Natural Language Learning by Recurrent Neural Networks: A Comparison with probabilistic approaches

Michael Towsey*, Joachim Diederich*, Ingo Schellhammer*#, Stephan Chalup*, Claudia Brugman**

* Neurocomputing Research Centre, Queensland University of Technology, QLD, 4001, Australia
# Dept of Information Systems, University of Muenster, D-48149 Muenster, Germany
**School of Languages, University of Otago, Dunedin, New Zealand
joachim@fit.qut.edu.au

Abstract

We present preliminary results of experiments with two types of recurrent neural networks for a natural language learning task. The neural networks, Elman networks and Recurrent Cascade Correlation (RCC), were trained on the text of a first-year primary school reader. The networks performed a one-step-look-ahead task, i.e. they had to predict the lexical category of the next following word. Elman networks with 9 hidden units gave the best training results (72% correct) but scored only 63% when tested for generalisation using a "leave-one-sentence-out" cross-validation technique. An RCC network could learn 99.6% of the training set by adding up to 42 hidden units but achieved best generalisation (63%) with only four hidden units. Results are presented showing network learning in relation to bi-, tri-, 4- and 5-gram performance. Greatest prediction uncertainty (measured as the entropy of the output units) occurred, not at the sentence boundaries but when the first verb was the input.

1. Introduction

Elman (1990) emphasised that natural language input unfolds in time and therefore, recurrent networks which can accept a sequence of input patterns are the preferred choice for many connectionist natural language processing tasks. In recurrent networks, knowledge is represented in activation patterns over hidden units and revealed (i.e. made explicit) by hierarchical cluster analysis or other statistical methods. Furthermore, recent evidence from cognitive neuroscience (Singer, 1995) points to the importance of recurrent connections for the formation of coherent cell assemblies.

Recent work on recurrent neural networks has focussed on formal languages (Wiles & Elman, 1995). In this paper, we present preliminary results of experiments with recurrent neural networks for a natural language learning task. Our strategy is to start with simple children's texts and to step-wise increase the complexity of these texts to explore the learning characteristics of recurrent neural networks. In the first experiments reported here, we are starting with a first-year primary school reader from which sentences with embedded structures have been eliminated. In future experiments, we will use unmodified first-year texts and will continue with second-year textbooks and so on.

1.1 Elman and RCC networks

Simple recurrent networks (SRN's) of the Elman type are similar to three-layer perceptrons but with recurrent connections from the hidden layer to a *context layer* (also called *state layer*) which becomes part of the input. The activation patterns of the hidden units at time step *t* are copied onto the context units and presented with the input at the next time step. The hidden units have the task of mapping both an external input, and also the previous internal state in order to produce some desired output. Thus, the internal representations that develop are sensitive to temporal context; the effect of time is implicit in these internal states (Elman, 1990).

Finding the optimum hidden layer size for an Elman network is a matter of trial and error and can be time consuming. One approach to finding the optimum hidden layer size is to train an RCC net which implements an incremental learning algorithm. The size of the generated RCC network gives an indication of a reasonable size for a SRN. It is important however to distinguish between the ability of an RCC network to learn the training set and the generalisation of the resulting network.

Fahlman (1991) introduced the RCC architecture. Instead of only adjusting the weights of a fixed network topology, the idea is to start with a minimal network and to add hidden units as necessary. The initial network starts with no hidden units, and only the weights to the outputs are trained. If the resulting performance is not satisfactory, a new hidden unit has to be added.

The learning and network construction process works as follows: The algorithm starts with a set of new units, called the candidate pool. These units have randomly initialized, weighted connections from the input nodes and all the hidden units already present in the network. At that time their outputs are unconnected. Then, their incoming links are trained on the training set and adjusted in order to maximize the correlation between the candidates' outputs and the remaining error (Fahlman, 1991). When there is no improvement, this process stops and the candidate with the best 'correlation score' is added permanently. These weights are then frozen, which means that this node becomes a new feature detector in the network. In order to integrate

Michael Towsey, Joachim Diederich, Ingo Schellhammer, Stephan Chalup and Claudia Brugman (1998) **Natural Language Learning by Recurrent Neural Networks: a comparaison with probabilistic approaches.** In D.M.W. Powers (ed.) *NeMLaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning,* ACL, pp 3-10.

this new hidden unit, the weights of the output layer are trained for multiple epochs (Fahlman, 1991). This scheme of adding hidden units and retraining the output layer weights is repeated until success (when the error falls below a predefined threshold or no bit errors are found) or a maximum number of hidden units is reached (which means failure). Fahlman points out that this learning algorithm is a 'greedy' algorithm because the network is extended stepwise until a solution is found.

Note that every new hidden unit adds a new hidden layer to the network, because it is connected to all previous hidden units. Recurrence is achieved by adding self-recurrent links to all hidden units. So at time step t+1 the activation of time step t is fed back to the hidden unit. Such a constructive learning algorithm is expected to build a near minimal network for a given problem, because every hidden unit serves as a unique feature detector, and no *a priori* and maybe inappropriate guesses about the size of hidden layers have to be made.

1.2. Overview

In the remainder of this paper, we present results for Elman networks which were evaluated by use of cross-validation. We elucidate the Elman network's learning phases and compare its performance with RCC networks. N-gram scores are used as benchmarks. We examine the Elman network's uncertainty regarding the prediction of the next lexical category and also the sequences it has correctly learned.

2. Methods

2.1. The data

The natural language corpus used in these experiments was obtained from a first-year primary school reader published circa 1950's (Hume). This text was chosen because of its limited vocabulary and sentence structure. For this initial study, sentences with embedded structures (relative clauses) and a length of more than eight words were eliminated. The resulting corpus consisted of 106 sentences ranging from three to eight words in length, average length 5.06 words. The words were converted to 10 lexical categories, including a sentence boundary marker. The categories and their abbreviations as used in the subsequent text and figures are listed in Table 1.

The resulting data consisted of a string of 643 categories in 106 sentences. There were 62 distinct sentence sequences of which 43 occurred only once, the rest being replicated. The maximum replication of any sequence was eight-fold. Category frequencies are given in Table 1.

For some experiments, the total data were used for training and in other experiments the data were divided into training and test sets. The test set consisted of every fourth sentence taken from the total data yielding a string of 158 categories in 26 sentences. The training set consisted of the remaining data, a string of 486 categories in 80 sentences. Due to replication of some sentences, the test set contained sentence sequences that also occurred in the training set.

TABLE 1: Percent frequencies of the ten lexical categories in the text.

Lexical Category		*% frequency*
Article	AR	8%
Conjunction	CC	1%
Preposition	IN	7%
Adjective	JJ	4%
Noun	NN	30%
Pronoun	PR	10%
Possessive ('s)	PS	2%
Adverb	RB	1%
Verb	VB	20%
Sentence boundary	/S	17%

2.2. The networks

The two networks used in this study were an Elman network and an RCC network. For both nets there were ten input and ten output units representing the sparsely coded lexical categories. The task in all cases was to predict the next lexical category given the current category. State unit activations were not reset to zero on presentation of a sentence boundary as is sometimes done. The Elman network was trained by standard backpropagation using momentum = 0.9. Step-size and number of training epochs varied depending on the requirement for slow or fast training. The slow training regime used stepsize = 0.0001 for 100,000 epochs and a typical fast training regime was 200 epochs at 0.01 followed by another 200 at 0.001. One training epoch consisted of one complete presentation of the training data. The RCC network was trained by Quickprop. Error of the outputs was measured as the root-mean-square (rms) of the difference between the output and some target or reference value averaged over the outputs. The entropy of the outputs was calculated as

$$H_o = -\sum_{i=1}^{10} o_i \log_2 o_i \ .$$

2.3. N-grams

In order to assess the learning of the neural networks, prediction performance was compared with that of n-grams obtained by a statistical analysis of the data. Using the complete data sequence of 643 word categories, 48%, 62%, 72% and 76% correct predictions could be obtained using bigram, trigram, 4-gram and 5-gram probabilities respectively.

3. Results

3.1. Learning by Elman networks

An Elman network having 9 hidden units and trained for 100,000 epochs was able to learn 72% of the total data. However using a “leave-one-sentence-out” 106-fold cross-validation technique, the best generalisation result following fast training was 63%. Figure 1 shows the fraction of the training data learned from 10 to 100,000 epochs of slow training. Early learning appears to proceed in discrete phases. In the first phase (up to 1000 epochs), the network predicts only NN, the category having highest frequency (30%). In phase 2 (1000 to 3000 epochs) the network predicts only NN or /S and scores 45% (the combined frequency of NN and /S is 47%). In phase 3 (3000 to 4000 epochs) the network predicts either NN, /S or VB, the three most common categories and at 5000 epochs it is predicting either NN, /S, VB or AR.

The network’s rms error with respect to the targets (labeled as *“target error”* in Figure 1) declined continuously during learning down to 0.160 at 80,000 epochs and increased slightly subsequently. It is also useful to measure the network’s *rms* error with respect to n-gram probabilities on the assumption that the network should be learning n-gram probabilities with *n* increasing during training. These errors are referred to as *bigram, trigram and 4-gram errors* in Figure 1. Bigram error is initially less than trigram and 4-gram errors and declines most rapidly from 800 to 3000 epochs. It begins to increase again after 4000 epochs while trigram and 4-gram errors continue to decline. After about 8,000 epochs, trigram error reaches a minimum value of 0.067 and then starts to increase. 4-gram error continues to decline to a value of 0.068 at 80,000 epochs after which it also starts to increase. 5-gram error (not shown in Figure 1 to preserve clarity) declines to a value of 0.076 at 100,000 iterations but is beginning to level out.

To confirm that the Elman network is making predictions based on conditional probabilities and also to justify the calculation of output entropy as defined in the Methods section, we require that the sum of outputs should be close to 1.0. In Figure 2 it can be observed that from about 100 epochs, the average sum of outputs is indeed close to 1.0, although the standard deviation of the average sum increases from 0.02 at 100 epochs to 0.19 at 100,000 epochs. The entropy of the outputs (a measure of the network’s ‘uncertainty’ about the next predicted category) declines as learning proceeds (Figure 2), but showing two ‘flat’ periods corresponding to ‘flat’ periods in target error.

3.2. Comparison of Elman and RCC networks

When trained on the set of 485 training patterns, the RCC network continued to add hidden units and was able to learn 99.6% of patterns after adding 42 hidden units (Figure 3). However a maximum generalisation of 63% on the test set was achieved after only 4 hidden units and generalisation declined with further addition of hidden units (Figure 3). By contrast, when Elman networks with 1-50 hidden units were trained on the same data, there was no simple recognisable relationship between generalisation and hidden layer size. An Elman network with 4 hidden units scored 60% on the test set, 3% lower than an RCC net of the same size. An Elman network with 9 hidden units scored 64%. However the best generalisation score of 68% was achieved with 42 hidden units.

3.3 Prediction Uncertainty

Figure 4 shows a graph of prediction uncertainty (measured as the entropy of the output units) over a part of the sequence of category targets. Each point is labeled with the target category. Highest entropy always occurs when the input is the first VB in the sentence. An increase in entropy is also associated with the first category in the sentence. By contrast there is a low entropy associated with the prediction of sentence termination, 89% of sentence endings being correctly predicted.

3.4. Correctly predicted sequences

It is possible to reconstruct the sequences correctly learned by the Elman network that had learned 72% of the training set. They are shown in Figure 5. The transitions marked with an asterix (>*) are those not predicted by trigram probabilities. Sequences 1 and 5 include complete and grammatical sentence structures.

4. Discussion

4.1. Training the Elman network

After 10 iterations, the network was predicting the NN category for every pattern. Since NN was the highest frequency category, this was the quickest way for the network to reduce its initial prediction error. It could be said that the network was performing equivalently to a unigram predictor.

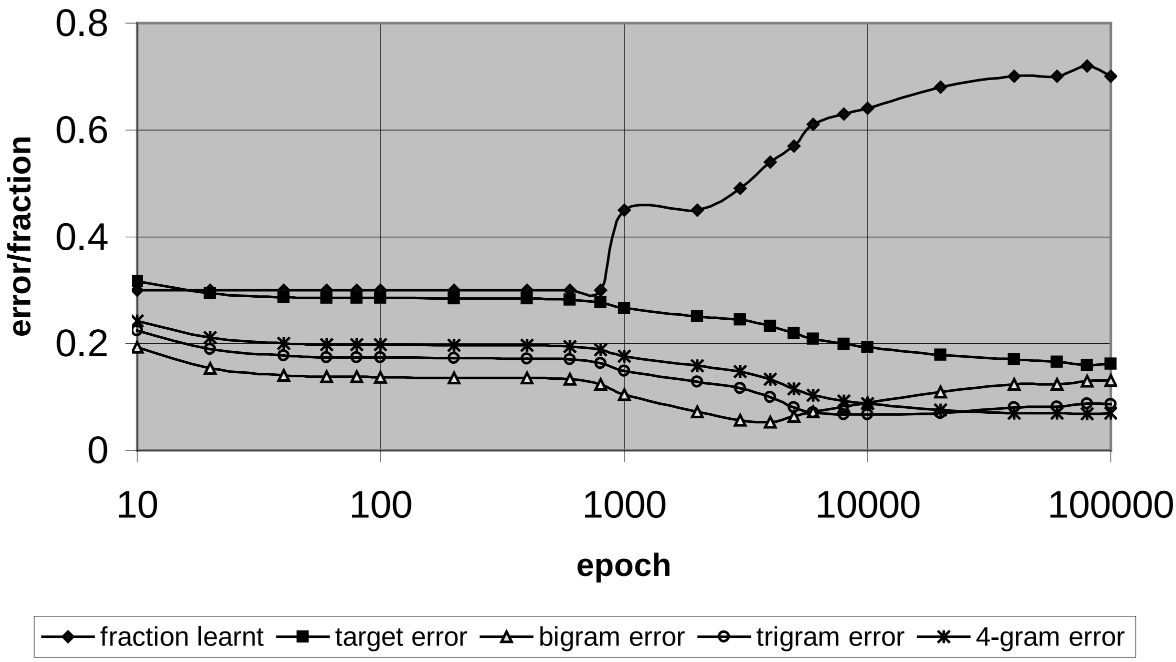

Figure 1: Learning by a simple recurrent neural network of the Elman type having 9 hidden units. Y-axis shows progression of r.m.s. error with respect to the targets, bigram and trigram probabilities and the fraction of training set learned as a function of training epochs.

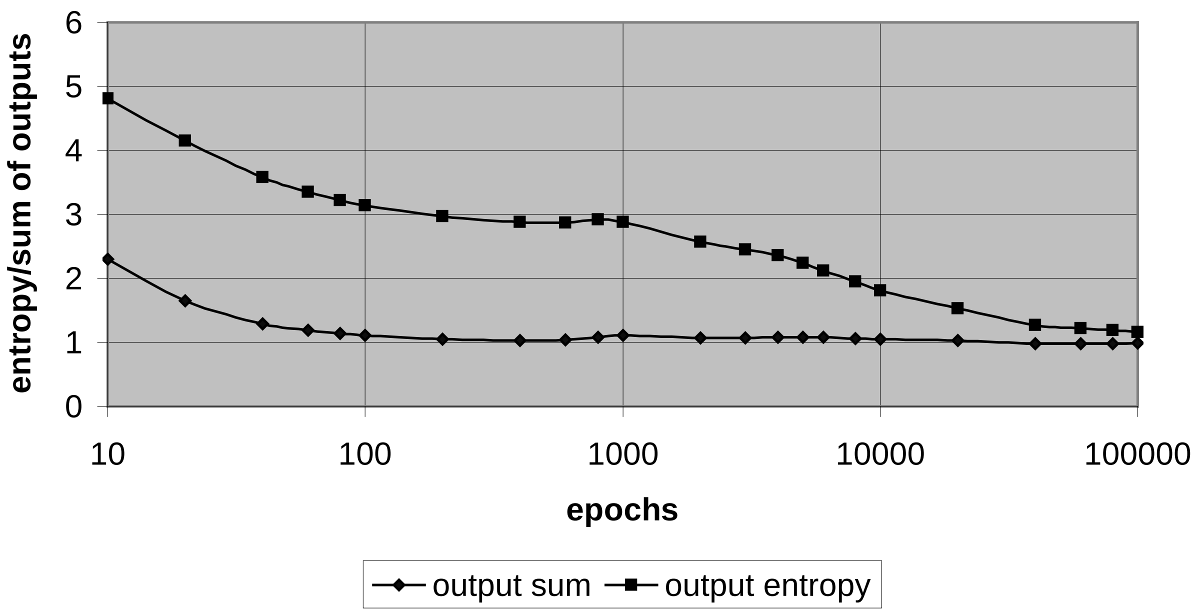

Figure 2: The progression of sum of outputs and entropy of the outputs of an SRN trained over 100,000 epochs.

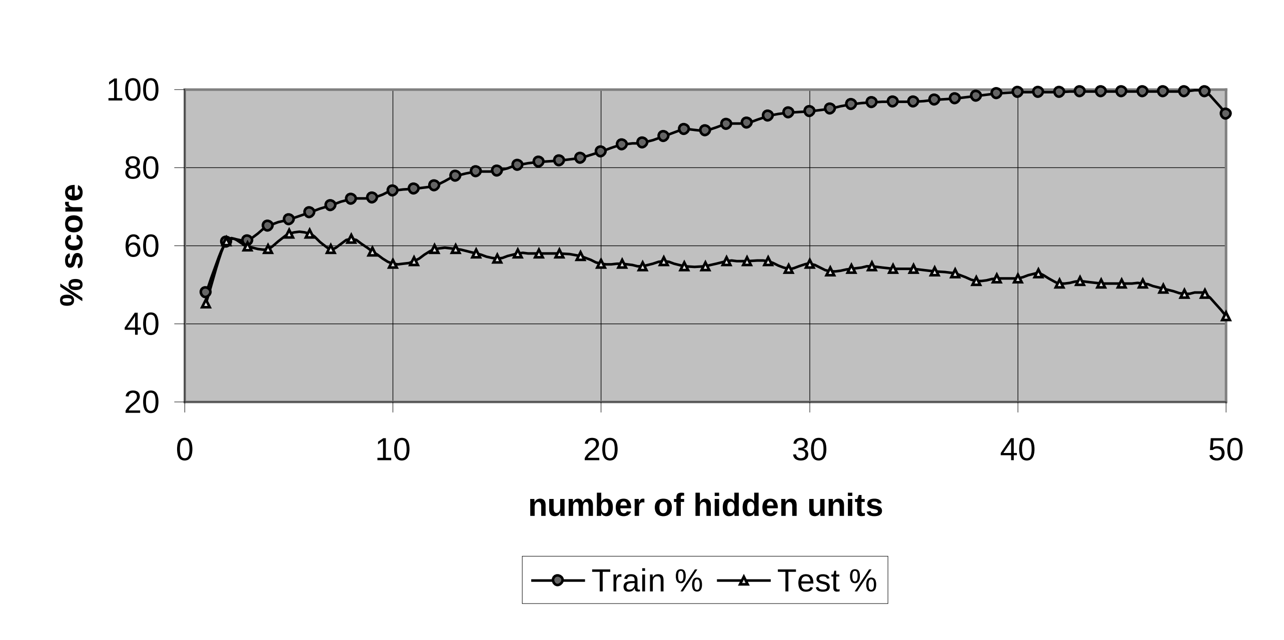

Figure 3: Progression of the score on training set and test set during the training of an RCCN.

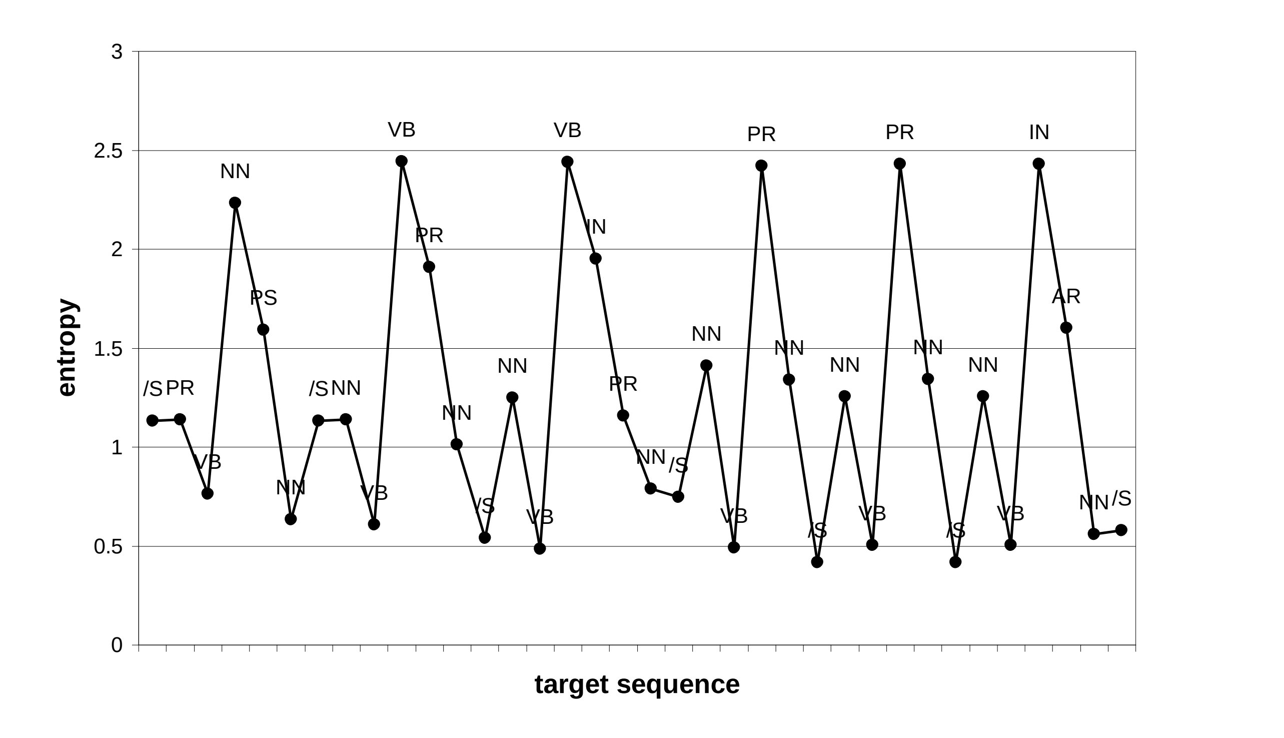

Figure 4: Graph of entropy of the output units over six sentences of the input sequence.

CC >* NN CC
∨* ∨
(1) /S > NN > VB > VB > IN > AR > NN > /S
^ ^
RB VB

(2) IN > NN > /S

(3) AR > JJ > NN > /S

(4) AR >* NN > IN > PR > NN > /S

IN
∨*
(5) /S >* PR >* VB > NN > PS > NN > /S

NN
∨*
(6) PR > VB > NN > /S

(7) AR >* NN >* VB >* JJ >* /S

(8) IN >* PR >* /S

Figure 5: Category sequences correctly learned by a simple recurrent neural network. Those sequences marked with >* are not predicted by trigram probabilities.

Although VB has the second highest frequency (higher than /S), during the second learning phase the network outputs were confined to NN or /S (not VB). This is because the network was beginning to learn bigram probabilities and an inspection of the bigram frequency table revealed that there were 100 instances of /S prediction using bigram probabilities but only 35 instances of predicting a VB. It is during this second learning phase that the *bigram error* decreases most rapidly.

In phase 3, the VB category is added to the network's prediction capability and in phase 4, AR is added, there being only 19 instances where AR would be predicted using bigram probabilities. In fact, using bigram probabilities only these four categories (NN, /S, VB and AR) can be predicted. After about 5000 epochs the network was also correctly predicting other categories, which indicates that it was making predictions based on the current and previous inputs. And indeed we observe that the trigram error rate falls below the bigram error rate around 5000 epochs (Figure 1).

In Figure 5 it is apparent that the network has correctly learned category transitions that are not predicted by trigram probabilities. This is indicative that the network was using at least the current and two previous inputs as context for its decisions. In fact 4-gram error continues to decline up to 80,000 epochs. Since the average sentence length is 5.05 words, it is not surprising that the 5-gram error remains above 4-gram error throughout learning.

Of course it is not being suggested here, that a recurrent network is first learning all the probabilities of a bigram model and then moves on to learn the trigram model and so on. Network learning is driven by the requirement to minimise predictive error. Thus longer sequences having high frequency will bias learning more than infrequently occurring short sequences. Nevertheless an interesting feature of learning apparent in Figure 1 was that minimum *bigram error* was achieved at 4000 epochs when the network had learned 48% of the training set, equivalent to the performance of a bigram predictor. Similarly minimum trigram and 4-gram error was achieved when the network had learned the equivalent of a trigram and 4-gram predictor respectively.

Mention should be made of the decision not to reset state unit activations to zero when the Elman network encountered a sentence boundary. When resets were used, network predictive performance dropped from 70% to 69% with otherwise similar training regimes. In other words, there was minimal information transfer over sentence boundaries and it is more interesting to observe this aspect of network learning than to impose 'forgetting' artificially. The slight increase in performance without resets was probably due to the repetitive nature of the sentences in this text meant for early readers. An additional reason for not using resets was that it made comparisons of network performance with n-gram statistics easier.

4.2. Comparison of Elman network and RCC nets

Although the RCC net was capable of learning almost the entire training set, the hidden unit representations that the network acquired did not generalise well. On the other hand, the best generalising RCC net with four hidden units did better than an Elman network with the same number of hidden units. Due to different learning algorithms, the two networks presumably acquired different hidden unit representations of the underlying task. It is clear that, for this task at least, training an RCC net to find the optimum number of hidden units for an Elman network is not a satisfactory technique.

The maximum RCC score on the test set of 63% was, in fact, an unexpectedly high score. A bigram model acquired from the training set of 80 sentences, predicted 48% and 45% of the training and test set words respectively. The equivalent scores for the trigram model were 63% and 17% respectively. The poor generalisation of the n-gram models for $n > 2$ arose because the test sequences did not have the same statistical structure as the training sequences for $n > 2$. This is the consequence of using natural language sentences and converting the words to lexical categories. A similar difficulty was noted by Lawrence et al (1996) for their NL task which required recurrent networks to classify sentences as either grammatical or ungrammatical.

The experimental paradigm used in our experiment demands alternative measures of generalisation. These might include (1) testing on an *artificially generated* sequence that has the same n-gram (statistical) structure as the NL training sequence (2) testing on the training sequence corrupted with output noise (3) testing on the training sequence but with the sentences in random order. This last is appropriate where resets are not used during training. Such alternative tests of generalisation will be considered in future work.

4.3. Prediction Uncertainty

Elman (1990) found that when a recurrent net was trained on letter sequences consisting of concatenated words, its prediction error tended to decrease from beginning to end of each word. Thus a sharp increase in prediction error could be used to segment the letter sequence into words.

In our study, there was low entropy associated with end-of-sentence prediction, 89% of /S being correctly predicted. Furthermore, when the input was /S, output entropy increased in 84% of cases. However by far the most obvious increase in prediction uncertainty occurred when the input was the first VB of the sentence (Figure 4).

We should not expect that prediction uncertainty will decrease from beginning to end of a sentence in the same way that it does for words, because the rules which govern word structure are different from those which govern sentence structure. For example, the inventory of units that makes up words is so much smaller and the articulation of phoneme sequences is more highly constrained. It is not surprising therefore, to find that in our task, a sharp increase in the network's prediction uncertainty occurs other than when it encounters a sentence boundary.

The first VB in our tagging system was either an auxiliary, or modal or the verb itself, if there was no auxiliary. In other words, the first VB has the largest number of highly probable successors. A linguistic interpretation of the network behaviour is complicated by the small number of lexical categories used in the study. If a more fine-grained system of tagging had been used, the progression of prediction uncertainty through the sentences would have been different. All the sentences in the text consisted of single clauses and the network behaviour is consistent with the verb being the most important determinant of sentence or clause structure.

5. Conclusions

We have described results for the training of Elman and RCC networks on a natural language task. The task is to predict the part-of-speech category of the next word in a sentence given the category of the current word as input. The Elman network appears to be a more useful model for this *one-step-look-ahead* task than the RCC network.

Elman networks are statistical learners and we have shown that network learning can be interpreted in terms of learning n-gram statistics. However because network learning is driven by minimisation of predictive error, longer sequences having high frequency bias learning more than infrequently occurring short sequences.

The sequences correctly learned by the Elman network included some that were not predicted by trigram probabilities, evidence that the network was using the previous three or more inputs as context for prediction.

Prediction uncertainty was highest when the input was the first verb category in the sentence, possibly consistent with the important role that the verb plays in the syntactic structure of a sentence.

6. References

Elman, J.L. (1990). Finding Structure in Time. *Cognitive Science 14*, 179-211.

Fahlman, S.E. (1991). *The Recurrent Cascade Correlation Architecture*. (Tech. Rep. CMU-CS-91-110). Pittsburgh, PA.: Carnegie Mellon University.

Lawrence, S., Fong, S. and Giles, C.L. (1996), Natural language grammatical inference: a comparison of recurrent neural networks and machine learning methods. In *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*. Eds S. Wermter, E. Riloff and G. Scheler. pub Springer-Verlag.

Hume, M.A. (circa 1950). *The Happy Way to Reading.* Blackie and Son Ltd, London and Glasgow.

Singer, W. (1995). Development and Plasticity of Cortical Processing Architectures. *Science, 270*, 758-764.

Wiles, J. & Elman, J.L. (1995). Learning to Count without a Counter: A case study of dynamics and activation landscapes in recurrent networks. *Proceedings of the Seventeenth Annual Conference. of the Cognitive Science Society*., Cambridge, MA: MIT Press.