# Towards Hybrid Neural Learning Internet Agents

Stefan Wermter, Garen Arevian and Christo Panchev

Hybrid Intelligent Systems Group
University of Sunderland, Centre for Informatics, SCET
St Peter's Way, Sunderland, SR6 0DD, UK
stefan.wermter@sunderland.ac.uk
http://www.his.sunderland.ac.uk/

**Abstract.** The following chapter explores learning internet agents. In recent years, with the massive increase in the amount of available information on the Internet, a need has arisen for being able to organize and access that data in a meaningful and directed way. Many well-explored techniques from the field of AI and machine learning have been applied in this context. In this paper, special emphasis is placed on neural network approaches in implementing a learning agent. First, various important approaches are summarized. Then, an approach for neural learning internet agents is presented, one that uses recurrent neural networks for the learning of classifying a textual stream of information. Experimental results are presented showing that a neural network model based on a recurrent plausibility network can act as a scalable, robust and useful news routing agent.

## 1 Introduction

The exponential expansion of Internet information has been very apparent; however, there is still a great deal that can be done in terms of improving the classification and subsequent access of the data that is potentially available. The motivation for trying various techniques from the field of machine learning arises from the fact that there is a great deal of unstructured data. Much time is spent on searching for information, filtering information down to essential data, reducing the search space for specific domains, classifying text and so on. The various techniques of machine learning are examined for automating the learning of these processes, and tested to address the problem of an expanding and dynamic Internet [26].

So-called "internet agents" are implemented to address some of these problems. The simplest definition of an agent is that it is a software system, to some degree autonomous, that is designed to perform or learn a specific task [2, 30] which is either one algorithm or a combination of several. Agents can be designed to perform various tasks including textual classification [34, 10], information retrieval and extraction [5, 9], routing of information such as email, news [3, 18, 6,

46], automating web browsing [1], organization [36, 4], personal assistance [39, 20, 17, 14] and learning for web-agents [28, 46].

In spite of a lot of work on internet agents, most systems currently do not have learning capabilities. In the context of this paper, a learning agent is taken to be an algorithmic approach to a classification problem that allows it to be dynamic, robust and able to handle noisy data, to a degree autonomously, while improving its performance through repeated experience [44]. Of course, learning *internet* agents can have a variety of definitions as well, and the emphasis within this context is more on autonomously functioning systems that can either classify or route information of a textual nature. In particular, after a summary of various approaches, the HyNeT recurrent neural network architecture will be described, which is shown to be a robust and scalable text routing agent for the Internet.

## 2 Different Approaches to Learning in Agents

The field of Machine Learning is concerned with the construction of computer programs that automatically improve their performance with experience [33].

A few examples of currently applied machine learning approaches for learning agents are decision trees [37], Bayesian statistical approaches [31], Kohonen networks [24, 22] and Support Vector Machines (SVMs) [19]. However, in the following summary, the potential use of *neural networks* is examined.

### 2.1 Neural Network Approaches

Many internet-related problems are neither discrete nor are the distributions known due to the dynamics of the medium. Therefore, internet agents can be made more powerful by employing various learning algorithms inspired by approaches from neural networks. Neural networks have several main properties which make them very useful for the Internet. The information processing is non-linear, allowing the learning of real-valued, discrete-valued and vector-valued examples; they are adaptable and dynamic in nature, and hence can cope with a varying operating environment. Contextual information and knowledge is represented by the structure and weights of a system, allowing interesting mappings to be extracted from the problem environment. Most importantly, neural networks are fault-tolerant and robust, being able to learn from noisy or incomplete data due to their distributed representations.

There are many different neural network algorithms; however, while bearing in mind the context of agents and learning, several types of neural network are more suitable than others for the task that is required. For a dynamic system like the Internet, an online agent needs to be as robust as possible, essentially to be left to the task of routing, classifying and organizing textual data in an autonomous and self-maintaining way by being able to generalize, to be fault-tolerant and adaptive. The three approaches so far shown to be most suitable are recurrent networks [46], Kohonen self-organizing maps (SOMs) [24, 22] and reinforcement learning [42, 43]. All these neural network approaches have properties which are briefly discussed and illustrated below.

**Supervised Recurrent Networks** Recurrent neural networks have shown great promise in many tasks. For example, certain natural language processing approaches require that context and time be incorporated as part of the model [8, 7]; hence, recent work has focused on developing networks that are able to create contextual representations of textual data which take into account the implicit representation of time, temporal sequencing and the context as a result of the internal representation that is created. These properties of recurrent neural networks can be useful for creating an agent that is able to derive information from text-based, noisy Internet input. In particular, recurrent plausibility networks have been found useful [45, 46].

Also, NARX (Nonlinear Autoregressive with eXogenous inputs) models have been shown to be very effective in learning many problems such as those that involve long-term dependencies [29]; NARX networks are formalized by [38]:

$$y(t) = f(x(t-n_x), \ldots, x(t-1), x(t), y(t-n_y), \ldots, y(t-1)),$$

where $x(t)$ and $y(t)$ are the input and output of the network at a time $t$; $n_x$ and $n_y$ represent the order of the input and output, and the function $f$ is the mapping performed by the multi-layer perceptron.

In some cases, it has been shown that NARX and RNN (Recurrent Neural Network) models are equivalent [40], and under conditions that the neuron transfer function is similar to the NARX transfer function, one may be transformed to the other and vice versa - the benefit being that if the output dimension of a NARX model is larger than the number of hidden units, training an equivalent RNN will be faster; pruning is also easier in an equivalent NARX whose stability behavior can be analyzed more readily.

**Unsupervised Models** Recently, applications of Kohonen nets have been extended to the realm of text processing [25, 16], to create browsable mappings of Internet-related hypertext data. A self-organizing map (SOM) forms a nonlinear projection from a high-dimensional data manifold onto a low-dimensional grid [24]. The SOM algorithm computes an optimal collection of models that approximates the data by applying a specified error criterion and takes into account the similarities and hence the relations between the models; this allows the ordering of the reduced-dimensionality data onto a grid.

The SOM algorithm [23, 24] is formalized as follows: there is an initialization step, where random values for the initial weight vectors $w_j(0)$ are set; if the total number of neurons in the lattice is $N$, $w_j(0)$ must be different for $j = 1, 2, \ldots, N$. The magnitude for the weights should be kept small for optimal performance. There is a sampling step where example vectors $x$ from the input distribution are taken that represent the sensory signal. The optimally matched 'winning' neuron $i(x)$ at discrete time $t$ is found using the minimum-distance Euclidean criterion by a process called similarity matching:

$$i(x) = arg_j min \parallel x(t) - w_j(t) \parallel \quad for \quad j = 1, 2, \ldots, N$$

The synaptic weight vectors of all the neurons are adjusted and updated, according to:

$$w_j(t+1) = \begin{cases} w_j(t) + \mu(t)[x(t) - w_j(t)] & for \quad j \in \Lambda_{i(x)}(t) \\ w_j(t) & otherwise \end{cases}$$

The learning rate is $\mu(t)$, and $\Lambda_{i(x)}(t)$ is the neighborhood function centered around the winning neuron $i(x)$; both $\mu(t)$ and $\Lambda_{i(x)}(t)$ are continuously varied. The sampling, matching and update are repeated until no further changes are observed in the mappings.

In this way, the WEBSOM agent [25] can represent web documents statistically by their word frequency histograms or some reduced form of the data as vectors. The SOM here is acting as a similarity graph of the data. A simple graphical user interface is used to present the ordered data for navigation. This approach has been shown to be appropriate for the task of learning for newsgroup classification.

**Reinforcement Learning Approaches** This is the on-line learning of input-output mappings through a process of exploration of a problem space. Agents that use reinforcement learning rely on the use of training data that evaluates the final *actions* taken. There is active exploration with an explicit trial-and-error search for the desired behavior [43, 12]; evaluative feedback, specifically characteristic of this type of learning, indicates how good an action taken is, but not if it is the best or worst. All reinforcement algorithm approaches have explicit goals, interact with and influence their environments.

Reinforcement learning aims to find a *policy* that selects a sequence of actions which are statistically optimal. The probability that a specific environment makes a transition from a state $x(t)$ to $y$ at a time $t+1$, given that it was previously in states $x(0), x(1), ...$, and that the corresponding actions $a(0), a(1), ...$, were taken, depend entirely on the current state $x(t)$ and action $a(t)$ as shown by:

$$\Gamma\{x(t+1) = y | x(0), a(0); x(1), a(1); \ldots; x(t), a(t)\}$$

$$= \Gamma\{x(t+1) = y | x(t), a(t)\}$$

where $\Gamma(\cdot)$ is the transition probability or change of state.

If the environment is in a state $x(0) = x$, the evaluation function [43, 12] is given by:

$$H(x) = E\left[\sum_{k=0}^{\infty} \gamma^k r(k+1) | x(0) = x\right]$$

Here, $E$ is the *expectation operator*, taken with respect to the policy used to select actions by the agent. The summation is termed the *cumulative discounted*

*reinforcement*, and $r(k+1)$ is the reinforcement received from the environment after action $a(k)$ is taken by the agent. The reinforcement feedback can have a positive value (regarded as a 'reward' signal), a negative value (regarded as 'punishment') or unchanged; $\gamma$ is called the *discount-rate parameter* and lies in the range $0 \leq \gamma < 1$, where if $\gamma \rightarrow 0$, then the reinforcement is more short term, and if $\gamma \rightarrow 1$, then the cumulative actions are for the longer term. Learning the evaluation function $H(x)$ allows the use of the cumulative discounted reinforcement later on.

This approach, though not fully explored for sequential tasks on the Internet, holds promise for the design of a learning agent system that fulfills the necessary criteria - one that is autonomous, able to adapt, robust, can handle noise and sequential decisions.

## 3 Analysis and Discussion of a Specific Learning Internet Agent: HyNeT

A more detailed description of one particular learning agent will now be presented. A great deal of recent work on neural networks has shifted from the processing of strictly numerical data towards the processing of various corpora and the huge body of the Internet [35, 26, 5, 19]. Indeed, it has been an important goal to study the more fundamental issues of connectionist systems, and the way in which knowledge is encoded in neural networks and how knowledge can be derived from them [13, 32, 11, 41, 15]. A useful example, applicable as it is a real-world task, is the routing and classification of newswire titles and will now be described.

### 3.1 Recurrent Plausibility Networks

In this section, a detailed analysis of one such agent called HyNeT (**Hy**brid **Ne**ural/symbolic agents for **T**ext routing on the internet), which uses a recurrent neural network, is presented and experimental results are discussed.

The specific neural network explored here is a more developed version of the simple recurrent neural network, namely a Recurrent Plausibility Network [45, 46]. Recurrent neural networks are able to map both previous internal states and input to a desired output - essentially acting as short-term incremental memories that take time and context into consideration.

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. However, partially recurrent networks have recurrent connections between the hidden and context layer [7] or Jordan networks have connections between the output and context layer [21]; these allow previous states to be kept within the network structure.

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, the information of which is allowed to decay more slowly to keep information about previous events over a longer time-period. In other research [45], different decay memories were introduced by using distributed recurrent delays over the separate context layers representing the contexts at different time steps. 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. Figure 1 shows the general structure of our recurrent plausibility network.

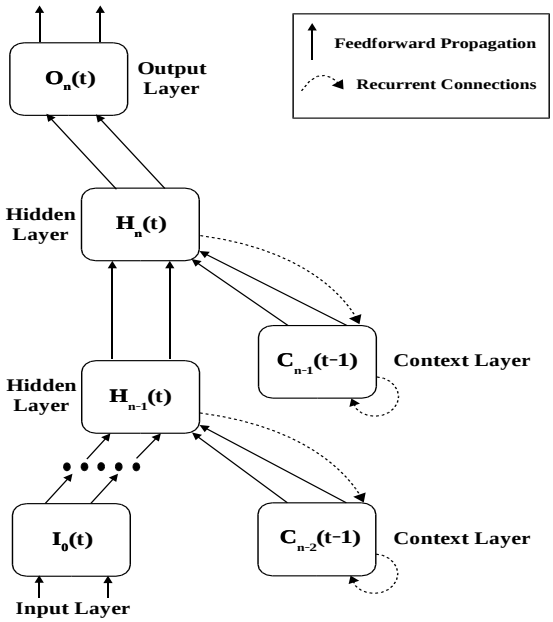


**Fig. 1.** General Representation of a Recurrent Plausibility Network.

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 a particular case, the following is used:

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

The units in the two context layers with one time a 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 of the context is controlled by the hysteresis value $\varphi_n$. 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.

### 3.2 Reuters-21578 Text Categorization Test Collection

The Reuters News Corpus is a collection of news articles that appeared on the Reuters Newswire; all the documents have been categorized by Reuters into several specific categories. Further formatting of the corpus [27] has produced the so-called ModApte Split; some examples of the news titles are given in Table 1.

| Semantic Category | Example Titles |
|---|---|
| money-fx | Bundesbank sets new re-purchase tender |
| shipping | US Navy said increasing presence near gulf |
| interest | Bank of Japan determined to keep easy money policy |
| economic | Miyazawa sees eventual lower US trade deficit |
| corporate | Oxford Financial buys Clancy Systems |
| commodity | Cattle being placed on feed lighter than normal |
| energy | Malaysia to cut oil output further traders say |
| shipping & energy | Soviet tankers set to carry Kuwaiti oil |
| money-fx & currency | Bank of Japan intervenes shortly after Tokyo opens |

**Table 1.** Example titles from the Reuters corpus.

All the news titles belong to one or more of eight main categories: Money and Foreign Exchange (**money-fx**, **MFX**), Shipping (**ship**, **SHP**), Interest Rates (**interest**, **INT**), Economic Indicators (**economic**, **ECN**), Currency (**currency**, **CRC**), Corporate (**corporate**, **CRP**), Commodity (**commodity**, **CMD**), Energy (**energy**, **ENG**).

### 3.3 Various Experiments Conducted

In order to get a comparison of performance, several experiments were conducted using different vector representations of the words in the Reuters corpus as

part of the preprocessing; the variously derived vector representations were fed into the input layer of simple recurrent networks, the output being the desired semantic routing category. The preprocessing strategies are briefly outlined and explained below. The recall/precision results are presented later in Table 2 for each experiment.

**Simple Recurrent Network and Significance Vectors** In the initial experiment, words were represented using significance vectors; these were obtained by determining the frequency of a word in different semantic categories using the following operation:

$$v(w, x_i) = \frac{Frequency\ of\ w\ in\ x_i}{\sum_j Frequency\ of\ w\ in\ x_j}\ for\ j \in \{1, \cdots n\}$$

If a *vector* $(x_1 x_2 \ldots x_n)$ represents each word $w$, and $x_i$ is a specific semantic category, then $v(w, x_i)$ is calculated for each dimension of the word vector, as the frequency of a word $w$ in the different semantic categories $x_i$ divided by the number of times the word $w$ appears in the corpus. The computed values are then presented at the input of a simple recurrent network [8] in the form $(v(w, x_1), v(w, x_2), \ldots, v(w, x_n))$.

**Simple Recurrent Network and Semantic Vectors** An alternative preprocessing strategy was to represent vectors as the *plausibility* of a specific word occurring in a particular semantic category, the main advantage being that they are independent of the number of examples present in each category:

$$v(w, x_i) = \frac{Normalized\ frequency\ of\ w\ in\ x_i}{\sum_j Normalized\ frequency\ of\ w\ in\ x_j},\ j \in \{1, \cdots n\}$$

where:

$$Normalized\ frequency\ of\ w\ in\ x_i = \frac{Frequency\ of\ w\ in\ x_i}{Number\ of\ titles\ in\ x_i}$$

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

**Recurrent Plausibility Network and Semantic Vectors** In the final experiment, a recurrent plausibility network, as shown in Figure 1 was used; the actual architecture used for the experiment was one with two hidden and two context layers. After empirically testing various combinations of settings for the values of the hysteresis value for the activation function of the context layers, it was found that the network performed optimally with a value of 0.2 for the first context layer, and 0.8 for the second.

| Type of Vector Representation Used in Experiment | Training set | | Test set | |
|---|---|---|---|---|
| | recall | precision | recall | precision |
| Significance Vectors and Simple Recurrent Network | 85.15 | 86.99 | **91.23** | **90.73** |
| Semantic Vectors and Simple Recurrent Network | 88.57 | 88.59 | **92.47** | **91.61** |
| Semantic Vectors with Recurrent Plausibility Network | 89.05 | 90.24 | **93.05** | **92.29** |
| "Bag of Words" with Recurrent Plausibility Network | - | - | **86.60** | **83.10** |

**Table 2.** Best recall/precision results from various experiments

### 3.4 Results of Experiments

The results in Table 2 show the clear improvement in the overall recall/precision values from the first experiment using the significance vectors, to the last using the plausibility network. The experiment with the semantic vector representation showed an improvement over the first. The best performance was shown by the use of the plausibility network.

In comparison, a bag-of-words approach, to test performance on sequences without order, reached 86.6% recall and 83.1% precision; this indicates that the order of significant words and hence the context are important as a source of information which the recurrent neural network learns, allowing better classification performance.

These results demonstrate that a carefully developed neural network agent architecture can deal with significantly large test and training sets. In some previous work [45], recall/precision accuracies of 95% were reached but the library titles used in the work were much less ambiguous than the Reuters Corpus (which had a few main categories and newstitles that could easily be misclassified due to the inherent ambiguity) and only 1 000 test titles were used in the approach while the plausibility network was scalable to 10 000 corrupted and ambiguous titles.

For general comparison with other approaches, interesting work on text categorization on the Reuters corpus has been done using whole documents [19] rather than titles. Taking the ten most frequently occurring categories, it has been shown that the recall/precision break-even point for Support Vector Machines was 86%, 82% for k-Nearest Neighbor, 72% for Naive Bayes. Though a different set of categories and whole documents were used, and therefore the results may not be directly comparable to results shown in Table 2, they do however give some indication of document classification performance on this corpus. Especially for medium text data sets or when only titles are available, the HyNeT agent compares favorably with the other machine learning techniques that have been tested on this corpus.

### 3.5 Analysis of the Output Representations

For a clear presentation of the network's behavior, the results are illustrated and analyzed below; the error surfaces show plots of the sum-squared error of

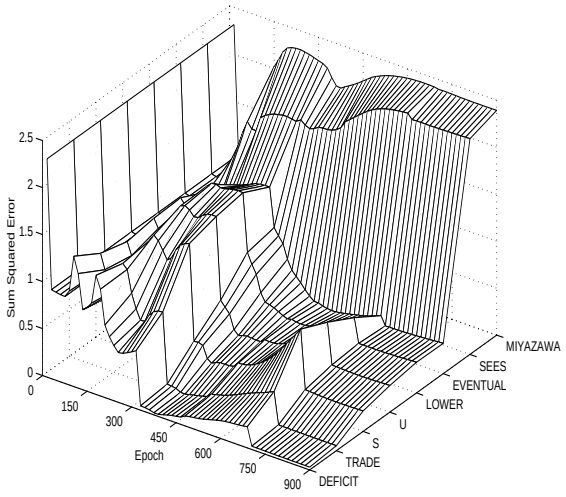


**Fig. 2.** The error surface of the title "Miyazawa Sees Eventual Lower US Trade Deficit"

the output preferences, plotted against the number of training epochs and each word of a title.

Figure 2 shows the surface error of the title "Miyazawa Sees Eventual Lower US Trade Deficit". In the Reuters Corpus this is classified under the "economic" category; as can be seen, the network does learn the correct category classification. The first two words, "Miyazawa" and "sees", are initially given several possible preferences to other categories and the errors are high early on in the training. However, the subsequent words "eventual", "lower", etc. cause the network to increasingly favor the correct classification, and at the end, the trained network has a very strong preference (shown by the low error value) for the incremental context of the desired category.

The second example is shown in Figure 3, titled "Bank of Japan Determined To Keep Easy Money Policy" and belonging to the "interest" category. This example shows a more complicated behavior in the contextual learning, in contrast to the previous one. The words beginning "Bank of Japan" are ambiguous and could be classified under different categories such as "money/foreign exchange" and "currency", and indeed the network shows some confused behavior; again however, the context of the latter words such as "easy money policy" eventually allow the network to learn the correct classification.

### 3.6 Context Building in Plausibility Neural Networks

Figures 5 and 7 present cluster dendrograms based on the internal context representations at the end of titles. The test includes 5 representative titles for each

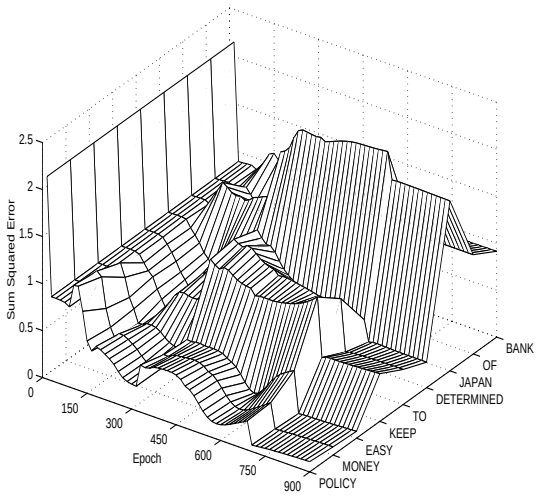


**Fig. 3.** The error surface of the title "Bank of Japan Determined To Keep Easy Money Policy"

category; each title belongs to only one category. All titles are correctly classified by the network. The first observation that can be made from these figures is that the dendrogram based on the activations of the second context layer (closer to the output layer) provides a better distinction between the classes. In other words, it can be seen that the second context layer is more representative of the title classification than the first one. This analysis aims to explore how these contexts are built and what the difference is between the two contexts along a title.

Using the data for Figures 5 and 7, the *class-activity* of a particular context unit is defined with respect to a given category as the activation of this unit when a title from this category has been presented to the network. That is, for example, at the end of a title from the category "economic", the units with the higher activation will be classified as being *more class-active* with respect to the "economic" category, and the units with lower activation as *less class-active*.

For the analysis of the context building in the plausibility network, the activation of the context units were taken while processing the title "Assets of money market mutual funds fell 35.3 mln dlrs in latest week to 237.43 billion". This title belongs to the "economic" category and the data was sorted with a key which is the activity of the neurons with respect to this category. The results are shown in Figures 4 and 6.

The most *class-active* unit for the class "economic" is given as unit 1 in the figure, and the lowest class-activity as unit 6. Thus, the ideal curve at a given word step for the title to be classified to the correct category will be a

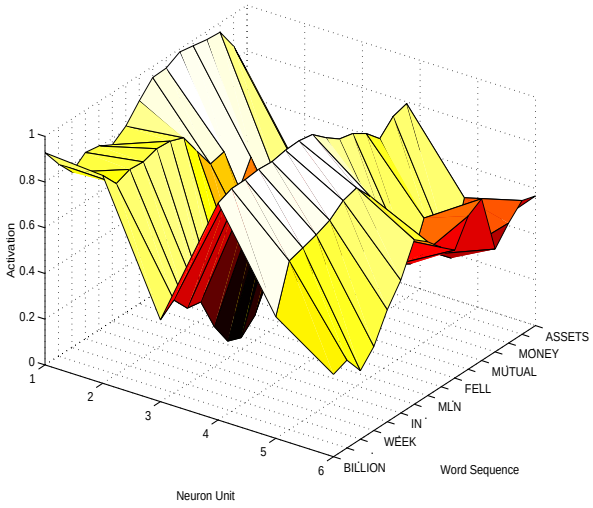


**Fig. 4.** The activation of the units in the first context layer. The order of the units is changed according to the class-activity

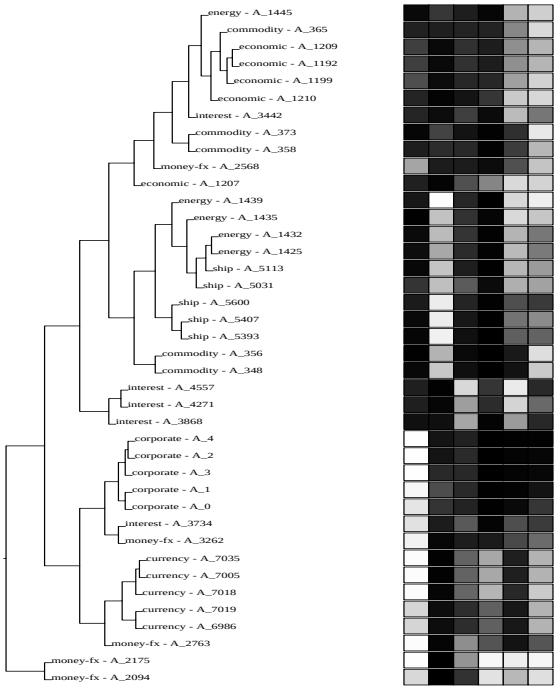


**Fig. 5.** The cluster dendrogram and internal context representations of the first context layer for 40 representative titles.

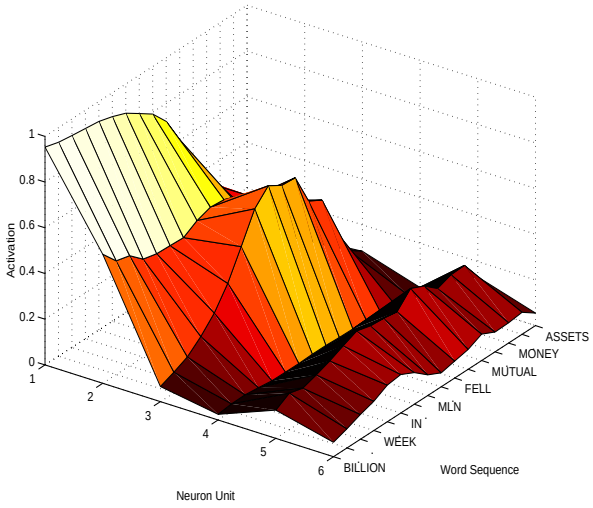


**Fig. 6.** The activation of the units in the second context layer. The order of the units is changed according to the class-activity

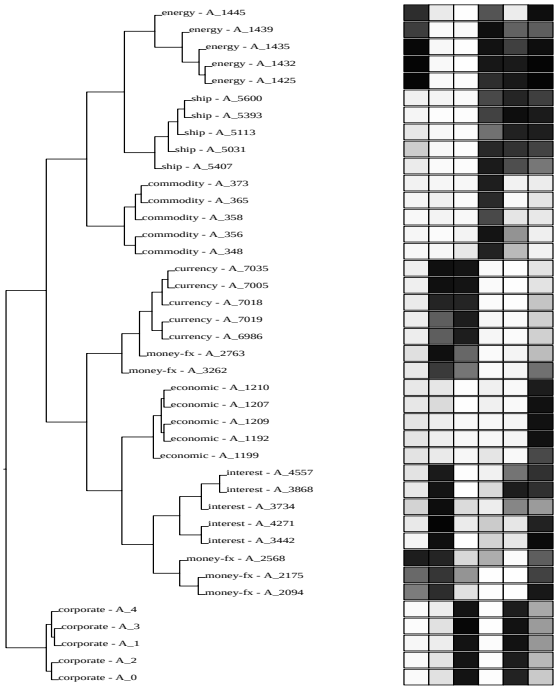


**Fig. 7.** The cluster dendrogram and internal context representations of the second context layer for 40 representative titles.

monotonically decreasing function starting from the units with the highest class-activity to the units with lower class-activity. As can be seen, most of the units in the first context layer (closer to the input) are more dynamic. They are highly dependent on the current word. Therefore the first context layer does not build a representative context for the required category at the end of the title. It rather responds to the incoming words, building a short dynamic context. However, the second context layer is incrementally building its context representation for the particular category. It is the context layer which is most responsible for a stable output and does not fluctuate so much with the different incoming words.

## 4 Conclusions

A variety of neural network learning techniques were presented which are considered relevant to the specific problem of classification on Internet texts. A new recurrent network architecture, HyNeT, was presented that is able to route news headlines. Similar to incremental language processing, plausibility networks also process news titles using previous context as extra information. At the beginning of a title, the network might predict an incorrect category which usually changes to the correct one later on when more contextual information is available.

Furthermore, the error of the network was also carefully examined at each epoch and for each word of the training headlines. These surface error figures allow a clear, comprehensive evaluation of training time, word sequence and overall classification error. In addition, this approach may be quite useful for any other learning technique involving sequences. Then, an analysis of the context layers was presented showing that the layers do indeed learn to use the information derived from context.

To date, recurrent neural networks have not been developed for a new task of such size and scale, in the design of title routing agents. HyNeT is robust, classifies noisy arbitrary real-world titles, processes titles incrementally from left to right, and shows better classification reliability towards the end of titles based on the learned context. Plausibility neural network architectures hold a lot of potential for building robust neural architectures for semantic news routing agents on the Internet.

## References


1. M. Balabanovic and 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.
2. M. Balabanovic, Y. Shoham, and Y. Yun. An adaptive agent for automated web browsing. Technical Report CS-TN-97-52, Stanford University, 1997.
3. W. Cohen. Learning rules that classify e-mail. In *AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA, 1996.

4. R. Cooley, B. Mobasher, and 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.
5. M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and 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.
6. P. Edwards, D. Bayer, C.L. Green, and T.R. Payne. Experience with learning agents which manage internet-based information. In *AAAI Spring Symposium on Machine Learning in Information Access*, pages 31–40, Stanford, CA, 1996.
7. J. L. Elman. Finding structure in time. Technical Report CRL 8901, University of California, San Diego, CA, 1988.
8. J. L. Elman. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7:195–226, 1991.
9. D. Freitag. Information extraction from html: Application of a general machine learning approach. In *National Conference on Artificial Intelligence*, pages 517–523, Madison, Wisconsin, 1998.
10. J. Fuernkranz, T. Mitchell, and E. Riloff. A case study in using linguistic phrases for text categorization on the WWW. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorisation*, Madison, WI, 1998.
11. L. Giles and C. W. Omlin. Extraction, insertion and refinement of symbolic rules in dynamically driven recurrent neural networks. *Connection Science*, 5:307–337, 1993.
12. S. Haykin. *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company, New York, 1994.
13. J. Hendler. Developing hybrid symbolic/connectionist models. In J. A. Barnden and J. B. Pollack, editors, *Advances in Connectionist and Neural Computation Theory, Vol.1: High Level Connectionist Models*, pages 165–179. Ablex Publishing Corporation, Norwood, NJ, 1991.
14. R. Holte and C. Drummond. A learning apprentice for browsing. In *AAAI Spring Symposium on Software Agents*, Stanford, CA, 1994.
15. V. Honavar. Symbolic artificial intelligence and numeric artificial neural networks: towards a resolution of the dichotomy. In R. Sun and L. A. Bookman, editors, *Computational Architectures integrating Neural and Symbolic Processes*, pages 351–388. Kluwer, Boston, 1995.
16. S. Honkela. Self-organizing maps in symbol processing. In *Hybrid Neural Systems (this volume)*. Springer-Verlag, 2000.
17. M.A. Hoyle and C. Lueg. Open SESAME: A look at personal assisitants. In *Proceedings of the Interanational Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology, London*, pages pp. 51–56, 1997.
18. D. Hull, J. Pedersen, and H. Schutze. Document routing as statistical classification. In *AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA, 1996.
19. 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.
20. T. Joachims, D. Freitag, and T. Mitchell. Webwatcher: A tour guide for the world wide web. In *Fifteenth International Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997.

21. M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Conference of the Cognitive Science Society*, pages 531–546, Amherst, MA, 1986.
22. S. Kaski, T. Honkela, K. Lagus, and T. Kohonen. WEBSOM - self-organizing maps of document collections. *Neurocomputing*, 21:101–117, 1998.
23. T. Kohonen. *Self-Organization and Associative Memory*. Springer, Berlin, third edition, 1989.
24. T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 1995.
25. T. Kohonen. Self-organisation of very large document collections: State of the art. In *Proceedings of the International Conference on Aritificial Neural Networks*, pages 65–74, Skovde, Sweden, 1998.
26. S. Lawrence and C. L. Giles. Searching the world wide web. *Science*, 280:98–100, 1998.
27. D. D. Lewis. Reuters-21578 text categorization test collection, 1997. http://www.research.att.com/~lewis.
28. R. Liere and P. Tadepalli. The use of active learning in text categorisation. In *AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA, 1996.
29. T. Lin, B. G. Horne, P. Tino, and C. L. Giles. Learning long-term dependencies in NARX recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, November 1996.
30. F. Menczer, R. Belew, and 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.
31. D. Michie, D. J. Spiegelhalter, and C. C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, New York, 1994.
32. R. Miikkulainen. *Subsymbolic Natural Language Processing*. MIT Press, Cambridge, MA, 1993.
33. T. M. Mitchell. *Machine Learning*. WCB/McGraw-Hill, New York, 1997.
34. K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of the National Conference on Artificial Intelligence*, Madison, WI, 1998.
35. R. Papka, J. P. Callan, and A. G. Barto. Text-based information retrieval using exponentiated gradient descent. In *Advances in Neural Information Processing Systems*, volume 9, Denver, CO, 1997. MIT Press.
36. M. Perkowitz and O. Etzioni. Adaptive web sites: an AI challenge. In *International Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997.
37. J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.
38. H. T. Siegelmann, B. G. Horne, and C. L. Giles. Computational capabilities of recurrent NARX neural networks. Technical Report CS-TR-3408, University of Maryland, College Park, 1995.
39. M. Spiliopoulou, L. C. Faulstich, and K. Winkler. A data miner analyzing the navigational behavior of web users. In *ACAI-99 Workshop on Machine Learning in User Modeling*, Crete, July 1999.
40. J.P.F. Sum, W.K. Kan, and G.H. Young. A note on the equivalence of NARX and RNN. *Neural Computing and Applications*, 8:33–39, 1999.
41. R. Sun. *Integrating Rules and Connectionism for Robust Commonsense Reasoning*. Wiley, New York, 1994.
42. R. Sun and T. Peterson. Multi-agent reinforcement learning: Weighting and partitioning. *Neural Networks*, 1999.

43. R. S. Sutton and A. G. Barto. *Reinforcement Learning: an Introduction*. MIT Press, Cambridge, MA, 1998.
44. G. Tecuci. *Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies*. Academic Press, San Diego, 1998.
45. S. Wermter. *Hybrid Connectionist Natural Language Processing*. Chapman and Hall, Thomson International, London, UK, 1995.
46. S. Wermter, C. Panchev, and G. Arevian. Hybrid neural plausibility networks for news agents. In *Proceedings of the National Conference on Artificial Intelligence*, pages 93–98, Orlando, USA, 1999.