Sparsely-connected recurrent neural networks for natural language learning

Claire D'Este, Michael Towsey, Joachim Diederich
PO Box 2434
Machine Learning Research Centre
Queensland University of Technology
Brisbane, Queensland 4001
AUSTRALIA
{deste,towsey,joachim}@fit.qut.edu.au

Abstract
This paper demonstrates empirically that simple recurrent networks (SRNs) can be more efficient learners when their nodes are sparsely connected rather than fully connected. Small increases in generalisation accuracy and learning speed are achieved when the number of hidden units is increased, while maintaining the same number of weights in the network. This performance increase is explained by the increased dimensionality of the hidden unit space without causing over-fitting by increasing the total number of weights. The impact of sparse connectivity between the different layers of units is studied. The networks are trained on a one-step-look-ahead task with natural language data.

Keywords: recurrent neural networks, sparse connectivity, natural language processing

1 Introduction

This paper demonstrates empirically that the learning performance of a simple recurrent network (SRN) can be increased by sparsely connecting its nodes. Recurrent neural networks are attractive structures for modeling higher cognitive processes, such as language processing, because of their dynamics [Hertz et. al., 1991] and similarity to some regions within the brain [Singer, 1995]. However, current implementations of SRNs concentrate on 'standard' or generic architectures (such as the Elman [1990] architecture) that are fully connected in the forwards direction and thus are not consistent with the sparse connectivity of the neo-cortex. The neo-cortex is sparsely connected, not just when viewed globally (any one neuron is typically connected to only 10^2 - 10^4 of the 10^{11} neurons in the total brain), but even at the local level (any one neuron is typically connected to a small fraction of neurons in the neighbouring 1 square cm of tissue). This raises the question of how the brain achieves its computational feats with sparse connectivity and casts doubt on the usefulness of neural models which rely upon full forward connectivity.

As a rule of thumb, the representational or memory capacity of a feed-forward network or an SRN is a function of the number of its modifiable weights. This heuristic has been formalised for feed-forward networks in terms of the VC dimension. This however leaves open the question of the effect of architecture on learning. That is, do two networks having the same number of connections but different numbers of neurons (and therefore connection density) have the same ability to learn?

Elman [1990] demonstrated the power of recurrent neural networks when learning a sequence of language input patterns. Towsey et. al. [1998] performed experiments that suggested that Elman-style networks were more suitable for the one-step-look-ahead task than Recurrent Cascade Correlation networks. Their experiments used natural language data sets, whereas Elman's experiments used artificial, formal languages [Wiles & Elman 1995].

Previous implementations of sparsely connected SRNs include Minai and Levy [1993] who attempted to simulate the CA3 region in the mammalian hippocampus with an SRN containing sparse asymmetric connections. The networks were random and untrained and produced interesting behaviour, including extremely long cycles.

Strom [1997] presents results with sparsely connected SRN and time-delay hybrid networks trained on a phoneme recognition task. The networks recorded 69.5% generalisation accuracy with approximately 33% connectivity, compared with 63.9% accuracy for fully connected networks. Unlike the networks discussed within this paper, where recurrent connections are added randomly, the probability of a recurrent connection between two hidden units within Ström's networks was dependent on the distance between them. Ström [1997] experimented with varying connectivity between the different subsets of connections. However, conclusions are difficult to derive from the results as to why the performance increases, or the impact of sparsely connecting the different weight subsets.

An additional method of reducing connections is termed pruning and consists of training a fully connected network and removing any connections that have weights below a certain threshold. The best known method of pruning is Optimal Brain Damage [Le Cun et. al. 1990]. Giles and Omlin, [1994], applied pruning to recurrent neural networks. The method they employed pruned the state units with the smallest weight vector and discovered that the networks could generalise better over the test set. Pruning does not decrease training time, but it can reduce run time, as less calculations are made.

Ghosh and Tumer [1994] experimented with sparse connectivity within feed-forward networks. Connectivity was decreased to the output units until they were clearly separated, or decoupled from the hidden units. Their belief was that this decoupling, and the decrease in parameters, provided the network with more 'freedom'. The generalisation performance of such networks was compared with fully connected networks and networks using Optimal Brain Damage pruning methods. Their results indicated that the sparsely connected networks had better generalisation performance only when there was noise in the input data or multi-dimensional output. Although, the performance increase was slight, there was no increase recorded in complexity or training time.

To produce more realistic models of biological memory, Vogel and Boos [1997] developed a series of nonrandom, multi-layer, sparsely connected networks, known as *P-nets*. In a P-net, all paths between neurons in the same layer are of length two, that is via a single neuron in another layer. The authors found that P-nets had a larger memory capacity (measured as bits per synapse) than did fully connected Hopfield type networks.

In this paper, we use the Elman-type SRN [Elman, 1990] as the standard fully connected architecture and modify the connectivity in a systematic way to identify the effect on learning when different sub-sets of connections are made sparse.

2 Methods

2.1 The architecture

The initial network consists of 10 input, 10 output, 8 hidden and 8 context/state units, conforming to the standard Elman architecture [Elman, 1990] [Figure 1]. This network had 224 weights and was chosen as the benchmark architecture because previous studies [Towsey et al, 1998] demonstrated it to yield the optimum generalisation score on the task at hand. Note that the number of weights cited above excludes the one-to-one copy back connections from the hidden to state units. Increasing the number of hidden and state units while maintaining a constant number of connections/weights, reduced the connection density of the network. For the requirements of this paper, the weights were divided into 3 subsets as specified in Figure 1. Networks with sparse connectivity in all weight subsets are termed *fully sparse*.

The connections were selected randomly for a sparse subset by assigning to each connection a probability of occurring in this layer. This probability was determined by dividing the number of connections wanted in this layer, by how many there would be if it were fully connected.

$$P(x) = \frac{W}{h(i+h+o)} \qquad (1)$$

Equation 1 specifies the method of connection selection used for fully sparse networks. W is the weight constant, i is the number of input units, h is the number of hidden and state units and o is the number of output units.

When selecting connections for the different weight subsets, the weights required for the subsets left fully connected must be subtracted from the weight constant . This is then divided by the number of connections there would be in a fully connected version of the weight subset to be made sparse. Equation 2 gives an example of this method for creating sparse connections in the recurrent subset of weights.

$$P(x) = \frac{W - h(i + o)}{h^2} \qquad (2)$$

When selecting connections for the different weight subsets, the weights required for the subsets left fully connected were subtracted from the constant weight total. This is then divided by the number of connections there would be in a fully connected version of the weight subset to be made sparse.

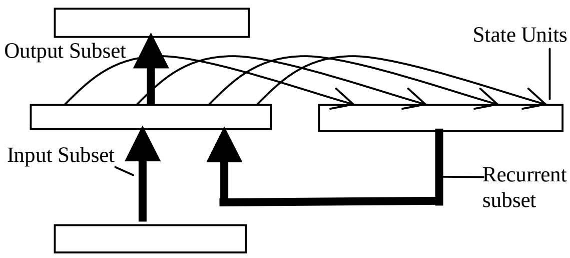

Fig. 1. Standard Elman Architecture and weight subsets

The networks were trained for 400 epochs with a learning rate of 0.01 and for a further 10000 epochs with a learning rate of 0.001. The momentum was set at 0.9 and the networks were tested every 500 epochs.

2.2 The task

The natural language input was gained from a 1950's first-year primary school reader. The input data then consisted of 106 sentences of three to eight words in length, with an average length of 5.06 words. These words were then converted to 10 grammatical, or lexical, categories (see Towsey et al. [1998] for details).

The networks were trained to predict the next lexical category given the current category. This is known as a one-step-look-ahead task. A bit error is recorded each time the network predicts an incorrect category. Hidden unit activations were not reset on sentence boundaries.

3 Results

3.1 Overview

Two dimensions of learning performance were measured, the accuracy of learning and speed of learning. Accuracy, measured as the percentage of correct predictions, was obtained for both the training and test sets over 10,000 training epochs. Only the best training score and test score over the 10,000 epochs are reported. Learning speed of the networks was measured as their test score at 4000 epochs.

3.2 Learning Accuracy

3.2.1 Sparse Weight Subsets

The first set of experiments was designed to explore the effect on learning of sparse connectivity for each weight subset. Networks were trained with of 9, 10 and 11 hidden and state units, this corresponds to weight subsets of approximately 55%, 24% and 3% connectivity. Sparse connections were created within each weight subset separately. In each case the connections between the other layers remained fully connected. For each weight subset and number of hidden and state units the networks were trained and tested on a single architecture of random connections, with 10 different sets of randomly initialised weights. The testing and training errors were then averaged across the 10 experiments for this weight subset.

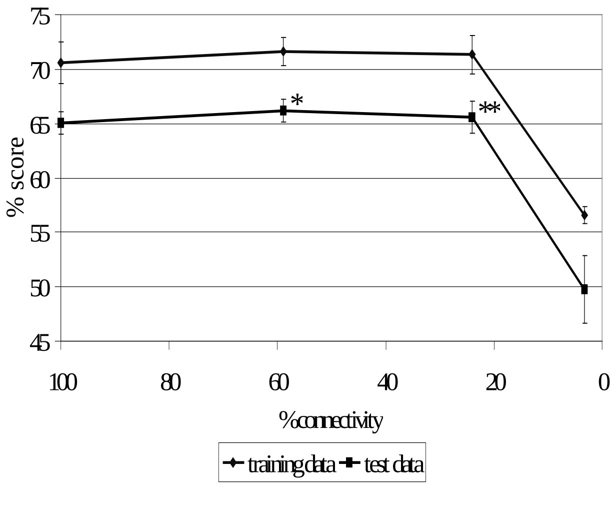

Figure 2: Learning accuracy with reduced connectivity between the input and hidden units
* P < 0.5% ** P < 5% significance compared with 100% connectivity

Each weight subset demonstrated relatively constant test and training score for networks with over 24% connectivity. Networks tested with lower connectivity showed a significant decrease in accuracy. A slight increase in training and test accuracy was observed in networks with 56% and 24% connectivity [Figures 2, 3, 4]. Statistical evaluation found that the increase in test score of the networks was statistically significant [Figure 2, 3, 4].

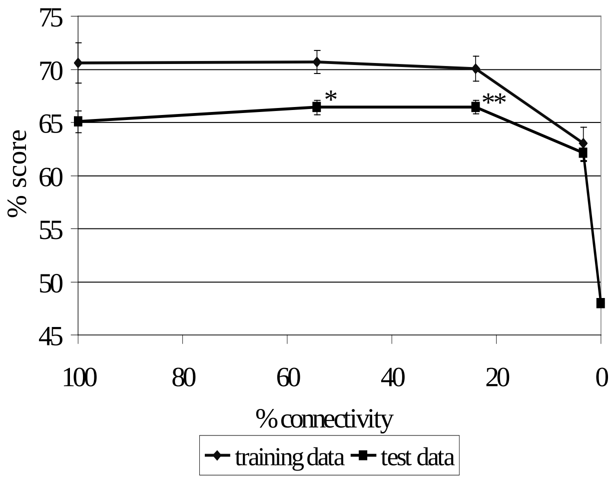

Fig. 3. Learning accuracy with reduced connectivity between the state and hidden units
***** $P < 0.5\%$ ** $P < 0.1\%$
significance compared with 100% connectivity

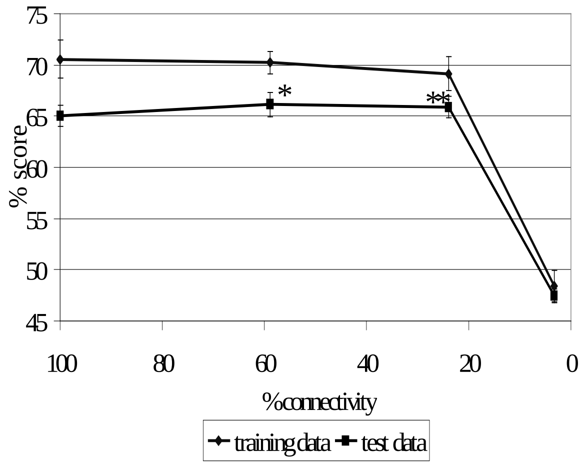

Fig. 4: Learning accuracy with reduced connectivity between the hidden and output units
* $P < 2.5\%$ ** $P < 5\%$
significance compared with 100% connectivity

3.2.2 Fully Sparse Networks

The second set of experiments examined the impact on learning of an entire network of sparse connections. Networks of 12, 16, 22 and 32 hidden units were examined. These correspond to networks of 58%, 40%, 24%, and 13% connectivity. Ten different architectures for each number of hidden units were created, trained and the average error recorded. In these experiments the separate weight subsets were not assigned portions of the weights, instead each connection had a probability of occurring in the entire network.

The fully sparse networks of less than 40% connectivity had a slight decrease in generalisation performance compared to the fully connected networks [Figure 5]. However, the generalisation accuracy of networks with sparse connectivity of up to 58% was directly comparable with fully connected networks [Figure 5].

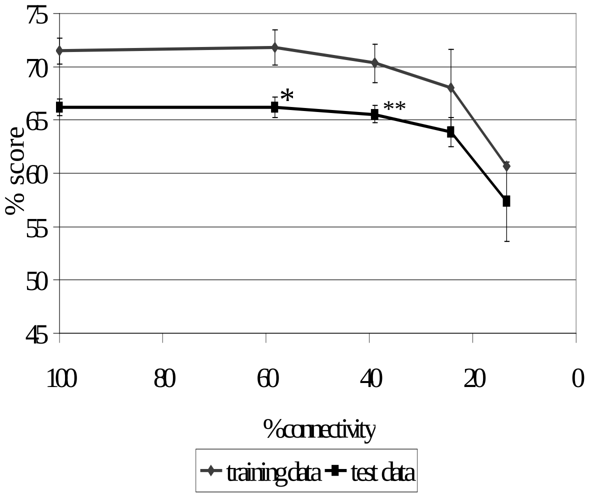

Fig. 5. Learning for fully sparse networks
* P > 5% ** P < 5% significance compared with 100% connectivity

3.3 Learning Speed

Networks with layers of sparse connectivity of 55 to 24% appear to learn faster than fully connected networks [Figure 6]. Even the fully sparse networks with 58% connectivity learned faster than the fully connected network.

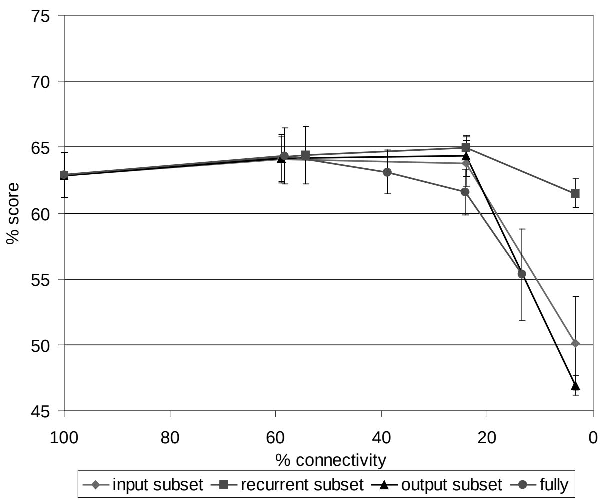

Fig. 6. Learning speed (measured as test score at 4000 epochs) as a function of connectivity.

4 Discussion

The increase in accuracy and speed of learning of sparse networks cannot be attributed to changes in overfitting to the training data since the number of weights in each of the networks remained the same. The Elman network architecture is frequently thought of as an MLP with an additional layer of state neurons having recurrent connectivity. However, an Elman network trained properly to perform the one-step-look-ahead task on temporal sequences can also be thought of as a finite state automaton (FSA). From this perspective, the hidden unit activations can collectively be viewed as encoding the states of the automaton. Increasing the number of hidden units to achieve sparse connectivity has the effect of increasing the dimensionality of the hidden unit space. Although the outputs of the hidden units are real valued, as training proceeds, weight values increase and the sigmoidal activation function drives the outputs to approach zero or one. We attribute the improved learning by sparsely connected SRNs to the increased dimensionality of the hidden unit space, where each vertex of the hypercube of hidden unit space comes to represent a state of the learned FSA. It is easier for the output units to partition hidden unit space, when the clusters of hidden unit activity are pushed to the vertices of a hypercube. If the hypercube has limited dimensionality, the hidden unit activation states tend to disperse through hidden unit space and become more difficult to partition. Of course this advantage has to be traded against the disadvantage that as sparseness increases, each hidden unit has smaller in-degree and therefore processes less information. The increase in learning speed can be attributed to the reduced in-degree or search space that each hidden unit must explore. However, too small an in-degree and the hidden unit has insufficient information to represent the target function.

Reasonable test scores of 62% could still be achieved on this task with networks having only 3% connectivity within the recurrent weight subset [Figure 3]. (Note that a prediction score of 48% can be achieved with a purely feed-forward network.) This corresponds to only 3 connections from context to hidden units and thus not all the hidden units received recurrent connections. Put another way, our very sparse network continued to demonstrate good memory despite the effective removal of many context units.

5 Conclusions

We conclude from these experiments that although the learning performance of an SRN is predominantly influenced by the number of its connections or weights, the dimensionality of the hidden unit space also has an effect even when the number of weights is held constant. Increasing dimensionality of the hidden unit space increases learning performance. Because the number of weights is held constant, we cannot turn to the usual explanation of under- or over-fitting. We found the optimum architecture for the standard Elman SRN, and then obtained further improvement by trading off an increase in number of hidden units with a decrease in in-degree.

The experimental results suggest that a range of sparsely connected architectures can achieve similar or better performance than the standard SRN architecture. In particular, the specific Elman architecture is not critical to performance. This understanding should increase the perceived biological plausibility of recurrent neural networks and may be useful for the modeling of cognitive tasks, such as natural language processing where standard Elman architectures have been commonly used.

References

Elman, JL. (1990) Finding Structure in Time. Cognitive Science, 14. pp. 179-211

Ghosh, J. Tumer, K. (1994) Structural Adaptation and generalisation in supervised feed-forward networks. Journal of Artificial Neural Networks, 1:4. pp. 431-458

Hertz, J. Krough, A. Palmer, RG (1991), Introduction to the theory of neural computation, Redwood City CA, Addison Wesley

Minai, AA. Levy, WB. (1993) The Dynamics of Sparse Random Networks. Biological Cybernetics, 70:2. pp. 177-187

Giles, CL. Omlin, CW. (1994) Pruning Recurrent Neural Networks for Improved Generalisation Performance. IEEE Transactions on Neural Networks, 5:5. p. 848

Shellhammer, I. Diederich, J. Towsey, M. Brugman, C. (1998) Knowledge Extraction and Recurrent Neural Networks: An Analysis of an Elman Network trained on a Natural Language Learning Task. In D.M.W Powers (ed.) NeMLaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning, ACL, pp. 73-78

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

Ström, N. (1997), Sparse connections and pruning in large dynamic artificial neural networks, http://www.speech.kth.se/~nikko/papers/euro97, April 1999

Towsey, M. Diederich, J. Schellhammer, I. Chalup, S. Brugman, C. (1998) Natural Language Learning by Recurrent Neural Networks: A Comparison 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

Vogel, DD. Boos, W. (1997) Sparsely Connected Hebbian Networks with Strikingly Large Storage Capacities, Neural Networks, 10:4. pp. 671-682

Wiles, J. Elman, JL. (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, MIT Press, Cambridge MA