representing the segments of $j$
          $Range_j$ – the range of values representing the segments of $j$
          $musBase$ – musical base, the number of tones to be used for
          sonification (default = 36 tones, which equals to 3 octaves, 12
          tones each)
          $musDif$ – musical difference, the lowest pitch that people can
          distinguish (default = 52)
          $minDur$ – the minimal duration of the sound of each segment in
          the final sonified sequence (default = 30 ticks, which equals to half
          a second)
Output:   $P_{jk}$ –pitch value representing segment $k$ of attribute $j$
          $Start\_T_{jk}$ – the start time of the sound of a segment $k$ of the
          attribute $j$ ($k = 1,\ldots, S_j$)
          $Finish\_T_{jk}$ – the end time of the sound of a segment $k$ of the
          attribute $j$ ($k = 1,\ldots, S_j$)
          $N_{jk}$ – the musical note representing a segment $k$ of the attribute
          $j$ ($k = 1,\ldots, S_j$)
Assign channel number, musical instrument, and volume to attribute $j$
$Start\_T_{j1} = 0$            // Set the start time of the first segment to zero
For $k = 1$ to $S_j$ do
    Sonify ( $Start\_T_{jk}$, $Count_{jk}$, $Value_{jk}$)  // Sonify each segment
    Write a new record to the MSQ file (Channel Number, Start_Tjk,
    Finish_Tjk, Pjk, Volume)
End for
```

Fig. 19.2 Algorithm 2—Sonification of continuous attributes

attribute to an appropriate musical pitch according to the range and the minimal value of that attribute. If the calculated pitch value is not equal to one of the seven tones of the C Major scale (0, 2, 4, 5, 7, 9, 11), it is decremented by one. Then the algorithm writes this value in a special format to the output file for the respective attribute. Each attribute obtains its own musical channel (1:10) and either specified or default instrument. Besides this, the pitches are written to output with the start and the stop times determined by the segmented file of the current attribute only. That is why in the output musical sequence we hear different melodies (each voice is played by a different instrument and progressing in its own way). Monitoring a specific instrument gives us an opportunity to know what happens to a specific attribute during a given time period. Nominal attributes are processed very similar; the only difference is that we use other default instruments for sonification, and that each value of a nominal attribute (out of maximum 10) is assigned its own predefined pitch value. Processing a volume attribute is different from processing standard (pitch or nominal) attributes. First, assigning a channel and an instrument to a given pitch attribute depends on the number of previously selected volume attributes. As explained above, the sonification of a volume attribute is secondary to sonification of a previously selected pitch attribute. Then, despite changing pitches according to values of segments in the input file, we change volume parameters for the previous channel—the channel of related pitch attribute. This means that we adjust the volume parameter of the previously selected (pitch) channel according to the segmented data of the current volume attribute. The start time of each new event sent to the volume parameter of a previous channel is determined by the duration of a segment from the segmented input file of the volume attribute (Fig. 19.3).

In the final step (Step 5), we merge all data from the previous SEG->MSQ step to one output file. This file starts with the header and channel definitions, according to the MSQ format, and then it contains all data about pitch and volume changes, sorted by the timestamp argument. The vertical mode is evaluated empirically in the following two sections on collections of univariate and bivariate time series, respectively.

19.3.3 Illustrative Example

In this example, we demonstrate the sonification of the univariate time series no. 1 in our online repository of musical examples [25]. The original 20 values of the continuous series are shown in Fig. 19.4 and its segmentation by the SWAB algorithm (resulting in 14 segments and *max_error* $= 0$) is presented in Table 19.1. The series has the minimum, the maximum, and the range values of 10, 60, and 50, respectively. The sonification of each segment by Algorithm 2 is presented in Table 19.1. All time variables are presented in ticks (60 ticks $= 1$ s). For example, row no. 1 shows the sonification of the first segment, which contains just one observation and hence its sound duration is 30 ticks (half a second). The original attribute value of 10 is converted into the pitch value of 52 using Algorithm 2. Since this pitch is tonal

```
// Sonify segment k of attribute j
Sonify ($Start\_T_{jk}$, $Count_{jk}$, $Value_{jk}$)
  // Compute the sound duration of segment k as a function of
  // the number of observations in the segment
  $SoundDuration$ <- $Count_{jk} \times minDur$
  // Compute the end time of the current segment
  $Finish\_T_{jk}$ <- $Start\_T_{jk} + SoundDuration$
  // Compute the start time of the next segment
  IF ($k < S_j$) THEN
        $Start\_T_{j,k+1}$ <- $Finish\_T_{jk}$
  // Compute the pitch using min-max normalization
  $CalcPitch$ <- $INT(((Value_{jk} - Min_j) / (Max_j - Min_j))*musBase + musDif)$
  IF ((($CalcPitch - musDif$) modulo 12) is tonal) THEN
    // If tonal, keep the calculated pitch
        $P_{jk}$ <- $CalcPitch$
  ELSE
    // If not tonal, decrement the calculated pitch by one
        $P_{jk}$ <- $CalcPitch - 1$
  END IF
  // Find the note corresponding to the tonal pitch
  $N_{jk}$ <- findNote ( $P_{jk}$ modulo 12)
```

Fig. 19.3 The sonify function (Segment Sonification)

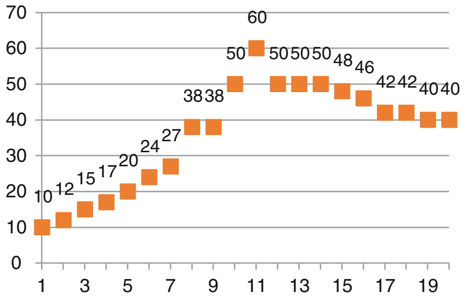


Fig. 19.4 Time series no. 1 original values

according to the C major scale, its value is not changed. In contrast, row no. 14 corresponds to a segment of two observations resulting in the sound duration of 60 ticks (1 s). The mean segment value of 40 is converted into the pitch value of 73, which is not tonal and thus the algorithm decreases it to the nearest lower value of

Table 19.1 Time series No. 1 segmentation and sonification

Segment no	Segment size	Duration	Start time	End time	Average value	Calculated pitch	Tonal pitch	Note
1	1	30	0	30	10	52	52	E
2	1	30	30	60	12	53	53	F
3	1	30	60	90	15	55	55	G
4	1	30	90	120	17	57	57	A
5	1	30	120	150	20	59	59	B
6	1	30	150	180	24	62	62	D
7	1	30	180	210	27	64	64	E
8	2	60	210	270	38	72	72	C
9	1	30	270	300	50	80	79	G
10	1	30	300	330	60	88	88	E
11	4	120	330	450	48	79	79	G
12	1	30	450	480	46	77	77	F
13	2	60	480	540	42	75	74	D
14	2	60	540	600	40	73	72	C

72. The last column of Table 19.1 shows the notes corresponding to the played tonal pitches.

19.4 The First Experiment: Univariate Data

19.4.1 Experimental Design

The goal of this experiment was to evaluate the proposed sonification algorithm on univariate time series. The experiment was supposed to answer the following questions:

- Can people use our sonification algorithm to detect similarity of two time series?
- Can people hear and find consistent patterns in the data?
- How do people interpret and make decisions from this representation?
- Can people perform standard data mining tasks, like classification and clustering using our auditory representation of the data?

Many time series datasets are available on the Internet. In our experiment, we used several popular datasets from the Time Series Data Mining Archive website [27] and some time series constructed from the data available on the websites of an Israeli bank [15] and Tel-Aviv Stock Exchange [26]. Specifically, we used information about some popular stocks and currency exchange rates.

The experiment was performed as an online user study. The study website contained a questionnaire to be filled by the subjects in their language of choice (English or Hebrew). The content of the questionnaire in both languages was the same.

The structure of the questionnaire was the following:

1. User identification and demographic information: username (fictional), age, gender, occupation.
2. Musical ability: musical experience (number of practice years, pure musical hearing).
3. Introduction: the subjects were provided a brief explanation of our approach to sonification of time-series data.
4. Training session: we gave the subjects an opportunity to try answering each of the typical questions and explained how to approach each of the questions in order to provide correct answers.
5. Evaluation session: the subjects received five sets of questions; each set consisting of four types of tasks specified below.

Task 1: Listen to the musical sequence, representing some data, and answer, when the data values were higher: in the beginning of the sequence, at the end of the sequence, or they were the same in the beginning and at the end. (Choose one out of three possible answers). This question was designed to check the user's ability to detect changes in data and to test the general ability of the user to "understand" musical information—his/her level of musical hearing.

Task 2: Are sounds propagating in some well-defined direction in most cases, or is the sound's propagation is rather chaotic? (Choose one out of two possible answers). This question was designed to test the user's ability to identify any well-defined direction of propagation. This task was found ambiguous during the evaluation of the experimental results, and its answers can be discarded.

Task 3: Which of the following charts describes the data the best way? (Choose one out of five charts). This task was designed to test the user's ability to classify a given object into one of predefined categories. The object here is a musical sequence and the categories are defined by very schematic visual representations of the following sequences: monotonically increasing, monotonically decreasing, first increasing then decreasing, first decreasing then increasing, and chaotic.

Task 4: Which of the two musical sequences (Seq B/Seq C) is more similar to the original sequence (Seq A)? (Choose one out of two answers). This task was designed to test the user's ability to perform one of the most important data mining tasks—similarity search (an essential operation in clustering).

All univariate time series in this experiment were represented by the pitch-based sonification (see Section y2 above). The subjects received all data "as is", without any possibility to change any sonification parameters. Before each set of questions, there were brief explanations about the data source (e.g., day closing stock price for some period, currency exchange rate, etc…). Only fully completed questionnaires were taken into consideration. An example of a sonified univariate time series presented to the subjects in this experiment is shown in Fig. 19.4.

The only dependent variable in this experiment was the subject's total score S. The independent variables were the subject's demographic and musical characteristics: Gender G (m/f), Age A (6 groups), Occupation O (8 groups), Musical Experience E (4 groups), Absolute Pitch Ability P (yes/no).

19.4.2 Research Hypotheses

Our null hypothesis was that the subjects were unable to provide correct answers to the questions using musical sequences, which were generated from the original time series by our sonification technique. The alternative hypothesis was that the subjects were able to provide correct answers to the questions using generated musical sequences.

We compared the results of the test questions to random decisions. If the subjects were guessing their answers randomly, then the expected results would be distributed uniformly with an equal probability of every possible answer. To determine how well the subjects were performing, we compared the result distributions to the uniform distributions that would be obtained by chance.

After rejecting the null hypothesis and then proving that our technique is good enough for sonification of univariate time series, some other hypotheses were tested:

- Musical experience of the subjects has an influence on their performance.
- Subject's age has an effect on his/her performance.
- Subject's gender has an effect on his/her performance.
- Subject's occupation has an effect on his/her performance.

By the subject's "performance", we mean the total number of correct answers the subject gave to the questions of the test (experiment). By "technique is good enough" we mean that our technique is better than random guesses for each of the questions as well as for all questions together and that the average number of correct answers will be more than 2/3 (66 %).

19.4.3 Subjects

The experiment was carried out using the experimental website of our online user study. Its web address was posted on several student mailing lists and through the mailing list of "Ihud"—the Israel Kibbutz choir. The experiment took place from July to September 2007. During that period, 44 subjects took part in the experiment (only those who answered all questions). There were 27 male and 17 female subjects between the ages of 18 and 70. The age distribution of the subjects is shown in Fig. 19.5. Only one subject claimed to have absolute pitch hearing, no one had any familiarity with any sonification technique, only few deal with time series databases

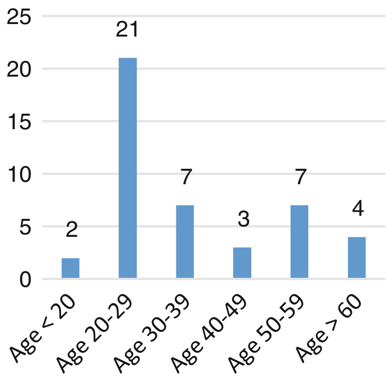


Fig. 19.5 Experiment 1: Age distribution

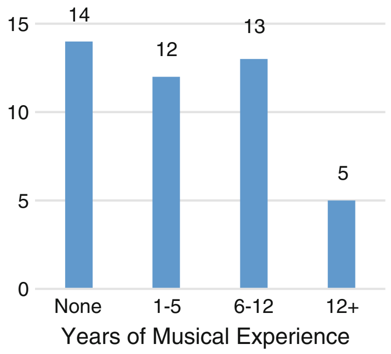


Fig. 19.6 Experiment 1: Musical experience distribution

and data mining techniques. The distribution of the subjects' musical experience (if any) is shown in Fig. 19.6. Each subject could participate in the test at his/her time of convenience and no time limitations were imposed either at the stage of explanations and examples, or during the test itself. All necessary information for completing the test was posted on the experimental website. No previous experience in the time series data mining tasks, sonification techniques, or the dataset domain (e.g., finance) was required.

19.4.4 Experimental Results

The test was designed in increasing order of difficulty. We started from relatively simple time series (the first set) and proceeded to the last series (the fifth set) which was relatively complex one. The total score for the test was calculated as follows:

- Each correct answer to a multiple-choice question has received one point.
- No points were deducted for a wrong answer.

Table 19.2 Number of correct answers per task

	Percentage of correct answers (%)	Percentage of correct answers by chance (%)	P-value
Task 1	86.36	33	0.00000
Task 2	66.36	50	0.00000
Task 3	76.36	20	0.00000
Task 4	81.36	50	0.00000

Table 19.3 Number of correct answers per set of questions

	Number of answers	Number of correct answers	Percentage of correct answers (%)
Set 1	176	167	94.89
Set 2	176	138	78.41
Set 3	176	130	73.86
Set 4	176	120	68.18
Set 5	176	128	72.73

Hence, the maximum total score for each set was 4 points, and the maximum total score for the test was 20 points (5 sets, 4 questions in each set). The sample size of 44 subjects enabled us to reach conclusions at the significance level of 0.05 and higher.

In the first phase of analysis, we calculated the significance of the results for each task (Table 19.2), and for each set of questions (Table 19.3). We compared the results against chance using a two-tailed binomial distribution. The analysis of results per task shows that for each task, the subjects performed at a level significantly above chance, whereas the best result was for first task, and the worst was for second task. The interesting fact is that subjects perform significantly better (81.36 %) than chance answering the fourth task. It means that people can detect similarity of sequences by listening. The results for the third task are significantly better than chance as well. It means that people can classify a given sequence into given clusters (classes). Here, the number of possible answers was the highest (5), and nevertheless people were able to choose the correct answer in 76.36 % cases. The analysis of results per set of questions shows that for each set of questions, the subjects performed at a level significantly above chance as well, whereas the best result was for 1st set of questions (94.89 %), and the worst was for the 4th set of question (68.18 %).

Given our main goal to test the hypothesis that our sonification algorithm is a useful method for decision making and time series datasets exploration including the ability to perform some popular data mining tasks, our next hypothesis is defined, as follows: "Some information about a subject can be useful to predict her/his result in the experiment." In other words, some people are more likely to succeed working with sonified data than others. Then our goal is to find personal attributes/abilities that affect the results. In the first part of the test, our subjects were asked some simple questions. Here they are:

- Age; 6 categories: (<20, 20–29 ,30–39, 40–49, 50–59, >60)

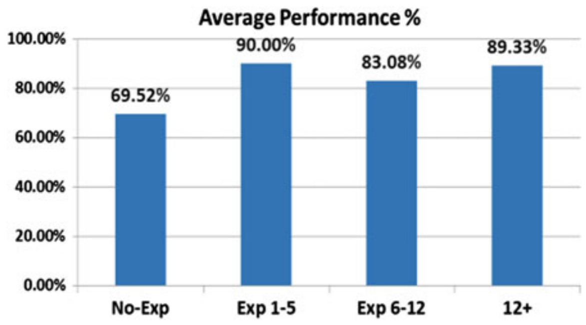


Fig. 19.7 Experiment 1: The effect of musical experience

- Gender; 2 categories: m/f
- Occupation; 8 categories: student(stu), lecturer/investigator(lec), teacher(tea), engineer(eng), manager(man), journalist(jou), musician(mus), other(oth).
- Musical experience; 4 categories: No-Exp (none), Exp 1–5 (1–5 years playing some instrument or studying vocal), Exp 6–12 (6–12 years playing some instrument or studying vocal), 12+ (more than 12 years playing some instrument or studying vocal).
- Absolute pitch ability; two categories: yes/no.

According to the results of ANOVA test, there is no statistically significant difference between total scores for different age, gender, or occupation groups. However, there is statistically significant difference between total scores for different musical experience groups. The performance of each group is shown in Fig. 19.7. Although subjects without *any* musical experience certainly performed worse, the *amount* of musical experience does not seem to matter for the experienced subjects' performance.

19.5 The Second Experiment: Bivariate Data

19.5.1 Experimental Design and Variables

The goal of this study was to evaluate the proposed sonification algorithm on bivariate time series. Specifically, we were interested to test the user's ability to "hear" the information represented by each one of the voices, to understand what is represented and to make decisions relying on a given representation. Additional objective of this experiment was to evaluate various methods of sonification. Particularly, we evaluated three variants of sonification of bivariate time series: one nominal + one continuous attribute, where the last one is represented by pitch changes; two continuous attributes, where the first one is represented by pitch changes and the second one—by changes of volume; and two continuous attributes represented by pitch changes on two different instruments. In our experiment, we used piano and flute timbres to represent continuous attributes, and bell sound to represent the nominal one.

Another objective was to explore possible association between the subject's musical ability and his/her success in answering the questionnaire. This time, in contrast to the first experiment, we used three parameters to test the musical ability. One was "musical experience"—it is an objective parameter, where the user can choose one of four answers according to the number of years he studied music. The second one was subjective—the subjects were asked to indicate their level of musical hearing. This question was aimed at identifying subjects with good musical hearing, who have no formal musical education. This question was also supposed to identify the subjects with formal musical education, whose musical hearing was not developed too much. Some of these users for example, could study music during their childhood, but never returned to this activity.

The last question connected to music ability was not very straightforward—the subject's occupation. Relying on the results of the first experiment, we were able to recognize some associations between the subject's profession/occupation, and his/her musical ability. It is obvious, that if a subject is closely involved with music in his everyday life either as a professional musician, or as a music hobbyist, his musical ability is potentially high. However, we also tried to find some other occupations where people could use our algorithm for their professional or other needs. Consequently, we allowed the subjects to choose multiple occupations. For example, the second author of this chapter would choose "student", "engineer" and "musician" altogether. During the analysis of the results, we could refer each user to any of the occupation categories he/she chose.

The second experiment, like the first one, was performed as an online user study and it took place between February and April 2008 using the same experimental website (see subsection 4.3 above). The questionnaire had the same sections as the first experiment: User Identification, Musical Ability, Introduction, Training Session, and Evaluation Session. The Evaluation Session included the following five types of tasks:

Task 1: This task was designed to test the user's ability to detect changes in data and to "understand" musical information, i.e., to test his level of musical hearing. In this task, we have sonified one nominal and one continuous attribute from a weather time series. The nominal attribute referred to a weather event (e.g., raining/snow). Each time there was an event we used the same bell sound. The continuous attribute (e.g., temperature) was played by piano voice with pitch sonification. In the first question, the subjects were asked to count the events during the given period, whereas in the second question, they were asked about the trend (increasing / decreasing / stable) in the continuous attribute. Two such datasets are included in the Mul1 folder of our online repository [25]. One of the datasets (Weather-April2007) is shown in Fig. 19.8.

Task 2: This task was designed to test the user's ability to distinguish between changes in two real-valued time series represented by pitch and volume, respectively. Since using volume to represent numerical values can be problematic, we represented the more important time series by pitch, and the less important series by volume. In the two questions, the subjects were asked about the trend in the attributes represented by each sonification method. An example of such dataset (included in the Mul2 folder of our online repository [25]) is shown in Fig. 19.9.

Task 3: This task was designed to test the user's ability to distinguish between changes in two real-valued time series each played by a different instrument (piano, flute). Pitch sonification was used in both cases. In the two questions, the subjects were asked about the trend in the attributes represented by each musical instrument. An example of such dataset (included in the Mul2 folder of our online repository [25]) is shown in Fig. 19.10.

Task 4: This task was designed to test the user's ability to make decisions based on sonified information only. A broker can take buy/sell decisions by watching the changes in some stocks using a visual representation. However, since, as we explained earlier, our sense of hearing is superior to our sense of sight, we can take our decisions faster by listening to the data instead or in addition to watching it. In the first experiment, we checked if one stream of data can be interpreted correctly by hearing only, this time we test the ability to interpret two streams played simultaneously. The two real-valued time-series are represented by pitch sonification, each played by its own instrument (piano, flute). The subjects were asked one question on long-term investment in one of the stocks and two questions on the best time to buy/sell each stock.

Task 5: This task was designed to test the user's ability to detect similarity/dissimilarity of two musical sequences played simultaneously. This task is particularly relevant as a part of data mining process. Machine learning algorithms sometimes make mistakes in their similarity decisions because these algorithms are based on precise mathematical calculations. In contrast, people make decisions according to the "overall" situation. For example, when using a visual representation, people can determine the general trend, without calculating the exact differences between each pair of consecutive values. Moreover, looking at a graphical representation, people can identify similar sequences even if they are shifted with respect to each other. Our goal was to test, whether such kind of decisions can be taken using a musical representation instead. The subjects were asked five questions, each referring to a pair of real-valued sequences represented by pitch sonification played on two different instruments. The data came from the weather and the financial domains.

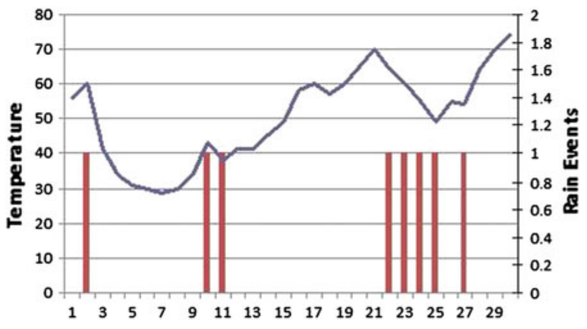


Fig. 19.8 Weather-April 2007 dataset

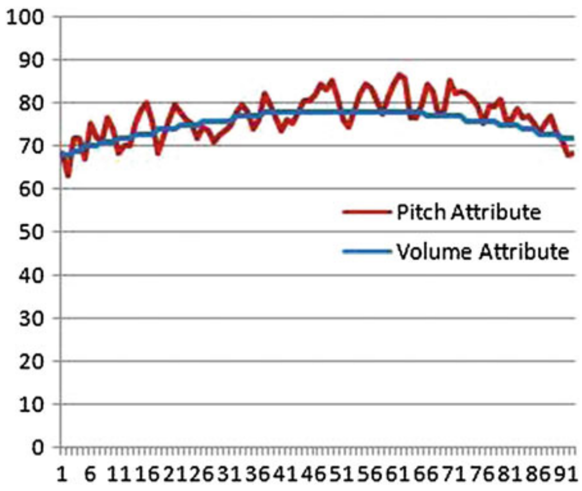


Fig. 19.9 Summer 03–07 dataset

In this experiment, like in the first one, the dependent variable was the subject's total score *S*, whereas the independent variables represented the demographic and musical characteristics of the subjects.

19.5.2 Hypotheses

Similar to the first experiment (see Sect. 19.4.2), we have tested the null hypothesis that the subjects are unable to provide correct answers or to take the right decisions by listening to the musical sequences generated by our sonification methodology. Thus,

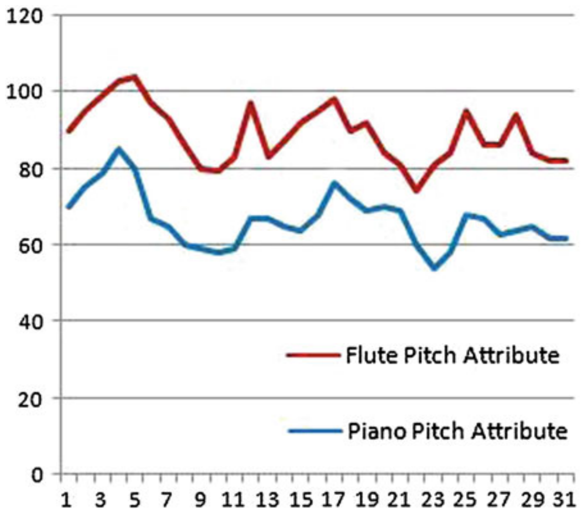


Fig. 19.10 Weather-July1978-1 dataset

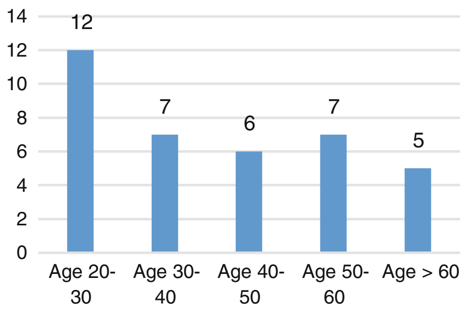


Fig. 19.11 Experiment 2: Age distribution

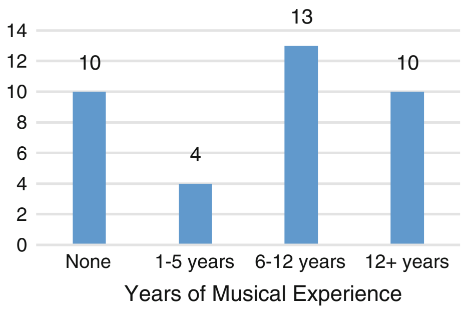


Fig. 19.12 Experiment 2: Musical experience distribution

we have compared the distribution of the subjects' answers/decisions to the uniform distribution. For tasks where the null hypothesis was rejected, we have explored the effect of the following subject's characteristics on his/her performance in the test: age, gender, occupation, musical experience, and musical hearing ability. The test performance measures were identical to the ones defined in Sect. 19.4.2.

19.5.3 Subjects

A total of 37 subjects took part in the second experiment (only those who answered all the questions). All data about the subjects was taken from their answers to the first part of the questionnaire (user information) only. According to above, there were 21 male and 16 female subjects between the ages of 20 and 70. The age distribution of the subjects is shown in Fig. 19.11. No one had any familiarity with sonification techniques (excluding possible participation in the first experiment, half a year earlier) and very few had experience with time series databases or data mining techniques. The distribution of the subjects' musical experience (if any) is shown in Fig. 19.12. Each subject could participate in the test at his/her time of convenience, without any timing constraints at any stage of the experiment.

Table 19.4 Number of correct answers per-task

	Number of answers	Number of correct answers	Percentage of correct answers (%)
Task 1	148	137	92.57
Task 2	148	125	84.46
Task 3	148	106	71.62
Task 4	111	82	73.87
Task 5	185	148	80.00

19.5.4 Experimental Results

The experiment was designed in the increasing order of difficulty. We started from relatively simple sonification techniques and tasks (the first and the second task) and up to the last task (no. 5), which was a relatively complex one. For all questions in every task, the percentage of correct answers has been significantly higher than the percentage that would have been obtained by chance, at the 99.9 % significance level. As shown in Table 19.4, the best result was obtained for the first task (92.57 %), whereas the worst one was for the third task (71.62 %). In the first four tasks, the subjects were required to distinguish between two sequences and to analyze each one of them separately. Only in the fifth task, they had to listen to the two sequences simultaneously. It is noteworthy that for this, more complex task, the percentage of correct answers was still relatively high—80 %.

Having verified that our sonification algorithm is a useful method for decision making and time series databases examination with ability to perform some popular data mining tasks, we have proceeded with the second part of our experiment aimed at finding the personal characteristics/abilities that affect the subjects performance. We have evaluated the same attributes as in the first experiment, namely Age, Gender, Occupation, Musical Experience, and Musical Pitch Ability. The attributes have been partitioned into the same categories except for the Musical Pitch Ability, which had the following five categories: mus_1—None, mus_2—Low musical pitch

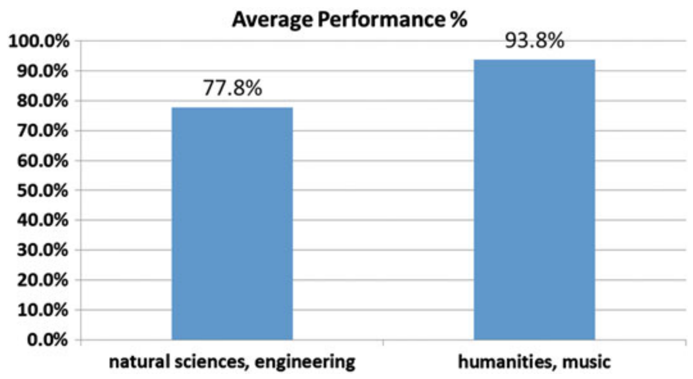


Fig. 19.13 Experiment 2: The effect of occupation groups

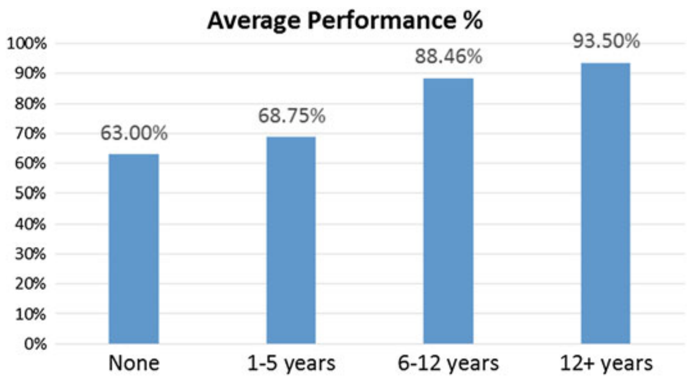


Fig. 19.14 Experiment 2: The effect of musical experience

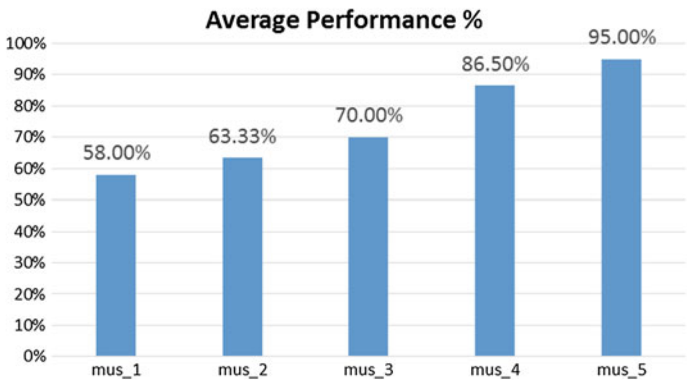


Fig. 19.15 Experiment 2: The effect of pitch ability

ability, mus_3—Average musical pitch ability, mus_4—Good musical pitch ability, mus_5—Excellent musical pitch ability.

According to the results of ANOVA test, there is no statistically significant difference in total scores across different age, gender, or occupation groups in general. However, we have found a statistically significant difference in total scores between the following two specific occupation groups: {natural sciences, engineering} versus {humanities, music} (see Fig. 19.13). Apparently, people with more technical background (engineers, physicists, etc.) find it more difficult to understand musified information. We have also found a statistically significant difference in total scores across different musical experience groups (contrary to the results of the first experiment) and different pitch ability categories. These results are demonstrated in Figs. 19.14 and 19.15, respectively. As expected, the subjects' performance is improving with the amount of their musical experience as well as with their pitch ability. On the other hand, even subjects without any musical experience or pitch ability can still perform much better than chance (58–63 % of correct answers versus 29 % with a random guess).

19.6 Conclusions

In this chapter, we have presented and evaluated a novel sonification methodology for representing information in time series databases so that humans can perform interactive data mining tasks without the need to view the actual data. Our algorithm can be used for sonification of univariate and multivariate time series. The algorithm can use three different sonification techniques: two for sonification of real-valued attributes and one for sonification of nominal attributes. The user can choose a specific sonification technique for each time series, according to the task performed.

Our generic sonification methodology can be used for representing time series from various domains, like weather forecasting, stock market, health care, process manufacturing, etc. It can assist people who are unable to utilize visual representations (physically or due to other simultaneously performed tasks), by providing them with the necessary tools to acquire, understand, and analyze time series data. The innovative features of our methodology include using a segmentation algorithm as a pre-sonification step and representing time series data on the Western musical scale.

The empirical evaluation of our technique included two online user studies. The first user study was designed to evaluate the basic ability of the subjects to use our sonification technique for performing some basic data mining tasks on a univariate time series. The second user study has evaluated some more complex tasks on bivariate time series. There were 44 subjects in the first experiment, and 37 subjects in the second experiment. The questionnaire of the second experiment has been prepared in the view of the lessons learned from the first experiment. Both experiments have shown that using our sonification algorithm for exploring univariate and bivariate time series provides very promising results, in terms of some important data mining tasks, like classification, clustering, and change detection. In the second experiment, we have also demonstrated the use of our algorithm for effective decision-making. Furthermore, we have discovered three factors that can determine the user's ability to successfully mine sonified data. These are (in decreasing order of importance): musical pitch ability, musical experience, and occupation. The average number of correct answers for both experiments was about 80 %.

Interested readers can listen to examples of sonified time series (including those used in our experiments) at [25].

In future experiments, one can study the effect of training on the users' performance as well as evaluate the user ability to perform various data exploration tasks on more than two simultaneously played time series. Future research may also include developing an "online" version of the proposed sonification algorithm for interactive mining of continuous, nonstationary data streams.

References

1. Barras S, Kramer G (1999) Using sonification. Multimed Syst 7(1):23–31
2. Brockwell PJ, Davis RA (2002) Time series: theory and methods, 2nd edn. Springer, New York
3. El-Azm F (2005) Sonification and augmented data sets in binary classification. PhD Dissertation, Institute of Informatics and Mathematical Modeling at the Technical University of Denmark
4. Gaver WW (1994) Using and creating auditory icons. In: Kramer G (ed) Auditory display: sonification, audification, and auditory interfaces. Addison-Wesley, Reading, pp 417–446
5. Hermann T (2008) Taxonomy and definitions for sonification and auditory display. In: Susini P, Warusfel O (eds) Proceedings of the 14th international conference on auditory display (ICAD 2008). IRCAM, Paris
6. Hermann T, Hunt A (2005) An introduction to interactive sonification. IEEE Multimed 12(2):20–24
7. Hermann T, Ritter H (1999) Listen to your data: Model-based sonification for data analysis. In: Proceedings of the international symposium on intelligent multimedia and distance education (ISIMADE'99), Baden-Baden
8. Keogh E, Kasetty S (2003) On the need for time series data mining benchmarks: a survey and empirical demonstration. Data Min Knowl Discov 7(4):349–371
9. Keogh E, Chu S, Hart D, Pazzani M (2004) Segmenting time series: a survey and novel approach. In: Last M, Kandel A, Bunke H (eds) Data mining in time series databases. World Scientific Publishing Company, Singapore, pp 1–21
10. Kramer G (1994) An introduction to auditory display. In: Kramer G (ed) Auditory display: sonification, audification and auditory interfaces, vol. XVIII. Addison-Wesley, pp 1–78
11. Kramer G, Walker B, Bonebright T, Cook P, Flowers J, Miner N, Neuhoff J (1999) Sonification report: status of the field and research agenda. Technical report, ICAD
12. Last M, Gorelik A (2008) Using sonification for mining time series data. In: Proceedings of the 9th international. Workshop on Multimedia Data Mining (MDM/KDD 2008), Las Vegas. 24 August 2008, pp 63–72
13. Last M, Kandel A, Bunke H (2004) Data mining in time series databases. In: Machine Perception and Artificial Intelligence, vol. 57. World Scientific, Singapore
14. Last M, Klein Y, Kandel A (2001) Knowledge discovery in time series databases. IEEE Trans Syst Man Cybern, Part B—Cybern, 31(1): 160–169
15. Leumi Group. http://www.bankleumi.co.il
16. Liu L-M, Bhattacharyya S, Sclove SL, Chena R, Lattyak WJ (2001) Data mining on time series: an illustration using fast-food restaurant franchise data. Comput Stat Data Anal 37(4):455–476
17. MSQ Project, http://www.aconnect.de/friends/editions/computer/msq2/msq.html
18. Muller W, Schumann H (2003) Visualization methods for time-dependent data—an overview. In: Proceedings of the 2003 winter simulation conference, vol 1, pp 737–745
19. Nesbitt KV, Barrass S (2004) Finding trading patterns in stock market data. IEEE Comput Graph Appl 24(5):45–55
20. Noirhomme-Fraiture M, Schöller O, Demoulin C, Simoff S (2002) Sonification of time dependent data. In: Proceedings of international workshop on visual data mining. Helsinki, pp 113–125
21. Patterson, R (1982) Guidelines for auditory warning systems on civil aircraft. Civil aviation authority
22. Pauletto S, Hunt A (2009) Interactive sonification of complex data. Int J Hum-Comput Stud 67(11):923–933
23. Pauletto S, Hunt A (2004) A toolkit for interactive sonification. In: Proceedings of the international conference of auditory display (ICAD). Sydney
24. Peretz I, Zatorre RJ (2005) Brain organization for music processing. Annu Rev Psychol 56:89–114
25. Sonification examples, http://www.ise.bgu.ac.il/faculty/mlast/data/MidiandPics.zip

26. Tel-Aviv Stock Exchange, http://www.tase.co.il
27. UCR Time series classification/clustering page, http://www.cs.ucr.edu/~eamonn/time_series_data/
28. Walker BN, Godfrey MT, Orlosky JE, Bruce C, Sanford J (2006) Aquarium sonification: soundscapes for accessible dynamic informal learning environments. In: Proceedings of the international conference on auditory display (ICAD2006). London, pp 238–241
29. Walker BN, Kim J, Pendse A (2007) Musical soundscapes for an accessible aquarium: Bringing dynamic exhibits to the visually impaired. In: Proceedings of the international computer music conference (ICMC 2007). Denmark
30. Walker BN, Lindsay J, Nance A, Nakano Y, Palladino DK, Dingler T, Jeon M (2013) Spearcons (speech-based earcons) improve navigation performance in advanced auditory menus. Hum Factors: J Hum Factors Ergon Soc 55(1):157–182
31. Williamson J, Murray-Smith, R (2002) Audio feedback with gesture recognition. Technical report TR-2002-127, Department of Computer Science, University of Glasgow

Author Index

A
Alcic, Sadet, 229
Amato, Christel, 147

B
Baluch, Farhan, 311
Baughman, Aaron K., 3, 177

C
Cataldi, Mario, 61
Chandramouli, Krishna, 295
Chen, Lei, 327
Chen, Zhongxue, 327
Choi, Jaeyoung, 157
Claudio Schifanella, 61
Conrad, Stefan, 229

D
Di Caro, Luigi, 61
Dojchinovski, Milan, 295

F
Ferrer, Luciana, 206
Ferré, Wilfredo, 147
Friedland, Gerald, 157

G
Galanopoulos, Damianos, 295
Gan, Rui, 397
Gao, Jiang, 3
Gong, Yunchao, 347
(Gorelik), Anna Usyskin, 419
Grzegorzek, Marcin, 269
Guo, Baining, 397

I
Itti, Laurent, 311

J
Janin, Adam, 157

K
Ke, Qifa, 373
Kliegr, Tomáš, 295
Kumabuchi, Kenji, 269
Kumar, Sanjiv, 347

L
Last, Mark, 419
Lawson, Aaron, 205
Lei, Howard, 157
Li, Shipeng, 373, 397
Liu, Qingzhong, 327
Luigi Di Caro, 61
Luo, Jiebo, 31

M
Mario Cataldi, 61
Mezaris, Vasileios, 295
Murray, John, 206

P
Pan, Jia-Yu, 3

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1

Peng, Wen-Chih, 93
Perlitz, Michael, 177
Petrushin, Valery A., 3

S
Schifanella, Claudio, 61
Shirahama, Kimiaki, 269
Sommer, Robin, 157
Sung, Andrew H., 327

T
Tang, Suhua, 117

U
Uehara, Kuniaki, 269

V
van der Stockt, Stefan, 177

W
Wang, Haoran, 253
Wang, Jing, 373, 397
Wang, Jingdong, 373, 397
Wang, Wen, 206
Wei, Ling-Yin, 93

X
Xiao, Changcheng, 253

Y
You, Quanzeng, 31
Yuan, Jianbo, 31
Yu, Felix X., 347
Yu, Yi, 117
Yvon, Marc, 147

Z
Zeng, Gang, 373, 397
Zhang, Liqing, 253
Zheng, Yu, 93
Zhou, Zhengzhong, 253
Zimmermann, Roger, 117

Subject Index

0–9symbols
3D recordings, 161

A
Account linking, 157
Active authentication, 206, 218
Aging population, 148, 149
Alternating optimization, 352, 358, 362
Analytics, 148, 152
Anonymity, 161, 164
Approximate nearest neighbor search, 398
AQBC, 363
Architectures, 202
Attack vectors, 157
Audification, 421
Auditory display, 420
Auditory interface, 420
Auto Color Correlogram, 166
Automatic speech recognition, 297

B
Big data
  DeepQA, 17
  Map Reduce, 17
  Meets fast analytics, 16
  Stream computing, 17
Bilinear Binary Embedding (BBE), 347, 349–351, 354, 364, 365, 367, 370
Bilinear projection, 347, 349, 363, 367
Binary embedding, 347, 351, 354, 356, 358
Biometric identification, 179, 202
Biometrics, 219
Bottom up attention, 320
Burstiness
  concept of, 65
  definition of, 71
  mathematical definition of, 73

C
Calibration, 206, 219
Cartesian concatenation, 404, 415
Circulant Binary Embedding (CBE), 347, 349, 354–358, 363, 364, 370
Circulant matrix, 347, 354, 355, 357
Circulant projection, 349, 357
Circular convolution, 355
Clothing recommendations, 253, 266
Cognitive computing, 18, 19
Collaborative learning, 93, 98
Color and Edge Directivity Descriptor, 166
Computer vision, 313
  limitations of, 32
Concept, 271, 272, 277, 281, 282
Concept-based video retrieval, 269, 271, 272, 291
Concept detection, 273, 274, 276, 277, 280, 282, 287, 289, 290, 291
Concept learning
  based on visual features, 32
Connected objects, 147, 148
Content-based video retrieval, 270, 271
Consumer-produced videos, 164
Content management system, 238
Content personalization
  Online Topic Model, 66
Copy-paste, 329, 338, 339
Correlation, 153
Creative Commons, 164
Creativity
  yes and, 4
Cropping, 328, 332, 341

A.K. Baughman et al. (eds.), *Multimedia Data Mining and Analytics*,
DOI 10.1007/978-3-319-14998-1

Cybercasing, 158

D
Daily routine, 153
Data acquisition, 148, 149
Data Law, 10, 11, 13
  definition of, 10
  multimedia, 10
  nanotechnology, 10
  National Institutes of Standards and Technology, 12
  principles of, 11
  relation to Moore's Law, 12–13
Demographics, 206, 221
Dempster–Shafer theory (DST), 278
Density ratio, 279, 284, 286, 290
Density ratio estimation, 279, 280, 284, 286
Digitized world
  3D models, 6
  Google Maps, 6
Dimensionality reduction, 354
Discrete Fourier Transformation (DFT), 355, 359
Disney
  imagine, innovate, impact, 5
  imagineers, 4
Disruptive innovation, 3–5, 7, 18
  definition of S-curves, 5
  relation to multimedia data media, 4
Document Object Model (DOM), 233

E
Educational techniques (privacy), 157
Eigenface
  accuracy, 35
  eigenfaces, 39
  facial expressions, 32
  facial sentiment recognition, 35
Emerging terms
  automatic keyword selection, 74
  mathematical definition of, 74
  user driven, 73
  to emerging topics, 77
Emerging topics
  correlation vector, 78
  from emerging terms, 77
  mathematical definition of, 78
  relationship to tweets, 78
  users' interests, 66
Environmental acoustic noise, 161
Evaluation
  automatic ranking, 85
  history, 84
  user selected threshold, 85
  worthiness, 83
Evaluation framework, 229, 231, 234–236, 247
Evaluation metrics, 230, 233, 247
Event identification
  social networks, 65
EXIF, 171
Exotic sensors, 161
Eye movements, 312, 313, 315, 317, 324

F
Facebook, 31, 62
Facial detection
  fdlibmex, 41
Factor analysis, 167
Fast Fourier Transformation (FFT), 349, 355, 363, 368
Fisher Vector, 348, 350
Flickr, 31, 64, 160, 161, 164, 167
Forgery detection, 328
Fuzzy Color and Texture Histogram, 166

G
Gabor, 166
Gaussian Mixture Model, 166, 167
Gaze, 312
Genetic algorithm, 322
Geographic popularity, 138
Geo-location, 159
Geo-social behaviors, 118, 144
Geo-tagged multimedia, 118, 119
GPU, 363
Ground truth, 229, 233, 235, 237, 247

H
Hamming distance, 348, 356–358
Hashing, 348, 357
Health care, 149
Hierarchical orientation features, 255, 257
Hit list, 158
Human computer interfaces
  relation to multimedia, 6
Human emotions
  anger, 37
  disgust, 37
  fear, 37
  happiness, 37
  sadness, 37
  surprise, 37

Human mood, 33
Human visual system, 311

I
Identity, 158, 165
image filter, 235
Image forensics, 327
Image search, 256, 262, 266
Image sentiment classification
  asymmetric bagging, 39
  bag of words, 43
  decision fusion, 43
  eigenvectors, 41
  logistic regression, 42
  performance of, 42
  support vector machine (SVM), 42
  textual content, 42
Imbalanced problem, 284, 288, 290
Immersive environments
  Kinect, 8
Inference, 157, 159, 170
Inference chain, 163
Information leakage, 159
Innovation
  Moore's Law, 10
  S-curves, 5
Interactive data exploration, 420
Internet
  Advanced Research Projects Agency Network (ARPANET), 7
  Defense Advanced Research Projects Agency (DARPA), 7
  Hypertext Transfer Protocol (HTTP), 7
  World Wide Web (WWW), 7
Inverse Discrete Fourier Transformation (IDFT), 355, 356, 359
ITQ, 363, 366

J
JPEG, 327–329, 332

K
Karalinska Directed Emotional Faces (KFEF), 37, 40
K-Nearest Neighbor-based Undersampling (kNNU), 284
Kronecker product, 351

L
Labeling, 79
Language, 206, 210, 212, 213, 219, 221
Large scale computing
  cloud computing, 15
  High Performance Computing (HPC), 15
Large-Scale Concept Ontology for Multimedia(LSCOM), 273
Large-scale facial identification, 188, 201, 202
Large-scale fingerprint identification, 180
LibSVM, 334, 335, 339
Locality sensitive Hashing (LSH), 357, 364–366
Location-aware preference mining, 134, 144
Location estimation, 157, 160, 171
Location recommendations, 118
Log-likelihood ratio, 166
Logistic regression, 336, 337, 340
Longest common subsequence, 246

M
Machine learning, 207, 210
Mass function, 278, 279, 282
Massive Open Online Course
  multimedia in society, 7
MFCC features, 166, 167
Microblogging, 62
Misaligned recompression, 329, 332, 341
Mitigation (privacy attack), 157, 158, 161, 169, 171
Mobile computing
  devices, 10
  social media, 10
Modeling attention, 313
Monash extractor, 232, 233, 242, 250
Moore's Law, 9, 10, 13
  definition, 9–10
  mechanical computing, 9
  relation to data law, 12–13
  semiconductors, 10
MQTT, 150
Multimedia, 147, 148, 150, 153, 154
  conferences, 19
  dataset examples, 12
Multimedia analytics, 157, 161
Multimedia content diffusion, 142, 144
Multimedia Data Mining, 3, 4, 5, 7, 13, 21
Multimodal fusion, 300
Musical sonification, 420

N
Natural language processing, 205
Neighborhood graph search, 400, 402, 409

Neuromorphic, 312
N-Terms Window, 240

O
Object detection, 161
Online Topic Model (OLDA)
  content personalization, 66
  definition of, 67
Organic computing
  artificial life, 14
  DNA computing, 15
  evolutionary computing, 14
  principles of, 13
  von Neumann architectures, 14
Overview of the book
  chapters, 4–26

P
Paragraph extractor, 241, 242
Parseval's theorem, 359
Pattern, 153
Personalization
  mathematical definition of, 76
  relationship to burstiness, 77
  users' context, 74
Person detection, 161
Plausibility, 279, 280, 284, 286, 288
Plausibility function, 279, 280, 283, 285–288, 290, 291
Predicting gaze, 317
Principal Component Analysis, 166
Privacy, 157–161, 164, 169–171
Probabilistic Linear Discriminant Analysis, 166
Product quantization, 363
Prosody, 214, 216, 219, 223

Q
Quality, 149, 152
Quantization, 327, 328, 331, 334, 341

R
Ranking
  discussion, 82
  mathematical definition of, 80
Real-time, 147, 148, 152
Real-time Vectorization
  tweet vector, 67
  tweets, 67
Recommendation
  aggregation, 63
  social networks, 63
Region descriptor, 275
Region detector, 275
Reputation
  definition of, 72
Risks, 157, 158, 170
Route Inference, 93

S
S-curves, 5, 7, 12
  definition of, 5
Salience, 313, 317
Saliency map, 314
Security, 157, 164, 170
Segmentation, 424–426, 428, 444
Semantic gap, 230, 270, 271
Semantic learning
  based on visual features, 32
Semi-supervised learning
  example of, 33
Sensor detection, 161
Sensors, 147, 148, 150–152
Sentiment analysis
  applications of, 32
  correlation between images and text, 51
  description of, 31
  multimedia, 44, 46, 54, 56
  number of images, 47
  overview of, 35
  sentiment prediction, 35
  short tweets, 44, 47, 52
  tweets, 46, 48
Sentiment analysis multimedia
  correlation between images and text, 54
  low level features, 34
  short tweets, 52
  Twitter API, 49
  visual sentiment classification, 47
Sentiment ontology
  Sentribute, 35
Sentribute
  definition of, 36
  eigenfaces, 39
  facial sentiment recognition, 39
  feature selection, 36
  framework, 35
  image sentiment classification, 33
  liblinear, 35, 37
  low-level features, 35
  mid-level attribute, 36
  sentiment analysis, 36

sentiment detection results, 45
sentiment ontology, 36
SUN Database, 36
SH, 363
Shape Topic Model, 256
Shift-recompression, 327, 328, 331, 332
SKLSH, 363
Smart home, 148
Social interactions, 134
Social media, 114
mobile devices, 8
Social networking, 157, 162, 171
Social networks
applications, 32
description of, 19
event identification, 65
semi-supervised learning, 33
Social platform
Facebook, 31
Flickr, 31
Twitter, 32
Social relations
graphs, 33
Sociolinguistics, 207, 223
Spatial Bag of Features, 256, 261
Speaker recognition, 161, 164, 212, 213, 215, 217
Speech, 206, 212–214, 218
Statistics, 148, 153
Surveillance, 154

T
Tags, 164
Tamura, 166
Temporal Stream Processing
burstiness, 68
follower, 69
graph model, 70
nutrition, 69
quality, 69
reputation of users, 70
vitality, 68
Text analysis, 296
Text similarity
TF-IDF, 65
Textual description, 164
Textual sentiment analysis, 47
Thinking machines
Antikythera, 18
Deep Blue, 18
ENIAC, 18
evolution of, 18–19
Napier's rods, 18
System 360, 18
Watson, 18
Time estimation, 160
Time series, 419, 420, 423, 427, 433
Topic detection
discussion, 82
user study, 82
Topic graph, 79
Trajectory data mining, 93
TRECVID, 276, 281, 285
TRECVID Search task, 285, 288
TRECVID Semantic Indexing task (SIN), 277
Trend analysis
Trendistic, 64
Twopular, 64
Trend discovery, 420
Tumblr, 61
Twitter
data sets, 33
definition of, 69
microblogging, 62

U
Uncertain concept detection, 280, 287, 289
Uncertain Data, 290
Uncertainty, 278, 282, 284, 290
Unconstrained Least-Squares Importance Fitting, 284
Undersampling, 284, 286, 291
User matching, 168
User study
evaluating personalization, 85
topic detection, 85

V
Vector of Locally Aggregated Descriptors (VLAD), 348, 350, 353
Video analysis, 296
VIPS, 232, 233, 244, 245, 249, 250
Virtual worlds, 207, 208
Visual attention, 313
Visual concept detection, 296, 305
Visual content
overview of, 35
topics of sentiment analysis, 31

W
Web document, 229, 230, 232–236, 238, 245
Web image, 230, 232, 233

Web image context, 229, 230, 233
Weibo
  microblogging, 32
Wellness, 148, 154
Western tonal music, 420
Within-Class Covariance Normalization, 166
Writing, 206, 221

Z
Zigbee, 151