Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

ELSEVIER International Journal of Approximate Reasoning 32 (2003) 237–258

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

www.elsevier.com/locate/ijar

Symbolic state transducers and recurrent neural preference machines for text mining

Garen Arevian, Stefan Wermter, Christo Panchev

The Informatics Centre, School of Computing and Technology, University of Sunderland, St. Peter's Campus, St. Peter's Way, Sunderland SR6 0DD, UK

Received 1 January 2002; accepted 1 April 2002

Abstract

This paper focuses on symbolic transducers and recurrent neural preference machines to support the task of mining and classifying textual information. These encoding symbolic transducers and learning neural preference machines can be seen as independent agents, each one tackling the same task in a different manner. Systems combining such machines can potentially be more robust as the strengths and weaknesses of the different approaches yield complementary knowledge, wherein each machine models the same information content via different paradigms. An experimental analysis of the performance of these symbolic transducer and neural preference machines is presented. It is demonstrated that each approach can be successfully used for information mining and news classification using the Reuters news corpus. Symbolic transducer machines can be used to manually encode relevant knowledge quickly in a data-driven approach with no training, while trained neural preference machines can give better performance based on additional training.
© 2002 Elsevier Science Inc. All rights reserved.

Keywords: Finite state automata; Symbolic transducers; Recurrent neural networks; Preference Moore Machines; Hybrid systems; Text classification

E-mail addresses: garen.arevian@sunderland.ac.uk (G. Arevian), stefan.wermter@sunderland.ac.uk (S. Wermter), christo.panchev@sunderland.ac.uk (C. Panchev).
URL: http://www.his.sunderland.ac.uk.

1. Introduction

1.1. General background and motivation

The quantity of information on the Internet has motivated a need to design more sophisticated learning systems that are capable of classifying the massively heterogeneous, faulty, incomplete and ever-changing amount of textual information. This need is particularly apparent for classifying news, for example from the Reuters newswire and the World Wide Web. Much initial work in the field of text mining and classification has used manual encoding techniques or techniques from information retrieval [31]. However, a great many of these approaches do not have appropriate, robust properties that will enable them to handle the variety of unconstrained real-world data. The need for information extraction and classification approaches to have automatic adaptation capabilities, built-in learning algorithms such as those from artificial neural networks, the ability to deal with incomplete text information, and just the sheer scale of available text data are more important constraints now [43]. Hence, there has been a greater focus on machine learning techniques that are able to handle natural language processing [8,23,25].

Various learning agent systems [3,22] have been designed to perform a number of tasks, whether they be classification [14,29], information retrieval and extraction [8,11], routing of information [40,42] or automated web browsing [2,7,27]. In general, robust learning architectures have been identified as important current areas for natural language processing [4,9]. One class of techniques which have been widely used are statistical techniques. Such approaches have been shown to perform successfully in the classification and parsing of text data [5]. However, these statistical methods also require assumptions about the distribution, and are less effective when the classification task has to be achieved with no a priori knowledge.

Hence, some types of artificial neural networks, with their adaptive, distributed and robust learning capabilities have been applied to the task of textual classification. For example, self-organizing maps (SOMs) [17] have been used to create contextual mappings of newsgroup corpora; a SOM forms a non-linear projection from a high-dimensional space onto low-dimensional space and has been used in the WEBSOM project [16,18] to create contextual mappings of word-vector representations. The SOM algorithm computes a collection of models that approximate the data by applying a specified error criterion; this allows an ordering of the reduced dimensionality onto a map. The SOM is acting as a similarity graph of the data and is useful for structure visualization, data mining, knowledge discovery and retrieval [1,13].

Learning web agents using neural networks such as [18,34,40] hold a lot of promise as they support robustness and learning, are relatively autonomous in their learning behaviour and offer the potential of on-line adaptivity. Recurrent neural systems not only are able to embody some sort of contextual information, but they have the inherent ability to simulate any finite-state machine [19], essentially allowing an abstraction of the information within a recurrent neural network [6,32] into discrete representations such as in symbolic transducers. This relationship between recurrent networks and symbolic transducers is addressed in this paper. The experiments with recurrent neural networks and transducers presented in this paper have been tested on noisy, real-world data and benchmarked on the Reuters news corpus [40,41].

One main motivation for using different modular components is that they can lead to greater robustness, can generalize better, and classification tasks and target functions may be reached more readily [33]. Failure of one component does not necessarily mean an overall failure of the task, and indeed benefits can arise from a subsequent combination of the various modular components. Another benefit is that such systems, for instance for modular classification, could form their own representations for a specific subtask. For example, mixture of experts approaches show that performance can be improved. Furthermore, though recurrent networks are able to encode sequentiality, finite state machines can encode rules directly, and combinations of them could give better generalization.

1.2. Preference Moore Machines

There has been previous work on introducing Preference Moore Machines [38] as a method of interpreting symbolic and neural machines. Here this framework is applied to a real-world test of learning news classification.

A Preference Moore Machine is formally defined as a synchronous sequential machine that codes a sequential preference mapping, using current state S and the input preferences I, to assign an output preference O and a new state S. A Moore Machine is able to transduce knowledge from an input to output whilst maintaining context [38,39].

Preference Moore Machines can be seen as neural networks (Neural Preference Moore Machines) or as symbolic transducers (Symbolic Preference Moore Machines) as shown in Fig. 1. For a Neural Preference Moore Machine, the internal state of the system and the context are represented as a n-dimensional vector. Using the Euclidean distance metric, different mappings can be made between this vector representation and its symbolic interpretation and vice versa [39]. Symbolic transducers are considered to be symbolic Preference Moore Machines. Rather than extracting the rules from the training material, it is possible to encode the relationships and generate a transducer from regular expressions.

Symbolically encoded transducers and neurally learned versions of Preference Moore Machines potentially represent very different forms of knowledge, and can potentially be combined [38]. This leads to systems using different

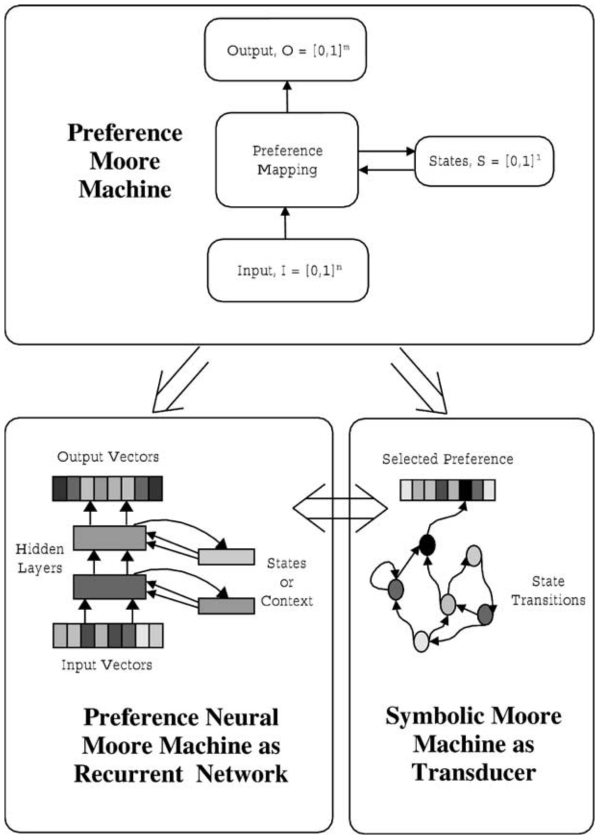

Fig. 1. Relationship between different types of Preference Moore Machines.

agents with different representations. It has been shown that recurrent neural networks can indeed act as robust and scalable classifying agents for sequential tasks such as the classification of a stream of textual information of arbitrary lengths [42]. This work on neural agents [40–43] has been demonstrated for the task of textual classification. In this current work, a multiple agent system is developed which compares the concepts and performance of symbolic transducers with the neural preference machines.

1.3. Classification corpus used

In order to test the concepts of preference machines, a news classification task is used, in this case, the Reuters-21578 text classification test collection [21]. This corpus contains documents from the Reuters newswire service. All

Table 1
Example titles from the Reuters-21578 corpus showing the different categories and the structure of the sentences

Semantic category	Examples
COrporate	"Nationale Nederlanden Profits, Sales Steady"
SHipping	"Japanese Shipyards To Form Cartel, Cut Output"
ENergy	"US Energy Futures Called Unchanged To Lower"
INterest	"Fed Adds Reserves Via Customer Repurchases"
MoneyFx	"UK Money Market Deficit Forecast Revised Upwards"
EConomic	"German Industrial Output Rises 3.2 PCT In February"
CuRrency	"Japan Business Leaders Say G-7 Accord Is Worrying"
CoMmodity	"Shanghai Tyre Factory To Raise 30 MLN US Dollars"
SHipping and ENergy	"Soviet Tankers Set To Carry Kuwaiti Oil"
MoneyFx and EConomic	"Taiwan Dollar And Reserves Seen Rising More Slowly"

news titles in the Reuters corpus belong to one or more of eight main categories: Money/Foreign Exchange (MoneyFx/Foreign Exchange, MF), Shipping (SHipping, SH), Interest Rates (INterest, IN), Economic Indicators (EConomic, EC), Currency (CuRrency, CR), Corporate (COrporate, CO), Commodity (CoMmodity, CM), Energy (ENergy, EN). This corpus allows a comparison between different approaches for the task of text classification. Table 1 describes some examples. The corpus used consists of approximately 10 733 titles, the documents of which have a single title and at least one associated topic category. For the training set used for the neural preference machine, 1040 news titles were used, the first 130 of each of the eight categories. All the other 9693 news titles were used for testing the generalization to new and unseen examples. For manually deriving the symbolic transducer machines, a smaller subset of 400 titles was used, randomly selecting subsets of 50 titles from each category.

2. Symbolic transducers

2.1. Definition of Moore Machine as symbolic transducer

A transducer is a synchronous sequential machine with output; it is defined as follows: a synchronous sequential machine M is a 5-tuple, with $M = (I, O, S, f_s, f_o)$, where

- I, O are finite non-empty input and output sets,
- S is a non-empty set of states,
- the function $f_s : I \times S \rightarrow S$ is a state transition mapping function which describes the transitions from state to state on given inputs,
- the function $f_o : S \rightarrow O$ is an output function.

While a finite state automaton is a system that either accepts or rejects a specific sequence, a transducer on the other hand transforms or "transduces" the sequence into a different output representation. This process generates new output sequences. Therefore word representations can be mapped to class representations, and thereby support classification.

2.2. Introductory example

A regular expression is an effective way of defining a pattern for classification. Firstly, regular expressions are specified which are then transformed into finite-state transducers using a transducer manipulator [35]. The example shown in Fig. 2 is a transducer that encodes the regular expression $(0^{*}\ en^{+}) \rightarrow energy$, where the asterisk "$*$" (Kleene star) indicates that the symbol can occur *zero or more* times in a sequence, and where the plus "+" (iteration; one or more concatenations of symbol) indicates that the symbol must occur at least *once or more than once* in a sequence.

The "en" is a symbol for a semantic word representation, the "0" is any single symbol for any tag, "energy" is the semantic class representation which is assigned to the sequence if transduced successfully. The symbol ":" separates input and output for a particular transition from one state to another, "[]" indicates no output. Thus, the transducer is able to read the input stream and produce sequences that are essentially tagged representations of the inputs, where the tags represent semantic classes.

Some examples of the transducer's behaviour are illustrated next which should give an insight into the behaviour of such systems; the transducer will

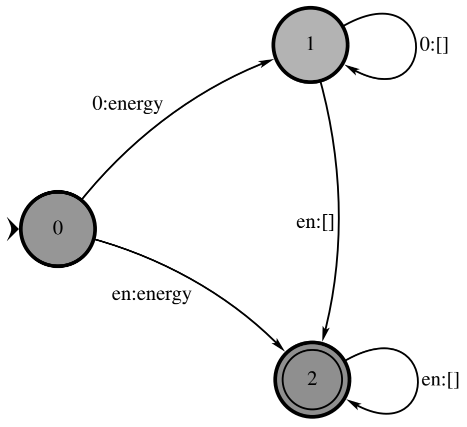

Fig. 2. A simple example of a regular expression shown as a finite-state automaton acting as a transducer. The node "0" is the start state, the node "1" an intermediate state and node "2" is the final or end state of the transducer.

output to the interest class according to the rules of the regular expression. The *0* is the initial state indicated by the arrow, *1* is the intermediate state indicated by a single circle, and *2* is the end or final state indicated by a double circle.

- $(0^* \; en^+) \rightarrow energy$,
- Input stream: (0) → *reject* (remains in state 1, hence fail),
- Input stream: (*en*) → *accept* → *energy* (directly jumps to end state 2),
- Input stream: (0 *en*) → *accept* → *energy* (ends in end state 2),
- Input stream: (0 0 *en* 0 → *reject* (last "0" symbol cannot be processed at end state 2, hence reject),
- Input stream: (0 *en* 0 0 0 0 0 0 *en*) → *reject* (goes from state 0 to state 1, then reaches end state 2; however, the subsequent "0" symbols cannot be processed at the end state, hence reject),
- Input stream: (0 *en en en en en*) → *accept* → *energy* (the transducer reaches the end state 2 when "0 *en*" are input, and since the successive "en" symbols cause no further change in the end state 2 that is reached, the sequence is accepted and transduced).

2.3. More examples from Reuters corpus

Fig. 3 shows the regular expression denoted as $((0^* \; en^+ \; mf^+ \; 0^*)^+) \rightarrow energy$. Therefore, the transducer would be able accept the sequence (0 *en mf* 0 *en mf* 0) but not (0 *mf en mf* 0) as it explicitly expects (*en*) at the start of a sequence followed by (*mf*) ultimately at the end of the sequence.

Fig. 4 is a transducer able to handle sequences that are encoded by the regular expression $(co^*mf^*in^+in^*mf^+)^+) \rightarrow interest$ – this only accepts

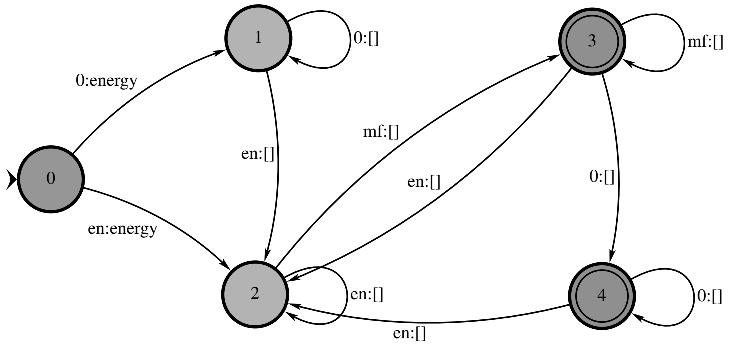

Fig. 3. A transducer encoding the regular expression $((0^* \; en^+ \; mf^+ \; 0^*)^+) \rightarrow energy$ for classifying a specific sequence of tags "en" followed by "mf" into the ENergy category. The tags "en" and "mf" must appear at least once in a stream of symbols interspersed with an arbitrary number of other tags – this transducer is more robust for sparser representations (e.g., the body of a newswire article or longer sequences from longer titles).

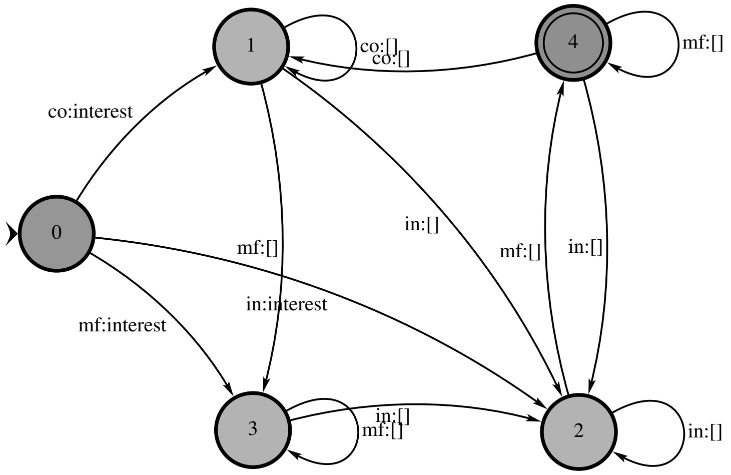

Fig. 4. A transducer encoding the regular expression $((co^{*}\ mf^{*}\ in^{+}\ in^{*}\ mf^{+})^{+}) \rightarrow interest$ for detecting a specific semantic sequence of tags "in" followed explicitly by "mf" that must appear at least once in a stream of tags interspersed with an arbitrary number of other tags, in this example, "co" and "mf", to give a transduction to the INterest category – this transducer is able to handle a denser representation of semantic sequences with more specific rules, and also shorter sequences that have a very specific pattern of tags.

sequences that are explicitly in the INterest category and have one instance of the symbol (*in*), followed by an arbitrary number of other (*in*) symbols, and one instance of the symbol (*mf*) must be present for correct transduction to the appropriate category. As before, 0 is the start state, 1, 2 and 3 are the intermediate states, and 4 is the final end state.

2.4. Construction of transducers from regular expression

A set of eight Preference Moore Transducers were constructed that encoded the regular expressions that would be able to classify correctly a discrete symbolic sequential input of word representations, and produce either an appropriate transduction to a semantic class or true/false value signifying acceptance/rejection.

Four of the actual regular expressions are shown in Table 2, derived to represent the possible semantic word representations within each class. During the coding stage, the derivation of transducer representations was achieved with ease, and improved performance was quickly observed by adjusting elements of the regular expression; this suggested that the generative rules for the classification task are relatively simple, though non-trivial. This also indicates that the derivation of the rules via a top-down approach is achievable in principle.

Table 2
The actual regular expressions used to generate the appropriate category transducers as shown by Figs. 5–8

Semantic category	Regular expression encoding semantic rules for category
MoneyFx	$(in^* cr^* ec^* mf^* cr^* en^* ec^* cr^* mf^* ec^* ec^* en^*)$
EConomy	$((cr^* mf^* cm^* in^* ec^* ec^+ in^* en^* ec^* ec^* cr^*)^+)$
CuRrency	$((in^* ec^* cr^* cr^+ mf^* cr^* sh^* cr^* ec^* in^* ec^*)^+)$
CoMmodity	$(cm^* sh^* cm^* cr^* cm^* ec^* cm^* sh^* cr^*)$

2.5. Experimental description and examples

The titles are symbolically tagged according to the most frequent occurrence of the tag for a particular word. This results in a sequence of tags, e.g., of the form (*en cm cm co co*), which represent a semantic tag sequence for one specific title from the corpus. Issues such as the exclusion of stop-words [42], stemming and rounding have been considered previously; for example, in the case of the removal of stop-words (i.e., insignificant words such as "the", "a", "and", etc., that may have an average distribution and are domain-independent across all categories), it was shown that there is only a little improvement in terms of classification accuracy. However, it can also be argued that in a semantic sequence, stop-words may indeed have an important influence since they may be an indication of a unique sequence; for example, the "of" in the phrase "Bank of England", could bias the sequence towards "England" and the EConomy category if there are enough examples of the phrase itself in the set of all titles.

One basic heuristic in the construction of the regular expressions for the semantic sequences was to encode the presence of the specific category tag itself somewhere within the sequence – i.e., it was assumed that in general, sequences would be weighted towards having a greater number of the semantic tags belonging to that category itself, as shown in Table 2; for example, it can be seen that in the regular expression for coding the CoMmodity transducer, there are four occurrences of the symbols "cm". This approach in designing the transducers can be seen as a top-down heuristic integration of a priori knowledge to aid classification.

Table 3 demonstrates the specific case of the derivation of the MoneyFx transducer (Fig. 5); sequences of regular expressions were systematically built-up from a basic example (e.g., stage 1), to give a final version that encoded a very complex expression (stage 11). The classification/recall figures of the resulting transducer at each stage were used as a guide to change the component expressions of the system to improve classification/recall performance. It can be seen, for example, that the introduction of the symbol "mf" in the penultimate position of the expression at stage 8 causes the expression at the next stage to improve classification performance by 16%. The introduction of

Table 3
Table showing the heuristic approach adopted for deriving the MoneyFx transducer (Fig. 5), using regular expressions that are systematically built-up to generate the appropriate sequential rules for that category

Stage	Regular expression	Classification/recall
1	(mf^*)	28%
2	(mf^*cr^*)	28%
3	$(cr^*mf^*in^*cr^*ec^*)$	32%
4	$(in^*cr^*mf^*in^*cr^*ec^*)$	36%
5	$(in^*cr^*ec^*mf^*in^*cr^*ec^*)$	48%
6	$(in^*cr^*ec^*mf^*cr^*cr^*ec^*en^*)$	48%
7	$(in^*cr^*ec^*mf^*cr^*en^*ec^*en^*)$	48%
8	$(in^*cr^*ec^*mf^*cr^*en^*ec^*cr^*en^*)$	56%
9	$(in^*cr^*ec^*mf^*cr^*en^*ec^*cr^*mf^*en^*)$	72%
10	$(in^*cr^*ec^*mf^*cr^*en^*ec^*cr^*ec^*ec^*en^*)$	72%
11	$(in^*cr^*ec^*mf^*cr^*en^*ec^*cr^*mf^*ec^*ec^*en^*)$	72%

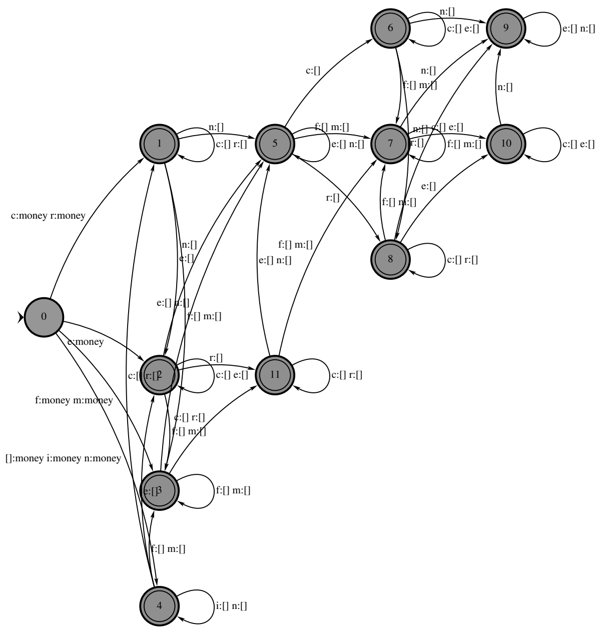

Fig. 5. MoneyFx Transducer.

further terms in stages 10 and 11 do not further improve classification performance, and it is assumed that a good solution has been reached, at least by using the top-down heuristic coding scheme.

2.6. Recall and precision

The two main parameters which are generally used to determine the effectiveness of how well a classification task has been achieved are *recall* and *precision* [30]. *Recall* is defined as the *ratio of the number of relevant titles of a specific category that are correctly classified over the total number of relevant titles for that category* (the total being a sum of the relevant titles classified plus relevant titles not classified correctly). That is, recall equals classified and relevant, divided by relevant titles. *Precision* is defined as the *ratio of the number of relevant titles correctly classified over the number of relevant titles correctly classified plus the non-relevant titles that are classified.* That is, precision equals classified and relevant divided by classified titles. By obtaining this value for performance, the effectiveness of the transducers can be compared and contrasted with other approaches. There is an inverse relationship between recall and precision values, and usually the performance of a system is a trade-off between the two. For example, high recall values indicate that a system may be generalizing too much, at a cost to precision; high precision but low recall indicates that a system may not be able to handle more ambiguous classes.

2.7. Results and discussion

Table 4 shows the recall values; for example, it can be seen on the first line that passing the CO data set sequences through the specifically designed COrporate transducer, the recall value is 66%; however, passing the SH data set sequences through the same transducer gives a value of 34%, showing that the transducer is fairly specific to the COrporate category. For the IN data set

Table 4
A breakdown of the recall values for the eight transducers of the eight categories – the bold figures show the actual recall value specific to that category. However, the breakdown allows a more detailed analysis of the recall behaviour of the transducers across all the eight category data subsets

Category of sequences	CO	SH	EN	IN	MF	EC	CR	CM
COrporate	**66%**	34%	16%	12%	16%	48%	42%	64%
Shipping	10%	**84%**	10%	6%	4%	10%	2%	16%
ENergy	16%	28%	**70%**	16%	12%	62%	24%	14%
INterest	12%	2%	50%	**76%**	16%	24%	12%	0%
MoneyFx	0%	0%	24%	16%	**72%**	64%	68%	40%
EConomy	0%	0%	40%	38%	26%	**70%**	68%	24%
CuRrency	0%	32%	10%	40%	36%	62%	**76%**	62%
CoMmodity	0%	40%	0%	0%	32%	32%	24%	**72%**

The eight transducers are represented respectively as follows by their symbol notation: COrporate, SHipping, ENergy, INterest, MoneyFx, EConomy, CuRrency, CoMmodity.

sequences, the value is 12%, suggesting that the INterest and COrporate categories are very different. The CR data set sequences passing through the MoneyFx and EConomy transducers give high values of 68% each, and 76% for the specific CuRrency transducer itself; this obviously suggests that there is a close relationship between these three classes. In general, these symbolic machines perform reasonably well, given the simple representation.

This breakdown of the experimental results for the recall behaviour gives relevant information and highlights the differences between the categories and how closely or not they may be related to one another. Semantic sequences from a particular category may wrongly be classified for several reasons – for example, the category allocations may depend on human-level interaction that does not take into consideration strict semantic representation but rather a more heuristic allocation to a particular category that may be arbitrary; there will also be ambiguities with specific words that form the title sequences which may belong to more than one or more categories.

Table 5 shows the recall and precision performance for each of the transducers; the percentage values for the irrelevant titles classified for each transducer are also shown as they form part of the function for deriving the precision value of the transducer; they can also be interpreted to be a measure of the classification "accuracy", the lower the percentage of non-relevant titles classified, the higher the recall and precision values. By cross-testing the respective data set collections with the transducers for the other categories as in the breakdown in Table 4, it was shown that the heuristic rules derived from the semantic sequences had poor overall precision, but gave relatively good recall values.

Figs. 5–8 show the four examples of the actual transducers constructed that performed the classification task; it can be seen that even relatively simple rules can generate automata that can be very complex in structure. Finally, the overall average recall and precision values for the symbolic Preference Moore Machines are shown in Table 6.

Table 5
Recall and precision performances for the transducers with various input sequences of semantic categories

Category of sequences	Percentage irrelevant	Recall	Precision
COrporate	29%	66%	22%
SHipping	7%	84%	59%
ENergy	22%	70%	29%
INterest	15%	76%	40%
MoneyFx	17%	72%	35%
EConomy	24%	70%	27%
CuRrency	31%	76%	24%
CoMmodity	7%	72%	58%

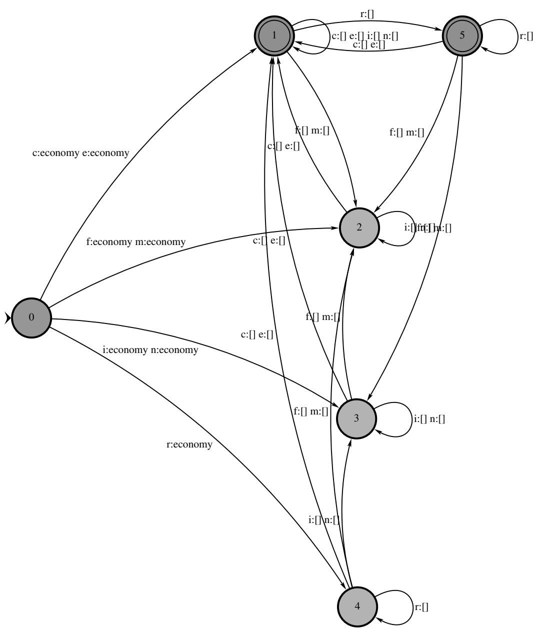

Fig. 6. EConomy Transducer.

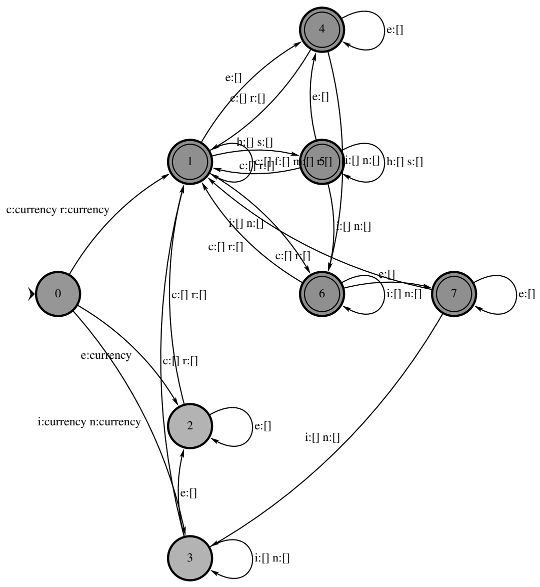

Fig. 7. CuRrency Transducer.

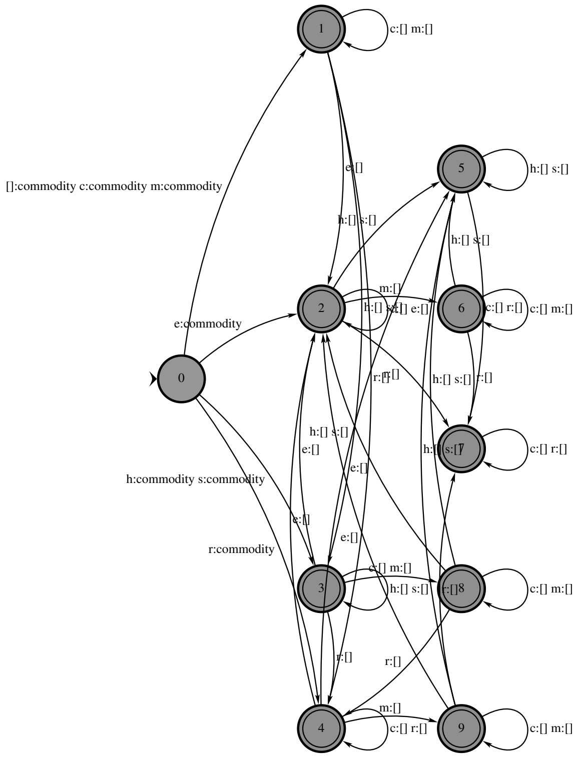

Fig. 8. CoMmodity Transducer.

Table 6
Average recall and precision values for the eight transducers used for the classification experiments

	Recall	Precision
Symbolic transducer performance	73%	37%

3. Neural Preference Machines

In recent years, there has been an increase in the application of neural networks to the task of textual classification; recurrent neural networks, which have feedback from the output to the hidden or input layers, are able to use information from a previous incremental step during training to give a sequential and gradual representation that is dependent on previous time steps. This allows sequential knowledge to be built up.

While a symbolic Preference Moore Machine is encoding top-down knowledge, a neural Preference Moore Machine is learning bottom-up. The

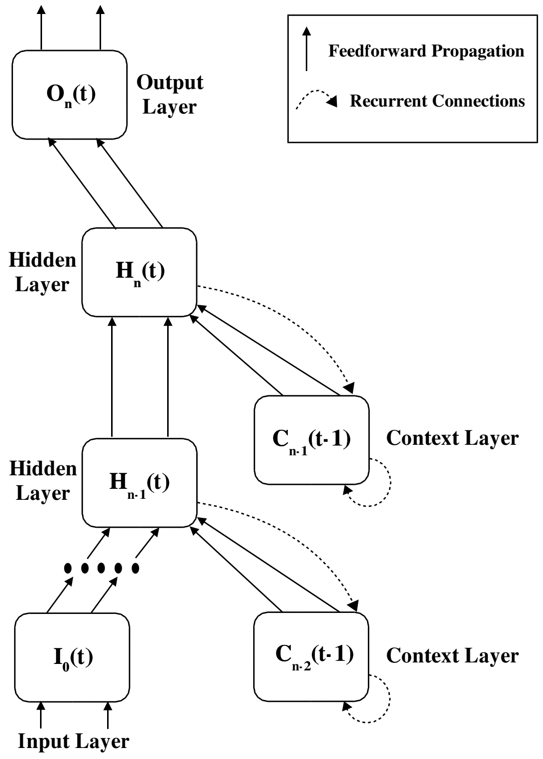

Fig. 9. Neural Preference Moore Machine with two hidden layers.

various forms of neural Preference Moore Machine used in this paper are a neural network with one context layer and a neural network with two hidden layers (Fig. 9) which are trained using semantic vector representations at the input layer [42].

3.1. Recurrent networks

The specific neural network explored here is a more developed version of the simple recurrent network, namely a Recurrent Plausibility Network [37,42]. Recurrent neural networks are able to map both previous internal states and input to a desired output. This makes the input/output mappings of the system dynamic.

Fully recurrent networks process all information and feed it back into a single layer, but for the purposes of maintaining contextual memory for processing arbitrary lengths of input, they are limited. For example, partially recurrent Elman networks have recurrent connections between the hidden and context layer [10] or Jordan networks have connections between the output and context layer [15]; these recurrent connections allow previous states to be kept within the network structure and temporal information is thus represented in the internal states which arise, and is in effect "memory". The temporal unfolding of this recurrent processing results in discrete states being represented over incremental time steps, resulting in the representation of the sequential context of information. A finite state transducer can also analogously represent this sequence of discrete states [6,12,24,32].

However, simple recurrent networks have a rapid rate of decay of information about states. For many classification tasks in general, recent events are more important but some information can also be gained from information that is more longer-term. With sequential textual processing, context within a specific processing time-frame is important and two kinds of short-term memory can be useful – one that is more dynamic and varying over time which keeps more recent information, and a more stable memory which is allowed to decay more slowly to keep information about previous events over a longer time-period.

3.2. Network architecture and learning

Fig. 9 shows the general structure of the recurrent network. Different decay memories were introduced by using distributed recurrent delays over the separate context layers representing the contexts at different time steps [37]. At a given time step, the network with n hidden layers processes the current input as well as the incremental contexts from the $n-1$ previous time steps.

The input to a hidden layer H_n is constrained by the underlying layer H_{n-1} as well as the incremental context layer C_{n-1}. The activation of a unit $H_{ni}(t)$ at time t is computed on the basis of the weighted activation of the units in the previous layer $H_{(n-1)i}(t)$ and the units in the current context of this layer $C_{(n-1)i}(t)$. In particular, the following is used:

$$H_{ni}(t) = f\left(\sum_k w_{ki} H_{(n-1)i}(t) + \sum_l w_{li} C_{(n-1)i}(t)\right).$$

The units in the two context layers with one time step are computed as follows:

$$C_{ni}(t) = (1-\varphi_n) H_{(n+1)i}(t-1) + \varphi_n C_{ni}(t-1),$$

where $C_{ni}(t)$ is the activation of a unit in the context layer at time t. The self-recurrency time span of the context is controlled by the hysteresis value φ_n. The hysteresis value of the context layer C_{n-1} is lower than the hysteresis value of the next context layer C_n. This ensures that the context layers closer to the input layer will perform as memory that represents a more dynamic context for small time periods.

Essentially the learning algorithm for the network uses the backpropagation through time (BPTT) rule [20,26,28,36], but with such a recurrent architecture, the gradients of each state are computed using information from a combination of previous states. One forward pass of the data is made through the network, and the synaptic weight states and desired responses recorded; this is followed by a single backward pass of the record, where local gradients are computed. Once this back-propagation has been done, the synaptic weights are adjusted. By taking as input the weighted sum of incoming activations at a time t, plus the weighted sum of incoming activations from time $t-1$, this second incoming activation allows the previous internal states of the network to be used.

3.3. Experimental description

From the Reuters-21578 described earlier, 10 733 titles of the so-called ModApte split were used, the documents of which have a single title and at least one associated topic category. For the training set, 1040 news titles were used, the first 130 of each of the eight categories. All the other 9693 news titles were used for testing the generalization to new and unseen examples.

The input representations obtained encoded the preference for a specific word to occur in a particular semantic category. The main advantage is that they are independent of the number of examples present in each category:

$$v(w,x_i) = \frac{\text{Norm. freq. of } w \text{ in } x_i}{\sum_j \text{Norm. freq. for } w \text{ in } x_j}, \quad j \in \{1,\dots n\},$$

where

$$\text{Norm. freq. of } w \text{ in } x_i = \frac{\text{Freq. of } w \text{ in } x_i}{\text{Number of titles in } x_i}.$$

The *normalized* frequency of the number of times a word w appears in a semantic category x_i (i.e., *the normalized category frequency*) was computed as a value $v(w,x_i)$ for each element of the semantic vector, divided by normalizing the frequency of the number of times a word w appears in the corpus (i.e., *the normalized corpus frequency*).

3.4. Results

Fig. 10 shows the plots of the sum-squared error of the output preferences against the number of training epochs and each word of the specific title. The network learns correctly the category to which this sentence belongs (in this case, ENergy) when the sum-squared error value is at a minimum at the end of the title. So initially, the word "China" is not correctly classified as it can indeed belong to many categories, but the words "Closes Second Round Of" reduce the error and the words "Offshore" followed by "Oil Bids" cause the

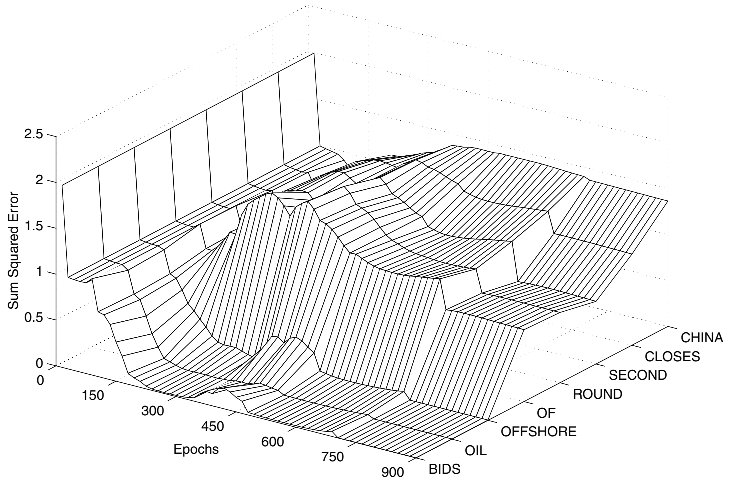

Fig. 10. Error-surface plot for sentence "China Closes Second Round Of Offshore Oil Bids".

network to switch to the correct category; this shows that the output preferences can be quickly reached from the appropriate input representations. The sentence is initially ambiguous but the final three words are very strongly associated with ENergy.

The network in Fig. 11 shows greater activity in its behaviour. The title belongs to the EConomy category, but the individual words of the title are ambiguous; all the words like "Money", "Supply" and "Falls" can also belong

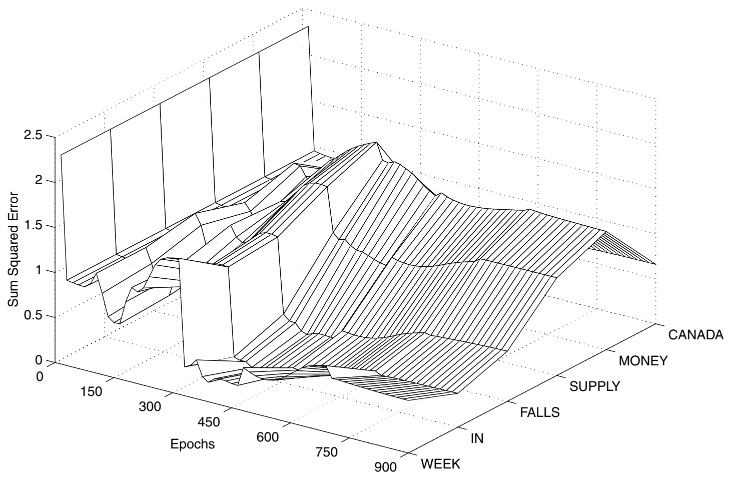

Fig. 11. Error-surface plot for sentence "Canada Money Supply Falls In Week".

Table 7
Recall and precision for classifying newswire titles using the various neural Preference Moore Machines

Evaluation	Recall	Precision
Neural Preference Machine 1 layer training	85.15	86.99
Neural Preference Machine 1 layer test	91.23	90.73
Neural Preference Machine 2 layers training	89.05	90.24
Neural Preference Machine 2 layers test	93.05	92.29

to other categories, and indeed this uncertainty is reflected by the network which does not confidently classify to the appropriate category as shown by the higher value of the error at the end of the title. The performance of the best trained neural Preference Moore Machines is shown in Table 7.

For both examples of neural Preference machines, it can be seen that the recall and precision values for the test sets were higher than those for the training sets, showing that the network was not overfitting the training set data; this again is reflected in the generalization performance and robustness of the network architecture, in that the less common categories would have occurred more frequently in the larger training set as compared to the test set, potentially causing the network to overlearn and hence overfit.

4. Discussion

Two types of different Preference Moore Machine agents were described – firstly, symbolic Preference Moore Machines based on finite-state automata theory which make use of transducers, and secondly, Neural Preference Moore Machines based on the distributed learning of neural networks. It is demonstrated that both approaches, though very different in their computational paradigm, can indeed produce two related modular agents that operate from a heuristically coded top-down mode in the case of the Preference Moore Transducer, and from a bottom-up supervised mode for the Neural Preference Machine. Using the formalism that introduced Preference Moore Machine integration [38], the potential for integrating the different computational approaches on a standard, real-world benchmarking corpus for the task of textual classification and information-mining has been demonstrated.

The symbolic agent is better able to handle exceptions by manually coded expressions, while neural classification agents are able to handle the more difficult and ambiguous semantic sequences, and can be trained using much larger amounts of data. Symbolic transducers are useful to very quickly encode the most relevant knowledge without training but are limited due to the manual coding that is done using a smaller amount of data. However, by using such a data-driven approach and neural preference machines, a much better performance can be reached – 93% versus 73% for the recall, and 92% versus 37% for the precision, for the neural and symbolic preference machines respectively. Transducers are more easily constructed and analyzed, and they are much faster for classification tasks. Neural preference machines have properties such as being adaptable and dynamic; their fault tolerant and robust nature allows them to handle noisy and incomplete data due to the distributed nature of the information contained in their architecture; in contrast, the preference transducer would not necessarily be able to handle new or faulty sequences, thus performance and precision would drop.

However, neural preference machines have learning times that are long, and can learn representations that may be difficult to interpret. By contrast, symbolic transducers have a well understood formalism that describes them, and this allows those systems to be better understood. Nevertheless, it has been demonstrated that the neural preference machines show much better performance than the symbolic transducers.

References

[1] N.M. Allinson, H. Yin, Interactive and semantic data visualisation using self-organizing maps, in: Proceedings of the IEE Colloquium on Neural Networks in Interactive Multimedia Systems, 1998.

[2] M. Balabanovic, Y. Shoham, Learning information retrieval agents: experiments with automated web browsing, in: Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, Stanford, CA, 1995.

[3] M. Balabanovic, Y. Shoham, Y. Yun, An adaptive agent for automated web browsing, Technical Report CS-TN-97-52, Stanford University, 1997.

[4] T. Briscoe, Co-evolution of language and of the language acquisition device, in: Proceedings of the Meeting of the Association for Computational Linguistics, 1997.

[5] E. Charniak, Statistical Language Learning, MIT Press, Cambridge, MA, 1993.

[6] A. Cleeremans, D. Servan-Schreiber, J. McClelland, Finite-state automata and simple recurrent networks, Neural Computation 1 (1989) 372–381.

[7] R. Cooley, B. Mobasher, J. Srivastava, Web mining: information and pattern discovery on the world wide web, in: International Conference on Tools for Artificial Intelligence, Newport Beach, CA, November 1997.

[8] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, S. Slattery, Learning to extract symbolic knowledge from the world wide web, in: Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, 1998.

[9] H. Cunningham, Y. Wilks, R. Gaizauskas, New methods, current trends and software infrastructure for NLP, in: Proceedings of the NEMLAP-2, Ankara, 1996.

[10] J.L. Elman, Finding structure in time, Technical Report CRL 8901, University of California, San Diego, CA, 1988.

[11] D. Freitag, Information extraction from html: application of a general machine learning approach, in: National Conference on Artificial Intelligence, Madison, WI, 1998, pp. 517–523.

[12] C. Lee Giles, B.G. Horne, T. Lin, Learning a class of large finite state machines with a recurrent neural network, Technical Report UMIACS-TR-94-94, NEC Research Institute, Princeton, NJ, August 1994.

[13] T. Honkela, Self-organizing maps in symbol processing, in: S. Wermter, R. Sun (Eds.), Hybrid Neural Systems, Springer, Heidelberg, Germany, 2000.

[14] T. Joachims, Text categorization with support vector machines: learning with many relevant features, in: Proceedings of the European Conference on Machine Learning, Chemnitz, Germany, 1998.

[15] M.I. Jordan, Attractor dynamics and parallelism in a connectionist sequential machine, in: Proceedings of the Eighth Conference of the Cognitive Science Society, Amherst, MA, 1986, pp. 531–546.

[16] S. Kaski, T. Honkela, K. Lagus, T. Kohonen, WEBSOM – self-organizing maps of document collections, Neurocomputing 21 (1998) 101–117.

[17] T. Kohonen, Self-Organizing Maps, Springer, Berlin, 1995.

[18] T. Kohonen, Self-organisation of very large document collections: state of the art, in: Proceedings of the International Conference on Aritificial Neural Networks, Skovde, Sweden, 1998, pp. 65–74.

[19] S.C. Kremer, On the computational power of Elman-style recurrent networks, IEEE Transactions on Neural Networks 6 (4) (1995) 1000–1004.

[20] Yann le Cun, Une procédure d'apprentissage pour réseau à seuil assymétrique, in: Cognitiva 85: A la Frontière de l'Intelligence Artificielle des Sciences de la Connaissance des Neurosciences, Paris, CESTA, 1985, pp. 599–604.

[21] D.D. Lewis, Reuters-21578 text categorization test collection, 1997. Available from http://www.research.att.com/~lewis.

[22] F. Menczer, R. Belew, W. Willuhn, Artificial life applied to adaptive information agents, in: Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, 1995.

[23] K. Niki, Self-organizing information retrieval system on the web: SirWeb, in: N. Kasabov, R. Kozma, K. Ko, R. O'Shea, G. Coghill, T. Gedeon (Eds.), Progress in Connectionist-Based Information Systems. Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Information Systems, vol. 2, Springer, Singapore, 1997, pp. 881–884.

[24] C.W. Omlin, C. Lee Giles, Constructing deterministic finite-state automata in recurrent neural networks, Technical Report 94-3, Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY 12180, 1994.

[25] R. Papka, J.P. Callan, A.G. Barto, Text-based information retrieval using exponentiated gradient descent, in: M.C. Mozer, M.I. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems, vol. 9, MIT Press, Cambridge, MA, 1997.

[26] D.B. Parker, Learning-logic, Technical Report TR-47, Sloan School of Management, MIT, Cambridge, MA, 1985.

[27] M. Perkowitz, O. Etzioni, Adaptive web sites: an AI challenge, in: International Joint Conference on Artificial Intelligence, Nagoya, Japan, 1997.

[28] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. McClelland (Eds.), Parallel Distributed Processing, vol. 1, MIT Press, Cambridge, MA, 1986, pp. 318–362.

[29] M. Sahami, M. Hearst, E. Saund, Applying the multiple cause mixture model to text categorization, Technical Report, AAAI Spring Symposium on Machine Learning in Information Access, 1996.

[30] G. Salton, Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer, Addison-Wesley, Reading, MA, 1989.

[31] H. Schuetze, D.A. Hull, J.O. Pedersen, A comparison of classifiers and document representations for the routing problem, in: Proceedings of the Special Interest Group on Information Retrieval, 1995.

[32] D. Servan-Schreiber, A. Cleeremans, J.L. McClelland, Encoding sequential structure in simple recurrent networks, Technical Report CMU-CS-88-183, Carnegie Mellon University, Pittsburgh, PA, 1988.

[33] N. Sharkey, A. Sharkey, Separating learning and representation, in: S. Wermter, E. Riloff, G. Scheler (Eds.), Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing, Springer, Berlin, 1996, pp. 17–32.

[34] R. Sun, T. Peterson, Multi-agent reinforcement learning: weighting and partitioning, Neural Networks (1999).

[35] G. van Noord, FSA utilities: a toolbox to manipulate finite-state automata, in: D. Raymond, D. Wood, S. Yu (Eds.), Automata Implementation, Lecture Notes in Computer Science, vol. 1260, Springer, New York, 1997, pp. 87–108.

[36] P.J. Werbos, Beyond regression: new tools for regression and analysis in the behavioral sciences, Ph.D. Thesis, Harvard University, Division of Engineering and Applied Physics, 1974.

[37] S. Wermter, Hybrid Connectionist Natural Language Processing, Chapman & Hall, Thomson International, London, UK, 1995.

[38] S. Wermter, Preference Moore machines for neural fuzzy integration, in: Proceedings of the International Joint Conference on Artificial Intelligence, Stockholm, 1999, pp. 840–845.

[39] S. Wermter, Neural fuzzy preference integration using neural preference moore machines, International Journal of Neural Systems 10 (4) (2000) 287–309.

[40] S. Wermter, G. Arevian, C. Panchev, Recurrent neural network learning for text routing, in: Proceedings of the International Conference on Artificial Neural Networks, Edinburgh, UK, 1999, pp. 898–903.

[41] S. Wermter, G. Arevian, C. Panchev, Network analysis in a neural learning internet agent, in: Proceedings of the International Conference on Computational Intelligence and Neurosciences, Atlantic City, PA, USA, 2000, pp. 880–884.

[42] S. Wermter, C. Panchev, G. Arevian, Hybrid neural plausibility networks for news agents, in: Proceedings of the National Conference on Artificial Intelligence, Orlando, USA, 1999, pp. 93–98.

[43] S. Wermter, R. Sun, Hybrid Neural Systems, Springer, Heidelberg, 2000.