Chapter 1

INTRODUCTION

Samir B. Unadkat, Mãlina M. Ciocoiu and Larry R. Medsker

Department of Computer Science and Information Systems
American University

I. OVERVIEW

Recurrent neural networks have been an important focus of research and development during the 1990's. They are designed to learn sequential or time-varying patterns. A recurrent net is a neural network with feedback (closed loop) connections [Fausett, 1994]. Examples include BAM, Hopfield, Boltzmann machine, and recurrent backpropagation nets [Hecht-Nielsen, 1990].

Recurrent neural network techniques have been applied to a wide variety of problems. Simple partially recurrent neural networks were introduced in the late 1980's by several researchers including Rumelhart, Hinton, and Williams [Rummelhart, 1986] to learn strings of characters. Many other applications have addressed problems involving dynamical systems with time sequences of events.

Table 1 gives some other interesting examples to give the idea of the breadth of recent applications of recurrent neural networks. For example, the dynamics of tracking the human head for virtual reality systems is being investigated. The

Table 1. Examples of recurrent neural network applications.

Topic	Authors	Reference
Predictive head tracking for virtual reality systems	Saad, Caudell, and Wunsch, II	[Saad, 1999]
Wind turbine power estimation	Li, Wunsch, O'Hair, and Giesselmann	[Li, 1999]
Financial prediction using recurrent neural networks	Giles, Lawrence, Tsoi	[Giles, 1997]
Music synthesis method for Chinese plucked-string instruments	Liang, Su, and Lin	[Liang, 1999]
Electric load forecasting	Costa, Pasero, Piglione, and Radasanu	[Costa, 1999]
Natural water inflows forecasting	Coulibaly, Anctil, and Rousselle	[Coulibaly, 1999]

forecasting of financial data and of electric power demand are the objects of other studies. Recurrent neural networks are being used to track water quality and minimize the additives needed for filtering water. And, the time sequences of musical notes have been studied with recurrent neural networks.

Some chapters in this book focus on systems for language processing. Others look at real-time systems, trajectory problems, and robotic behavior. Optimization and neuro-fuzzy systems are presented, and recurrent neural network implementations of filtering and control are described. Finally, the application of recurrent neural networks to chaotic systems is explored.

A. RECURRENT NEURAL NET ARCHITECTURES

The architectures range from fully interconnected (Figure 1) to partially connected nets (Figure 2), including multilayer feedforward networks with distinct input and output layers. Fully connected networks do not have distinct input layers of nodes, and each node has input from all other nodes. Feedback to the node itself is possible.

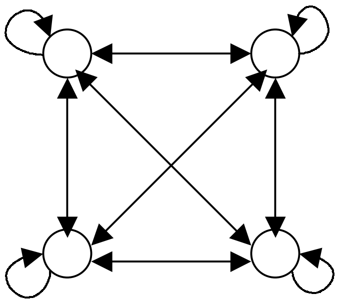

Figure 1. An example of a fully connected recurrent neural network.

Simple partially recurrent neural networks (Figure 2) have been used to learn strings of characters. Athough some nodes are part of a feedforward structure,

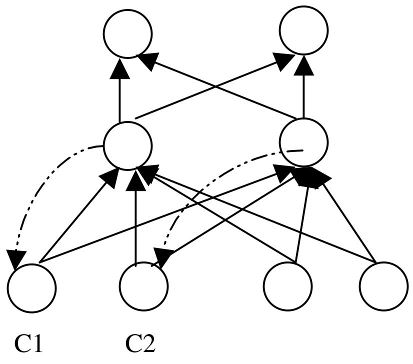

Figure 2. An example of a simple recurrent network.

other nodes provide the sequential context and receive feedback from other nodes. Weights from the context units (C1 and C2) are processed like those for the input units, for example, using backpropagation. The context units receive time-delayed feedback from, in the case of Figure 2, the second layer units. Training data consists of inputs and their desired successor outputs. The net can be trained to predict the next letter in a string of characters and to validate a string of characters.

Two fundamental ways can be used to add feedback into feedforward multilayer neural networks. Elman [Elman, 1990] introduced feedback from the hidden layer to the context portion of the input layer. This approach pays more attention to the sequence of input values. Jordan recurrent neural networks [Jordan, 1989] use feedback from the output layer to the context nodes of the input layer and give more emphasis to the sequence of output values. This book covers a range of variations on these fundamental concepts, presenting ideas for more efficient and effective recurrent neural networks designs and examples of interesting applications.

B. LEARNING IN RECURRENT NEURAL NETS

Learning is a fundamental aspect of neural networks and a major feature that makes the neural approach so attractive for applications that have from the beginning been an elusive goal for artificial intelligence. Learning algorithms have long been a focus of research (e.g., Nilsson [1965] and Mendel [1970]).

Hebbian learning and gradient descent learning are key concepts upon which neural network techniques have been based. A popular manifestation of gradient descent is back-error propagation introduced by Rumelhart [1986] and Werbos [1993]. While backpropagation is relatively simple to implement, several problems can occur in its use in practical applications, including the difficulty of avoiding entrapment in local minima. The added complexity of the dynamical processing in recurrent neural networks from the time-delayed updating of the input data requires more complex algorithms for representing the learning.

To realize the advantage of the dynamical processing of recurrent neural networks, one approach is to build on the effectiveness of feedforward networks that process stationary patterns. Researchers have developed a variety of schemes by which gradient methods, and in particular backpropagation learning, can be extended to recurrent neural networks. Werbos introduced the backpropagation through time approach [Werbos, 1990], approximating the time evolution of a recurrent neural network as a sequence of static networks using gradient methods. Another approach deploys a second, master, neural network to perform the required computations in programming the attractors of the original dynamical slave network [Lapedes and Farber, 1986]. Other techniques that have been investigated can be found in Pineda [1987], Almeida [1987], Williams and Zipser [1989], Sato [1990], and Pearlmutter [1989]. The various attempts to extend backpropagation learning to recurrent networks is summarized in Pearlmutter [1995].

II. DESIGN ISSUES AND THEORY

The first section of the book concentrates on ideas for alternate designs and advances in theoretical aspects of recurrent neural networks. The authors discuss aspects of improving recurrent neural network performance and connections with Bayesian analysis and knowledge representation.

A. OPTIMIZATION

Real-time solutions of optimization problems are often needed in scientific and engineering problems, including signal processing, system identification, filter design, function approximation, and regression analysis, and neural networks have been widely investigated for this purpose. The numbers of decision variables and constraints are usually very large, and large-scale optimization procedures are even more challenging when they have to be done in real time to optimize the performance of a dynamical system. For such applications, classical optimization techniques may not be adequate due to the problem dimensionality and stringent requirements on computational time. The neural network approach can solve optimization problems in running times orders of magnitude faster than the most popular optimization algorithms executed on general-purpose digital computers.

The chapter by Xia and Wang describes the use of neural networks for these problems and introduces a unified method for designing optimization neural network models with global convergence. They discuss continuous-time recurrent neural networks for solving linear and quadratic programming and for solving linear complementary problems and then focus on discrete-time neural networks. Assignment neural networks are discussed in detail, and some simulation examples are presented to demonstrate the operating characteristics of the neural networks.

The chapter first presents primal-dual neural networks for solving linear and quadratic programming problems (LP and QP) and develops the neural network for solving linear complementary problems (LCP). Following a unified method for designing neural network models, the first part of the chapter describes in detail primal-dual recurrent neural networks, with continuous time, for solving LP and QP. The second part of the chapter focuses on primal-dual discrete time neural networks for QP and LCP.

Although great progress has been made in using neural networks for optimization, many theoretical and practical problems remain unsolved. This chapter identifies areas for future research on the dynamics of recurrent neural networks for optimization problems, further application of recurrent neural networks to practical problems, and the hardware prototyping of recurrent neural networks for optimization.

B. DISCRETE-TIME SYSTEMS

Santos and Von Zuben discuss the practical requirement for efficient supervised learning algorithms, based on optimization procedures for adjusting the parameters. To improve performance, second order information is considered to minimize the error in the training. The first objective of their work is to describe systematic ways of obtaining exact second-order information for a range of recurrent neural network configurations, with a computational cost only two times higher than the cost to acquire first-order information. The second objective is to present an improved version of the conjugate gradient algorithm that can be used to effectively explore the available second-order information.

The dynamics of a recurrent neural network can be continuous or discrete in time. However, the simulation of a continuous-time recurrent neural network in digital computational devices requires the adoption of a discrete-time equivalent model. In their chapter, they discuss discrete-time recurrent neural network architectures, implemented by the use of one-step delay operators in the feedback paths. In doing so, digital filters of a desired order can be used to design the network by the appropriate definition of connections. The resulting nonlinear models for spatio-temporal representation can be directly simulated on a digital computer by means of a system of nonlinear difference equations. The nature of the equations depends on the kind of recurrent architecture adopted but may lead to very complex behaviors, even with a reduced number of parameters and associated equations.

Analysis and synthesis of recurrent neural networks of practical importance is a very demanding task, and second-order information should be considered in the training process. They present a low-cost procedure to obtain exact second-order information for a wide range of recurrent neural network architectures. They also present a very efficient and generic learning algorithm, an improved version of a scaled conjugate gradient algorithm, that can effectively be used to explore the available second-order information. They introduce a set of adaptive coefficients in replacement to fixed ones, and the new parameters of the algorithm are automatically adjusted. They show and interpret some simulation results.

The innovative aspects of this work are the proposition of a systematic procedure to obtain exact second-order information for a range of different recurrent neural network architectures, at a low computational cost, and an improved version of a scaled conjugate gradient algorithm to make use of this high-quality information. An important aspect is that, given the exact second-order information, the learning algorithm can be directly applied, without any kind of adaptation to the specific context.

C. BAYESIAN BELIEF REVISION

The Hopfield neural network has been used for a large number of optimization problems, ranging from object recognition to graph planarization to concentrator assignment. However, the fact that the Hopfield energy function is of quadratic order limits the problems to which it can be applied. Sometimes, objective functions that cannot be reduced to Hopfield's quadratic energy function can still be reasonably approximated by a quadratic energy function. For other problems, the objective function must be modeled by a higher-order energy function. Examples of such problems include the angular-metric TSP and belief revision, which is Abdelbar's subject in Chapter 4.

In his chapter, Abdelbar describes high-order recurrent neural networks and provides an efficient implementation data structure for sparse high-order networks. He also describes how such networks can be used for Bayesian belief revision and in important problems in diagnostic reasoning and commonsense reasoning under uncertainty.

D. KNOWLEDGE REPRESENTATION

Giles, Omlin, and Thornber discuss in their chapter neuro-fuzzy systems -- the combination of artificial neural networks with fuzzy logic -- which have become useful in many application domains. They explain, however, that conventional neuro-fuzzy models usually need enhanced representational power for applications that require context and state (e.g., speech, time series prediction, and control). Some of these applications can be readily modeled as finite state automata. Previously, it was proved that deterministic finite state automata (DFA) can be synthesized by or mapped into recurrent neural networks by directly programming the DFA structure into the weights of the neural network. Based on those results, they propose a synthesis method for mapping fuzzy finite state automata (FFA) into recurrent neural networks. This mapping is suitable for direct implementation in VLSI, i.e., the encoding of FFA as a generalization of the encoding of DFA in VLSI systems.

The synthesis method requires FFA to undergo a transformation prior to being mapped into recurrent networks. The neurons are provided with an enriched functionality in order to accommodate a fuzzy representation of FFA states. This enriched neuron functionality also permits fuzzy parameters of FFA to be directly represented as parameters of the neural network.

They also prove the stability of fuzzy finite state dynamics of the constructed neural networks for finite values of network weight and, through simulations, give empirical validation of the proofs. This proves the various knowledge equivalence representations between neural and fuzzy systems and models of automata.

E. LONG-TERM DEPENDENCIES

Gradient-descent learning algorithms for recurrent neural networks are known to perform poorly on tasks that involve long-term dependencies, i.e., those problems for which the desired output depends on inputs presented at times far in the past. Lin, Horne, Tino, and Giles discuss this in their chapter and show that the long-term dependencies problem is lessened for a class of architectures called NARX recurrent neural networks, which have powerful representational capabilities.

They have previously reported that gradient-descent learning can be more effective in NARX networks than in recurrent neural networks that have "hidden states" on problems including grammatical inference and nonlinear system identification. Typically the network converges much faster and generalizes better than other networks, and this chapter shows the same kinds of results.

They also present in this chapter some experimental results that show that NARX networks can often retain information for two to three times as long as conventional recurrent neural networks. They show that although NARX networks do not circumvent the problem of long-term dependenices, they can greatly improve performance on long-term dependency problems. They describe in detail some of the assumptions regarding what it means to latch information robustly and suggest possible ways to loosen these assumptions.

III. APPLICATIONS

This section looks at interesting modifications and applications of recurrent neural networks. Problems dealing with trajectories, control systems, robotics, and language learning are included, along with an interesting use of recurrent neural networks in chaotic systems.

A. CHAOTIC RECURRENT NETWORKS

Dayhoff, Palmadesso, and Richards present in their chapter work on the use of recurrent neural networks for chaotic systems. Dynamic neural networks are capable of a tremendous variety of oscillations, such as finite state oscillations, limit cycles, and chaotic behavior. The differing oscillations that are possible create an enormous repertoire of self-sustained activity patterns. This repertoire is very interesting because oscillations and changing activity patterns can potentially be exploited for computational purposes and for modeling physical phenomena.

In this chapter, they explore trends observed in a chaotic network when an external pattern is used as a stimulus. The pattern stimulus is a constant external input to all neurons in a single-layer recurrent network. The strength of the stimulus is varied to produce changes and trends in the complexity of the evoked oscillations. Stronger stimuli can evoke simpler and less varied oscillations. Resilience to noise occurs when noisy stimuli evoke the same or similar oscillations. Stronger stimuli can be more resilient to noise. They show examples of each of these observations. A pattern-to-oscillation map may eventually be exploited for pattern recognition and other computational purposes. In such a paradigm, the external pattern stimulus evokes an oscillation that is read off the network as the answer to a pattern association problem. They present evidence that this type of computational paradigm has higher potential for pattern capacity and boundary flexibility than a multilayer static feedforward network.

B. LANGUAGE LEARNING

The Kremer chapter examines the relationship between grammar induction or language learning and recurrent neural networks, asking how understanding formal language learning can help in designing and applying recurrent neural networks. The answer to this question comes in the form of four lessons: (1) training RNNs is difficult, (2) reducing the search space can accelerate or make learning possible, (3) ordering the search space can speed learning, and (4) ordering your training data helps. The chapter concerns dynamical recurrent neural networks, those that are presented with time-varying inputs and are designed to render outputs at various points in time. In this case, the operation of the network can be described by a function mapping an input sequence to an output value or sequence of values and is applied to the problem where inputs are selected from a discrete alphabet of valid values and output values fall into discrete categories. The problem of dealing with input sequences in which each item is selected from an input alphabet can also be cast as a formal language problem. This work uses recurrent neural networks to categorize subsets of an input language and reveals effective techniques for language learning.

C. SEQUENTIAL AUTOASSOCIATION

In spite of the growing research on connectionist Natural Language Processing (NLP), a number of problems need to be solved such as the development of proper linguistic representations. Natural language is a dynamic system with underlying hierarchical structure and sequential external appearance and needs an adequate hierarchical systematic way of linguistic representation. The development of global-memory recurrent neural networks, such as the Jordan Recurrent Networks [Jordan, 1986] and the Simple Recurrent Networks (SRN) by Elman [1990] stimulated the development of models that gradually build representations of their sequential input in this global memory

Stoianov in his chapter presents a novel connectionist architecture designed to build and process a hierarchical system of static distributed representations of complex sequential data. It follows upon the idea of building complex static representations of the input sequence but has been extended to reproduce these static representations in their original form by building unique representations for every input sequence. The model consists of sequential autoassociative modules called Recurrent Autoassociative Networks (RANs). Each of these modules learns to reproduce input sequences and as a side effect, develops static distributed representations of the sequences. If requested, these modules unpack static representations into their original sequential form. The complete architecture for processing sequentially represented hierarchical input data consists of a cascade of RANs. The input tokens of a RAN module from any but the lowest level in this cascade scheme are the static representations that the RAN module from the lower level has produced. The input data of the lowest level RAN module are percepts from the external world. The output of a module from the lowest level can be associated with an effector. Then, given a static representation set to the RAN hidden layer, this effector would receive commands sequentially during the unpacking process.

RAN is a recurrent neural network that conforms to the dynamics of natural languages, and RANs produce representations of sequences and interpret them by unpacking back to their sequential form. The more extended architecture, a cascade of RANs, resembles the hierarchy in natural languages. Furthermore, given a representative training environment, this architecture has the capacity to develop the distributed representations in a systematic way. He argues that RANs provide an account of systematicity, and therefore that the RAN and the RAN cascade can participate in a more global cognitive model, where the distributed representations they produce are extensively transformed and associated.

This chapter includes a discussion of hierarchy in dynamic data, and a small RAN example is presented for developing representations of syllables. Although the model solves the problem of developing representations of hierarchically structured sequences, some questions remain open, especially for developing an autonomous cognitive model. Nevertheless, the suggested model may be an important step in connectionist modeling.

D. TRAJECTORY PROBLEMS

An important application of recurrent neural networks is the modeling of dynamic systems involving trajectories, which are good examples of events with specific required time relationships. Typical test cases are the famous nonlinear and autonomous dynamic systems of the circle and the figure-eight.

The difficulty in training recurrent networks often results in the use of approximations that may result in inefficient training. Sundareshan, Wong, and Condarcure in their chapter describe two alternate learning procedures that do not require gradient evaluations. They demonstrate the performance of the two algorithms by use of a complex spatiotemporal learning task to produce continuous trajectories. They show significant advantages in implementation.

They describe two distinct approaches. One uses concepts from the theory of learning automata and the other is based on the classical simplex optimization approach. They demonstrate the training efficiency of these approaches with the task of spatiotemporal signal production by a trained neural network. The complexity of this task reveals the unique capability of recurrent neural networks for approximating temporal dynamics.

In their chapter, Hagner, Hassoun, and Watta compare different network architectures and learning rules, including single-layer fully recurrent networks and multilayer networks with external recurrence: incremental gradient descent, conjugate gradient descent, and three versions of the extended Kalman filter. The circle trajectory is shown to be relatively easily learned while the figure-eight trajectory is difficult. They give a qualitative and quantitative analysis of the neural net approximations of these internally and externally recurrent autonomous systems.

E. FILTERING AND CONTROL

Recurrent networks are more powerful than nonrecurrent networks, particularly for uses in control and signal processing applications. The chapter by Hagan, De Jesús, and Schultz introduces Layered Digital Recurrent Networks (LDRN), develops a general training algorithm for this network, and demonstrates the application of the LDRN to problems in controls and signal processing. They present a notation necessary to represent the LDRN and discuss the dynamic backpropagaion algorithms that are required to compute training gradients for recurrent networks. The concepts underlying the backpropagation-through-time and forward perturbation algorithms are presented in a unified framework, and are demonstrated for a simple, single-loop recurrent network. They also describe a general forward perturbation algorithm for computing training gradients for the LDRN.

Two application sections discuss dynamic backpropogation: implementation of the general dynamic backpropogation algorithm and the application of a neurocontrol architecture to the automatic equalization of an acoustic transmitter. A section on nonlinear filtering demonstrates the application of a recurrent filtering network to a noise-cancellation application.

F. ADAPTIVE ROBOT BEHAVIOR

The chapter by Ziemke discusses the use of recurrent neural networks for robot control and learning and investigates its relevance to different fields of research, including cognitive science, AI, and the engineering of robot control systems. Second-order RNNs, which so far only rarely have been used in robots, are discussed in particular detail, and their capacities for the realization of adaptive robot behavior are demonstrated and analyzed experimentally.

IV. FUTURE DIRECTIONS

This book represents the breadth and depth of interest in recurrent neural networks and points to several directions for ongoing research. The chapters address both new and improved algorithms and design techniques and also new applications. The topics are relevant to language processing, chaotic and real-time systems, optimization, trajectory problems, filtering and control, and robotic behavior.

Research in recurrent neural networks has occurred primarily in the 1990's, building on important fundamental work in the late 1980's. The next decade should produce significant improvements in theory and design as well as many more applications for the creative solution of important practical problems. The widespread application of recurrent neural networks should foster more interest in research and development and raise further theoretical and design questions. The ongoing interest in hybrid systems should also result in new and more powerful uses of recurrent neural networks.

REFERENCES

Almeida, L. B., A learning rule for asynchronous perceptrons with feedback in a combinatorial environment, *Proceedings of the IEEE 1st Annual International Conference on Neural Networks*, San Diego, 609, 1987.

Costa, M., Pasero, E., Piglione, F, and Radasanu, D., Short term load forecasting using a synchronously operated recurrent neural network, *Proceedings of the International Joint Conference on Neural Networks*, 1999.

Coulibay, P., Anctil, F., and Rousselle, J., Real-time short-term water inflows forecasting using recurrent neural networks, *Proceedings of the International Joint Conference on Neural Networks*, 1999.

Elman, J. L., Finding structure in time, *Cognitive Science*, 14, 179, 1990.

Fausett, L., *Fundamentals of Neural Networks*, Prentice Hall, Englewood Cliffs, NJ, 1994.

Giles, C. L., Lawrence, S., Tsoi, A.-C., Rule inference for financial prediction using recurrent neural networks, *IEEE Conference on Computational Intelligence for Financial Engineering*, IEEE Press, 253, 1997.

Hecht-Nielsen, R., *Neurocomputing*, Addison-Wesley, Reading, PA, 1990.

Jordan, M., Generic constraints on underspecified target trajectories, *Proceedings of the International Joint Conference on Neural Networks*, I, 217, 1989.

Lapedes, A. and Farber, R., Programming a massively parallel computation universal system: static behavior, in *Neural Networks for Computing,* Denker, J. S., Ed., AIP Conference Proceedings, 151, 283, 1986.

Li, S., Wunsch II, D. C., O'Hair, E., and Giesselmann, M. G., Wind turbine power estimation by neural networks with Kalman filter training on a SIMD parallel machine, *Proceedings of the International Joint Conference on Neural Networks*, 1999.

Liang, S.-F., Su, A. W. Y., and Lin, C.-T., A new recurrent-network-based music synthesis method for Chinese plucked-string instruments - pipa and qiu, *Proceedings of the International Joint Conference on Neural Networks*, 1999.

Mendel, J. M. and Fu, K. S., Eds., *Adaptive, Learning and Pattern Recognition Systems,* Academic, New York, 1970.

Nilsson, N. J., *Learning Machines: Foundations of Trainable Pattern Classifying Systems*, McGraw-Hill, New York, 1965.

Pearlmutter, B., Learning state space trajectories in recurrent neural networks, *Neural Computation,* 1, 263, 1989.

Pearlmutter, B., Gradient calculations for dynamic recurrent neural networks: A survey, *IEEE Transactions on Neural Networks,* 6, 1212, 1995.

Pineda, F. J., Generalization of backpropagation in recurrent neural networks, *Physical Review Letters*, 59 (19), 2229, 1987.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J., Learning internal representations by error propagation, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Rumelhart, D. E. and McClelland, J. L., Eds., MIT Press, Cambridge, 45, 1986.

Saad, E. W., Caudell, T. P., and Wunsch II, D. C., Predictive head tracking for virtual reality, *Proceedings of the International Joint Conference on Neural Networks*, 1999.

Sato, M., A real time running algorithm for recurrent neural networks, *Biological Cybernetics,* 62, 237, 1990.

Werbos, P., Backpropagation through time: what it does and how to do it, *Proceedings of the IEEE*, 78, 1550, 1990.

Werbos, P., *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*, Wiley, New York, 1993.

Williams, R. and Zipser, D., A learning algorithm for continually running fully recurrent neural networks, *Neural Computation*, 1, 270, 1989.