CHAPTER 7

NEURAL NETWORK LEARNING IN A TRAVEL RESERVATION DOMAIN

H.A. Aboulenien and **P. De Wilde**
Department of Electrical and Electronic Engineering
Imperial College of Science, Technology and Medicine
London, U.K.
`{h.aboulenien, p.dewilde}@ic.ac.uk`

This chapter presents an intelligent agent that employs a machine learning technique in order to provide assistance to users dealing with a particular computer application. Machine learning is a sub-field of artificial intelligence (AI) that includes the automated acquisition of knowledge. The aim is intelligent systems that learn, that is, improve their performance as a result of experience. The agent learns how to assist the user by being trained by the user using hypothetical examples and receiving user feedback when it makes a wrong decision. The main aim is to reduce work for the end-user by building a software agent that acts as a personal assistant. The proposed interface agent will potentially have the features of natural language interface and learning through interaction. The achievement of these innovations is mainly based on neural network learning techniques. The chapter presents preliminary results from a prototype agent built using this technique and applied on flight reservation domain.

1 Introduction

One of the obvious difficulties for building intelligent machines has been for many years the passive nature of computers. Computers do only what they were programmed to do and do not learn to adapt to changing circumstances. At the same time, it will become more and more difficult for untrained computer users to cope with the increasing complexity of computer applications and the growth of the computers' direct-manipulation interfaces. One of the aims for building intelligent machines, and also one of the biggest challenges, is to create a simple user interface so that the human-computer interaction will become as natural for end-users as picking up a phone or reading a newspaper.

Artificial intelligence researchers and software companies have set high hopes on so-called *Software Agents* which learn users' interests and can act autonomously on their behalf to contribute in solving the learning problem [1]. The learning approach has several advantages. First, it requires less work from the end-user and application developers. Second, the agent can easily adapt to the user over time and become customised to individual and organisational preferences and habits. Despite the huge diversity in this rapidly evolving area of agents' research, the most promising one in solving the human-computer interaction problems is called *Interface Agents*. Interface agents can be characterised as systems, which employ artificial intelligence techniques to provide assistance to users with a particular computer application [2].

An important property that any interface agent should have is the ability to communicate with the human user via some kind of natural language. This is based on the belief that computers will not be able to perform many of the tasks people do every day until they, too, share the ability to use their language. Despite more than twenty years of research into natural language understanding, the solutions for actual problems such as a natural language computer interface still suffer from inadequate performance. This problem is due to the fact that these systems depend on exact knowledge of how human language works. However, even now there is no complete and formal description of human language available. It has been argued that there is some evidence that the human brain is specially structured for language [3]. However, today's computer architecture is totally different from the human brain.

In an attempt to model the human mind/brain, it has been necessary to oversimplify the structure and the function. This has led to the development of an important area of research, namely neural computing. This area belongs to a larger research paradigm known as computational intelligence which aims to model functions associated with intelligence, at the signal level as a dynamical system. Neural computing is the study of Artificial Neural Networks (ANNs).

In this chapter, the main aim is to introduce the interface part of the proposed agent that communicates with the user and learns through this interaction to be able to assist him/her in the travel reservation domain. The learning machine of this agent is based on neural network techniques. In the next section, we introduce a brief definition of agents. Section 3 presents the main features of neural networks. The rest of the chapter explains the implementation of the proposed interface agent and some of the simulation results.

2 Agents

Software agents have evolved from multi-agent systems, which in turn form one of three broad areas, which fall under distributed artificial intelligence (the other two being distributed problem solving and parallel artificial intelligence). Although the term agent is a widely used term in computing, AI and other related areas, it is poorly defined. Perhaps the most general way in which the term agent is used is to denote a hardware or (more usually) software-based computer system that enjoys the following: [4]

- ***Autonomy***: Agents operate without the direct intervention of humans or other agents and have some kind of control over their own actions and internal state.
- ***Social Ability***: Agents interact with other agents (and possibly humans) via some kind of agent-communication language.
- ***Reactivity***: Agents perceive their environment (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET or perhaps all of these combined) and respond in a timely fashion to changes that occur in it.
- ***Pro-activeness***: Agents do not simply act in response to their environment; they are able to exhibit goal-directed behaviour by taking the initiative.

There are sometimes other agent's attributes, which are considered secondary attributes to those mentioned above such as *mobility, continuity, robustness* and *rationality*. Our proposed interface agent aimed to enjoy most of the mentioned attributes except the mobility feature. Generally, the implementation of interface agents is focused on autonomy and learning. The interactivity (social ability) attribute will be gained through the architecture of integrating many agents in a collaborative scheme.

3 Neural Network Role

Our aim was to benefit from the features of artificial neural networks (ANNs), which mimic the biological nervous system to perform information processing and learning. On top of the superficial resemblance, ANNs exhibit a surprising number of human brain characteristics such as learning, generalisation and robustness. One of the most important features of ANNs is the ease with which they can learn (modify the behaviour in response to the environment). *Learning* in ANNs is the process of adjusting the connection weights between the nodes. Neural networks are often required to learn an input/output mapping from existing data or learn from input data only when the output is not known. In the last case, ANNs are capable of *abstracting* the essence of a set of input data, i.e., learning to produce something never seen before. ANNs perform the equivalent of inductive learning in the symbolic paradigm.

Once trained, a network's response can be, to a degree, insensitive to minor variations in its inputs. *Generalisation* in learning enables ANNs to learn from incomplete data. This ability to see through noise and distortion to the pattern that lies within is vital to pattern recognition in a real-world environment. Producing a system that can deal with the imperfect world in which we live overcomes the literal mindedness of the conventional computer. This attribute is mainly due to its structure, not by using human intelligence, embedded in the form of *ad hoc* computer programs. Computer programs play no role. *Parallelism* allows ANNs to model complex relations and to perform complex tasks at speeds in excess of other algorithms (this feature can only be fully exploited if their hardware implementation is used). The above features are the main contribution of ANNs to intelligent systems [5].

As a matter of fact, ANNs have proved to complement conventional symbolic artificial intelligent techniques in applications where the theory is poor and the data are rich, such as pattern recognition, pattern matching, and adaptive non-linear control. Some researchers claim that ANNs will replace current AI, but there are many indications that the two will co-exist and be combined into systems in which each technique performs the tasks for which it is suited [6].

4 Agent Architecture

In this section, we introduce a brief description of the proposed interface agent's building blocks. Figure 1 shows the interface agent architecture [7].

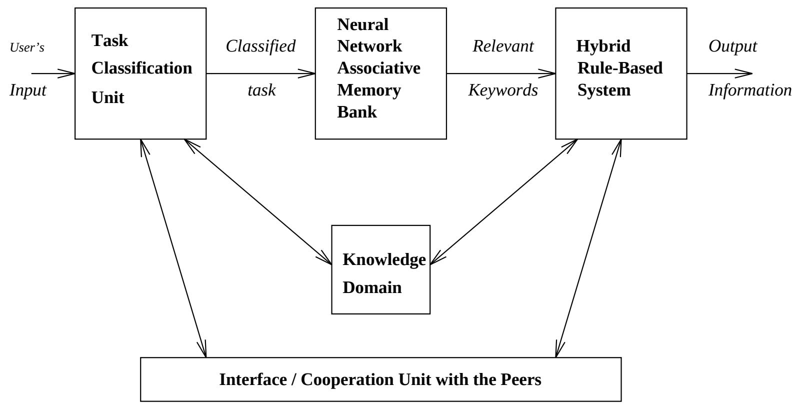

Figure 1. The interface agent architecture.

- **Task Classification Unit (TCU)**: This is the interface part where the user-system interactions are performed. The TCU accepts the input request as natural language sentences from the user. The sentences represent the required task that is required to deal with. TCU applies approximate string matching and neural network learning techniques to analyse the input sentences, then classifies the task according to a search process in its heuristic database system (concept maps).
- **Neural Network Associative Memory (NNAM)**: This part of the agent acts as a heteroassociative memory which is responsible for

generating keywords related to the classified task received from the TCU, though it works as a look-up table implemented using ANN technique.

- **Hybrid Rule-Based System (HRBS)**: HRBS applies simple rules on the agent's knowledge domain, using the keywords, which are generated from NNAM, to infer the actual information required in executing the user's task.
- **Peers Co-operation Unit (PCU)**: It is a channel to cooperate with other agents to ask and answer questions.

In the rest of this chapter, the discussion is mainly concentrated on the implementation issues of the task classification unit, in which the neural network learning techniques are applied.

4.1 Problem Domain

Travel reservation is considered as a problem domain to apply the ideas of agent-user interaction through natural language interface and learning over time in order to be able to assist the user. The aim is to build a task classification unit that is able to classify the user's input request to a specific task category.

By natural language interface is meant manipulating a short sentence or phrase and allowing misspelling and/or ungrammatical cases. We do not claim that the agent deals with this problem in a satisfactory way from the semantic or lexical analysis point of view. Since a deep text analysis cannot be undertaken in an unrestricted semantic environment, the approach must be to limit the task in order to analyse the user's input text as well as possible. It has been claimed that most of the successful natural language understanding systems share two properties: they are focused on a particular domain rather than allowing discussion of any arbitrary topic, and they are focused on a particular task rather than attempting to understand language completely [8].

In order to be able to assist users, an agent must be provided with knowledge of its domain. The agent is given a minimum of background knowledge and learns the appropriate behaviour either from the user or from other agents. By learning is meant learning by example and accumulating the experience from interacting with and observing its user.

To tackle the problems of learning and natural language interface, it is assumed that the agent's vocabulary is restricted to the task domain language, i.e., the whole language is divided into many sub-languages. Later, the results will show that the neural network approach is able to associate the user's input sentence (information as a stream of words) with the user's concept within this restricted domain. This assumption enables us to design a simple user-agent interface form to accept unrestricted simple natural sentences or phrases.

4.2 Data

Twenty-five e-mails written in the English language are collected from persons with different native languages. Each e-mail contains a few sentences or phrases representing the three specified categories in the airline travel reservation area. These three categories are:

1. Asking about travel reservation.
2. Asking about travel confirmation.
3. Asking about travel cancellation.

It has been asked that every respondent write a short sentence or a phrase representing as much as possible of the meaning without any concern about the correct grammar. The e-mails contain a mix of formal and informal English language. The only correction, which has been made before this set of data has been applied on the neural network for training, was a spelling check. In this approach, it is assumed that:

- The user's sentences (most of the input stream of words) are within the vocabulary domains.
- The sentences are not differentiated according to their grammar.
- The user's input sentences are not compound (simple requests within the domain language).

The collected e-mails (dataset) consist of a combination of more than one hundred sentences and phrases. This dataset contains more than three hundred different words. A few no-meaning phrases are added to represent the neutral category in the classification process. The neutral category is supposed to include all the common words that have no effect on the task identification process (reservation, confirmation or cancellation) such as a country name. Part of the dataset is chosen as a training group (approximately 30% of the whole dataset). All the sentences and phrases are contained in the test group.

Two types of ANN architectures have been trained using this dataset: a Multi-Layer Feedforward Network (MLF) and a Single-Layer Perceptron Network (SLP). With certain adjustments to the initial weights, learning constant and steepness coefficient, both neural network architectures give the same result. Changing the input representation from unipolar (where active neurons are represented by +1 and 0 represents in-active neurons) to bipolar (where active neurons are represented by +1 and in-active neurons are represented by -1) has a slight effect on the network performance. The activation of a neuron depends on the presence of the word that the neuron represents in the input stream of words. The order of the training data has more influence on the training process. This is due to the finite size of the training set. Here is an example to explain what training order effect means. The word Egypt has the following weights according to the training set in which the word is encountered four times (one time per category).

	reservation	confirmation	cancellation	neutral
Egypt	-4.414566	-8.017722	-0.757465	6.551010

However, the ideal weights for such a word should be approximately the same for all categories but the neutral category. In the actual case, there are differences because of the finite size and the order of the training set (order means which category was trained first). If the words appeared many times in a random category order, this problem could be overcome. On the other hand, this will elongate the training time. It has been deduced that, in order to obtain a fair word-to-concept map distribution from any trained neural network, the training dataset must be carefully selected. In other words, the training sentences and phrases should be prepared such that the neutral (common) words must be represented equally for all the defined categories in the training dataset. This is the main reason for adding the no-meaning phrases: to teach the neural network how to neutralise the effect of the presence (absence) of these words in the user’s input sentence/phrase. The neutral category contains all the words which should have equal weights towards the three categories mentioned above such as London, to, on, for, … and/or from.

The other factor which affects the learning time is the training set size. The two architectures are trained with different size training sets. It is obvious that the larger the training set the longer the training time. In the next section, we will explain two learning processes which complement each other to compensate the above mentioned pitfalls in the training set.

4.3 Network Training

There are two types of training for either an MLF network or an SLP network: an off-line training and an on-line training. The off-line training aim is to construct concept maps from the training examples. These concept maps relate the each input sentence/phrase to a specific concept in the problem domain. The off-line training takes place before all the operations in order to assign connection weights to all the words in the training set patterns. A pattern consists of a unipolar (bipolar) representation of the training sentence or phrase. For example, the sentence could be:

Due to unforeseen circumstances I have to cancel my flight to London

Then the pattern would be:

-1 -1 -1 -1 -1 -1 -1 1 -1 … -1 -1 1 -1 1 -1 -1 -1 1 -1 -1 -1 -1 -1 1 -1….

Each input neuron represents a word and each output neuron represents a category. The learning is according to the generalised Delta learning rule: [9]

$$c(d_i - o_i) f'(net_i) y_j \qquad j = 1,2,......,n$$

$$net_i = \sum_{j=1}^{J} w_{ij} y_j$$

$$o_i = f(net_i)$$

where

w_{ij}	is the weight that connects the output of the jth neuron with the input to the ith neuron
o	is the output vector
c	is the scaling factor
f	is the activation function (sigmoid function)
f'	is the activation function first derivative
y	is the input vector
d	is the target vector

Figure 2 is the graphical representation of the classified words into the correct category after the off-line training is completed.

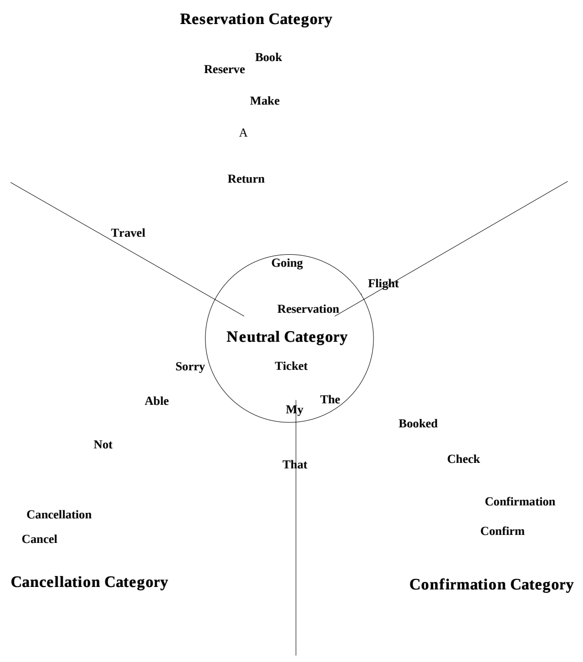

Figure 2. Word classification sample based on the collected data.

Table 1 shows a sample of the weights for some words after SLP[1] network training that takes place based on the collected dataset. Before the training all the words were assigned small random weights.

[1] SLP results show a clearer relation between words and categories than MLF.

Table 1. SLP training results (sample).

WORD	**CATEGORIES**			
	RESERVATION	*CONFIRMATION*	*CANCELLATION*	*NEUTRAL*
a*	8.160717	-6.392762	-9.916912	-11.262650
able	-3.469685	-1.873571	7.569377	-4.008598
airline	1.778800	-6.574209	-2.470539	-7.258359
belgium	-0.636226	-6.133667	-3.362907	3.883109
book	26.382210	-13.742558	-8.563011	-7.614223
booked	-2.502439	8.311028	-2.922199	-3.853302
cancel	-21.769215	-24.374889	53.473717	-20.940538
cancellation	-21.756435	-12.751144	31.506802	-2.847031
check	-14.144594	22.119167	-2.908264	-2.108434
confirm	-16.029274	34.444370	-20.502920	-8.255243
confirmation	-21.811161	13.502781	-17.121601	-5.858447
egypt	-4.414566	-8.017722	-0.757465	6.551010
flight	2.957987	3.815885	-7.900447	-8.968528
going	5.197784	-8.156089	-7.538003	5.770500
make	11.682374	-6.972619	-1.942448	-14.292900
my	-5.785409	2.654307	1.096750	3.675545
need	-3.990644	-0.253858	-8.594911	5.843002
not	-3.469685	-1.873571	7.569377	-4.008598
reservation	1.135525	2.715665	1.451366	0.494692
reserve	20.650588	-4.416985	-2.991275	-8.332147
return	6.652804	-1.805905	-2.937553	-0.380169
send*	-0.74305	3.753846	-3.041971	3.572052
sorry	-6.61513	-5.857497	2.198464	-0.346198
the*	-14.7968	2.616945	-0.955008	2.513896
ticket	1.48133	5.895713	2.516594	0.510546
travel	8.168785	-7.046576	1.490917	-8.466378
trip	-7.755359	-6.876982	-0.422680	5.903522
want	-4.800109	-5.958756	-3.849468	6.855282

Inspection of the above results indicates that there is a strong connection between some words and a specific category. For example: words like **book, reserve** and **make** are directly related to the *reservation* category, while words like **check** and **confirm** are connected to the *confirmation* category. On the other hand, words like **reservation** and **ticket** are common among more than two categories. We intentionally added words like **send** to the training set to illustrate the effect of the unequal representation of the neutral words. The word **send** has appeared only twice in the training dataset: one time in a confirmation training example and the second time within the no-meaning phrases. It can be seen from the table that this word is related to both *confirmation* and neutral categories; however, it is supposed to be in the *neutral* category only. Finally, it can be noticed that some words like **the** and **a** are assigned into a certain category and this is interpreted as language dependent. In other words, most of the users use **a** when they ask about a ticket reservation and they use **the** when they ask about the ticket confirmation.

The other learning process is called the on-line training. The on-line training takes place during the user-agent interaction cycle. The network adapts some of its weights according to the user's direct response in order to correct the misclassified sentence/phrase to a different category. Also, it assigns weights for new words. The on-line learning rule is:

$$\mu_{ij}\varepsilon(d_i - o_i)$$

where

μ_{ij}	is a multiplicative factor
ε	is an on-line learning coefficient
d, o	as defined above in the off-line training.

Each word has been assigned a correctness level value which is dependent on how many times the word is used before. The multiplicative factor is inversely related to the word correctness level value. Any word in the user's input and not belonging to the domain dictionary is considered new. The newer the word the higher the multiplicative factor and vice versa. The correctness level value of a word is increased each time the word is encountered in the input stream of words and the user's request is classified correctly. Hence, the change in the weights of the more often used words is much slower than the change in the weights of the new words during the on-line learning. The on-line learning coefficient is a small number defined by the user to control the speed of the on-line training process.

The on-line training is a complementary process to the off-line training. Also, it introduces a solution to the lack of enough data in the training set which leads to some sort of incorrect bias in the network weights. In addition, the whole learning process time is divided between those two processes.

5 Operation

Figure 3 shows the block diagram of the task classification unit in the interface agent. Once the off-line training process has been completed, the system is ready to move to inference and on-line training through agent-user interaction. The interaction cycle works as follows:

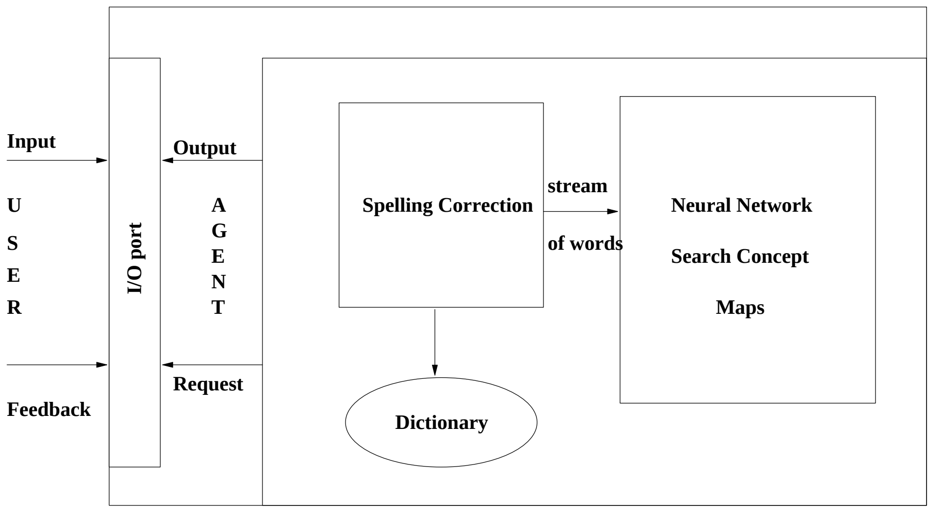

Figure 3. Task classification unit block diagram.

- **Step1**) *User-system interaction*: The user's information is provided as a simple sentence or phrase at the I/O port.
- **Step2**) *Approximate string matching*: Given the domain dictionary string set, the user's input stream of words is matched against the dictionary items to correct any spelling errors in the input words. A non-matching text item can be computed with a dictionary directly, using an iterative character-by-character matching scheme that determines the smallest number of single-character operations (insertions, deletions, substitutions, and transpositions) required to transform one of the words into the other. This is a well-known technique of approximate string matching [10]. It is assumed at this stage that the dictionary contains the travel domain vocabulary only. This means some correct English words might be considered misspelled and corrected according to this vocabulary domain. For example:

(Not in the dictionary) **reverse → reserve** (Nearest match)

The output of this step is the correct stream of words (if found) and the rest will be considered new words, even the wrong ones, until the system receives a feedback from the user in step5.

- **Step3**) *Pattern generation*: Some of the input neurons are activated according to the words presented in the user's input. The input neurons in the neural network are chosen large enough to accommodate the new words. The rest of the neurons that are not active indicate the absence of the corresponding words from the input stream of words. A new word can be assigned to any unused neuron or a neuron that represents the neutral word, which has the least correctness level value. This is the process of generating the patterns from the input stream of words. The generating patterns guide the neural network to an initial state (via a trajectory) in the concept maps that have been constructed during the off-line training phase.
- **Step4**) *Search process*: The neural network continues the search process to classify the input pattern to a certain category according to the off-line training. The output of this step is either directed to the I/O port to get a feedback from the user or is considered correct by the agent. This decision is mainly dependent on a factor defined by the user, called agent's confidence level. This factor should be increased with time as the neural network weights settle to the right direction. The outputs are the classified category and/or the words that were considered new or wrong in step2.
- **Step5**) *User-system interaction*: The user response to the output of the previous step is provided as corrected words, a corrected category and/or an acceptance of the agent’s decision.
- **Step6**) *On-line learning*: According to the user's feedback, the agent corrects the words, adds new words to the dictionary and updates the neural network weights according to the on-line learning rule.

Table 2 presents some examples of the agent-user interaction.

Table 2. Simulation transcription.

Example (1):

User Input:	I'd like to book an airline ticket to London
Agent Reply:	The input is considered as category 1, travel reservation(C/W?)
New words:	No words to correct.
User feedback:	Correct

Example (2):

User Input:	Contact the travel agency to cancel my ticket to London
Agent Reply:	The input is classified into category 3, travel cancellation(C/W?)
New words:	agency, contact
User feedback:	Correct

Example (3):

User Input:	Arrange me a trip to Cairo
Agent Reply:	The input can not be identified, unknown(C/W?)
New words:	arrange, Cairo
User feedback:	No. Category 1, reservation Cairo
Agent Reply:	Done

6 Summary

We have introduced the task classification unit in an interface agent that relies on the neural network learning techniques to acquire the knowledge it needs to assist its user. It is designed to achieve the properties of interacting with the user through natural language interface and learning over time. The ability to build such systems is based on the assumption of the restricted vocabulary domain. The agent learns gradually to better assist the user. We argued that such a gradual learning approach is beneficial as it allows the user to incrementally become confident with the agent’s decision. The implemented task classification unit has been tested with real world data in the travel reservation domain. The results show that the system is able to classify the user's input correctly and learn over time.

References

[1] Maes, P. (1995), "Intelligent software: programs that can act independently will ease the burdens that computers put on people," *Scientific American*, vol. 273, no. 3, pp. 66-68.

[2] Maes, P. (1994), "Agents that reduce work and information overload," *Communications of the ACM*, vol. 37, no. 7, pp. 31-40.

[3] Pun, C. and Li, Y. (1998), "Machine translation with corpus-base support," *Proceedings of Fourth International Conference on Computer Science and Informatics North Carolina,* pp.158-161.

[4] Wooldridge, M. and Jennings, N.R. (1995), "Intelligent agents: theory and practice," *The Knowledge Engineering Review,* vol. 10, no. 2, pp.115-152.

[5] Tsui, K.C., Azvine, B., and Plumbley, M. (1996), "The roles of neural and evolutionary computing in intelligent software systems," *BT Technology Journal*, vol. 14, no. 4, pp. 46-54.

[6] Wasserman, P. (1989), *Neural Computing Theory and Practice,* Van Nostrand Reinhold, New York.

[7] Aboulenien, H.A. and De Wilde, P. (1998), "A simple interface agent," *Proceedings of Fourth International Conference on Computer Science and Informatics North Carolina,* pp. 190-192.

[8] Russell, P. and Norvig, P. (1995), *Artificial Intelligence: A Modern Approach,* 2nd ed., Prentice-Hall, New Jersey.

[9] Zurada, J. (1992), *Introduction to Artificial Neural Systems,* West Publishing Company, St. Paul.

[10] Salton, G. (1989), *Automatic Text Processing*, Addison-Wesley, New York.