High Performance Associative Memories and Structured Weight Dilution

S.P Turvey, S.P.Hunt, N.Davey, R.J.Frank

Department of Computer Science, University of Hertfordshire,
College Lane, Hatfield, AL10 9AB. United Kingdom
email: {s.p.turvey, s.p.hunt, n.davey, r.j.frank}@herts.ac.uk

Abstract *The consequences of two techniques for symmetrically diluting the weights of the standard Hopfield architecture associative memory model, trained using a non-Hebbian learning rule, are examined. This paper reports experimental investigations into the effect of dilution on factors such as: pattern stability and attractor performance. It is concluded that these networks maintain a reasonable level of performance at fairly high dilution rates.*

Key-Words *Associative Memory, Hopfield Networks, Weight Dilution, Capacity, Basins of Attraction, Perceptron Learning.*

Introduction

The associative memories examined here are based on the standard Hopfield architecture [10]. It has been known for some time [1] that networks with superior performance to that of the original model can be built. Performance may be improved by using an alternative learning rule: either one that finds an approximation to the projection weight matrix, or one that implements perceptron-style learning. (See [6,7,14] for a comparison of performance of different models).

Weight dilution is a technique for reducing the degree of connectivity within a network. Connections are removed after training has taken place (*post-training dilution*). For one-shot Hebbian learning, as employed in the 'standard' Hopfield model, it is known [13] that capacity drops linearly with the fraction of connections removed. It has even been suggested that an associative memory may be trained by starting with a fully connected network with random fixed weights and systematically removing a fraction of the connections [12].

Models Examined

In each experiment we train a network of N units with a set *bipolar* (+1/-1) training vectors, $\{\xi^p\}$. The resulting weight matrix is denoted by $\mathbf{W}$. During recall the net input, or *local field*, h, of a unit and its *next* state, S', are calculated as follows:

$$h_i = \sum_{j \neq i} w_{ij} S_j \qquad S'_i = \begin{cases} +1 & \text{if } h_i > \theta_i \\ -1 & \text{if } h_i < \theta_i \\ S_i & \text{if } h_i = \theta_i \end{cases}$$

where w_{ij} is the weight on the connection from unit j to unit i, S_j is the *current* state of unit j, and θ_i is the *threshold* on unit i (zero in all of our experiments).

Unit states are updated asynchronously in random order. This and a symmetric weight matrix guarantee simple point attractors in the network's state space, each of which is a stable state of the network.

A training vector, ξ^p, will be a stable state of the network if the *aligned local fields*, $h_i\xi_i^p$, are non-negative for all i (assuming all θ_i are zero). Each training vector that is a stable state is known as a *fundamental memory* of the trained network. The *capacity* of a network is the maximum number of fundamental memories it can store. The *loading*, α, on a network is calculated by dividing the number of vectors in the training set by the number of units in the network, N.

Learning Rules

Two learning rules have been employed in this work. The first, described by Blatt & Vergini [3], approximates the projection matrix generated using the pseudo-inverse rule (see [8] for details). The second is Gardner's perceptron-like symmetric local learning rule [6,9].

Blatt & Vergini's Rule (BV)

Blatt & Vergini's [3] rule is an iterative method for approximating the projection weight matrix. The algorithm is guaranteed to find an appropriate weight matrix within a finite number of presentations of each pattern if such a matrix exists.
The minimum number of presentations of the training set to perform, P, is calculated as being the smallest integer conforming to:

$$P \geq \log_k \left(\frac{N}{(1-T)^2} \right)$$

where k and T are real valued constants such that $1 < k \leq 4$ and $0 < T < 1$. k is referred to as the *memory coefficient* of the network; the larger it is, the fewer steps are required to train the network. In this work, k=4 and T=0.5 for all networks trained by this rule.

The algorithm is as follows:

```
BEGINNING WITH A ZERO WEIGHT MATRIX
FOR EACH PATTERN IN TURN
    APPLY THE PATTERN ONTO THE NETWORK
    FOR $m$ := 1 TO $P$
        FOR EACH PROCESSING ELEMENT IN TURN
            UPDATE INCOMING WEIGHTS ...  $\Delta w_{ij} = \left(\frac{k^{m-1}}{N}\right)\left(\xi_i^p - h_i\right)\left(\xi_j^p - h_j\right)$
REMOVE ALL SELF-CONNECTIONS
```

Note: patterns are added *incrementally*.

Symmetric Local Learning (SLL)

Gardner [9] pointed out that an iterative perceptron-like training rule could be made to produce symmetric weights by simply updating both w_{ij} and w_{ji} when either changes. Gardner also showed that such algorithms would find a symmetric weight matrix, if one existed, for a particular training set. The SLL algorithm is:

```
BEGIN WITH A ZERO WEIGHT MATRIX
REPEAT UNTIL ALL LOCAL FIELDS ARE CORRECT
    SET THE STATE OF NETWORK TO ONE OF THE $\xi^p$
    FOR EACH UNIT, $i$, IN TURN
        IF $h_i^p \xi_i^p$ IS LESS THAN $T$ THEN
            UPDATE WEIGHTS ON CONNECTIONS INTO AND OUT OF UNIT $i$:
            $\Delta w_{ij} = \Delta w_{ji} = \frac{\xi_i^p \xi_j^p}{N}$
            OTHERWISE DO NOTHING
```

This is a symmetric version of perceptron learning, with a fixed margin, T, and learning rate $1/N$. We call T the *learning threshold* for the network. Since a set of patterns is stable when the aligned local fields of those patterns are all non-negative, we could set T to zero. However, based on previous results [6], we choose T=10 in order to achieve a better attractor performance for the networks.

Training Sets

Throughout this work we employ training sets made up of pseudo-random training vectors. An uncorrelated training set is one in which the patterns are *completely* random. Correlation can be increased by varying the probability that a given bit in a training pattern is +1 (or –1). We refer to the probability of any bit being +1 in each training vector as the *bias*, b, on the training set, so $\forall\, i,p \bullet prob\,(\xi_i^p = +1) = b$. Thus, a bias of 0.5 corresponds to an uncorrelated training set and a bias of 1 corresponds to a completely correlated one (as does a bias of 0).

Weight Dilution

We present two approaches to weight dilution. In the first, pairs of units are chosen at random and, if the units are connected, *both* connections between them are removed, until the correct proportion of connections has been removed from the network. By removing both connections between units we ensure that the weight matrix remains symmetrical.

The second approach involves using an heuristic to select the connections to be removed, in an attempt to ensure that the most efficacious connections are retained [2,4,5]. In this case, the connection pair with the weight of least magnitude is identified, and both connections in the pair are removed from the network. This is repeated until the required number of connections have been eliminated.

Analysing Performance

Two aspects of network performance were measured: *pattern stability* and *attractor performance*. A series of experiments were carried out on networks of size N=100 using training sets of bias 0.5 and 0.9 and at a fixed loading of α=0.50. Networks were trained using either the BV rule, or the SLL rule. The connections in the networks were then diluted according to the methods described above.

Pattern Stability

The proportion of fundamental memories that are stable after dilution provides an indicator of the robustness of a model. Networks were trained to below maximum capacity, so all training patterns were fundamental memories prior to dilution. Figs 1 and 2 show the *proportion* of stable training patterns at various dilutions.

Attractor Performance

For an associative memory to be effective, the training patterns should be both stable states of the network and attractors in its state space.

We use R, the normalized mean radius of the basins of attraction [11], as a measure of attractor performance. It is defined as:

$$R = \left\langle \left\langle \frac{1 - m_0}{1 - m_1} \right\rangle \right\rangle$$

where m_0 is the minimum overlap an initial state must have with a fundamental memory for the network to converge on that fundamental memory, and m_1 is the largest overlap of the initial state with the rest of the fundamental memories. The angled braces denote an average over sets of training patterns. Details of the algorithm used can be found in [11].

Results

Figs 1 and 2 below show the effect of varying the degree of weight dilution within the network on the proportion of trained patterns that are stable, whilst figs 3 and 4 show the effect of varying the degree of weight dilution on attractor performance (*N*=100, □=0.50 throughout).

In each figure, the left hand graph shows performance of networks trained with uncorrelated patterns (b=0.5), whilst the right hand graph shows performance of networks trained with correlated patterns (b=0.9). The upper line on each of the eight graphs represents the results obtained using informed dilution.

Pattern Stability

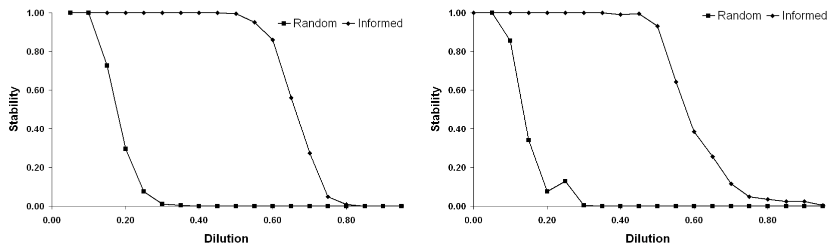

Figure 1. Pattern stability in networks trained with the BV rule.
Left-hand graph: bias=0.5. Right-hand graph: bias = 0.9.

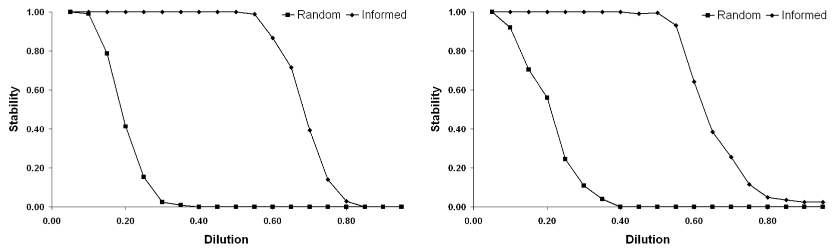

Figure 2. Pattern stability in networks trained with the SLL rule.
Left-hand graph: bias=0.5. Right-hand graph: bias = 0.9.

Attractor performance

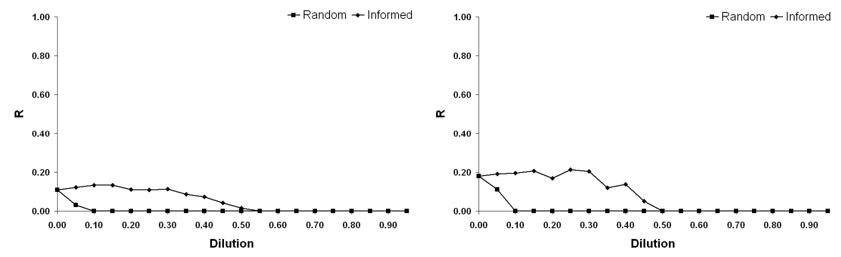

Figure 3. Attractor performance in networks trained with the BV rule.
Left-hand graph: bias=0.5. Right-hand graph: bias = 0.9.

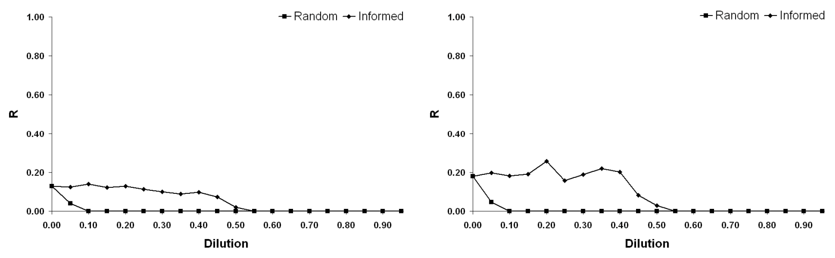

Figure 4. Attractor performance in networks trained with the SLL rule.
Left-hand graph: bias=0.5. Right-hand graph: bias = 0.9.

Discussion

Observations on Pattern Stability

We make four key observations here:

1) Informed dilution gives a significant improvement in pattern stability over simple random dilution. This takes the form of an increase in the level of dilution at which the networks retain memory of all the trained patterns.
2) It is possible to remove around 50-60% of the networks' connections without a serious decline in the stability of the trained patterns.
3) The bias in the training set makes very little difference to the pattern stability.
4) The learning rule used appears to make little difference to the effect of dilution on pattern stability.

Observations on Attractor Performance

The pattern here is similar to that for pattern stability. Specifically:

1) Informed dilution performs significantly better than simple random dilution.
2) It is possible to remove up to approximately 40% of the networks' connectivity without serious damage to the attractor performance of the network.
3) The bias in the training set makes little difference to the attractor performance.
4) The learning rule used appears to make little difference to the effect of dilution on attractor performance.

Conclusions

This paper reports two important results:
1) Informed dilution is markedly better than random dilution.
2) Informed dilution demonstrates that a large number of connections are redundant in networks of this type and at these loadings.

As the loading of these networks is □=0.5 they are below their maximum storage capacity; it may be of interest to repeat these experiments at higher loadings where the networks may be under greater stress.

Failure, when it occurs, proceeds with great rapidity. There is a sharp decrease in both proportion of stable patterns and attractor performance once the networks begin to lose their stability and ability to act as attractors. In this respect, our results differ from those of Sompolinksy, whose work on randomly diluting the traditional Hopfield network [13] showed a linear decline in pattern stability.

The system of informed dilution we have presented is very simple; no re-training of the network is required. It is possible that in biological systems complex strategies may be similarly unnecessary. Chechik *et al* [5] have noted that during brain maturation there is a reduction in connectivity that is expensive to maintain from an energy perspective. It is interesting that we have also found redundancy in connectivity, albeit in a much simpler system.

Informed dilution, as implemented in this work, is functionally equivalent to the system of *annealed dilution* proposed by Bouten *et al.* [4] in which the dilution is performed as part of the learning process. Our results concur with their prediction that 60% dilution is the approximate limit beyond which network capacity is compromised.

The work presented here concentrates on the dilution of *fully connected* networks, whereas our current work focuses on networks that have been created as sparsely-connected *tabula rasa.* Training these networks has presented new challenges and performance characteristics. We expect to be able to present these new findings in the near future.

References:

[1] Abbott, L.F (1990) Learning in neural network memories. *Network: Computational Neural Systems*, **1**, 105-122

[2] Barbato, D.M.L. and O.Kinouchi (2000) Optimal pruning in neural networks. *Physical Review E* **62**(6), 8387-8394

[3] Blatt, M.G. and E.G.Vergini (1991) Neural networks: a local learning prescription for arbitrary correlated patterns. *Physical Review Letters* **66**(13), 1793-1797

[4] Bouten, M., A.Engel, A.Komoda, and R.Serneels (1990) Quenched versus annealed dilution in neural networks. *Journal of Physics A*, **23**, 4643-4657

[5] Chechik, G., I.Meilijson and E.Ruppin (1998) Synaptic Pruning in Development: A Computational Account, *Neural Computation* **10**, 1759-1777

[6] Davey, N., R.G.Adams, and S.P.Hunt. High Performance Associative Memory Models and Symmetric Connections (2000) *Proceedings of the International ICSC Congress on Intelligent Systems and Applications (ISA'2000): Symposium on Computational Intelligence (CI'2000),* **2**, 326-331

[7] Davey,N. and S.P.Hunt (2000) A Comparative Analysis of High Performance Associative Memory Models. *Proceedings of 2nd International ICSC Symposium on Neural Computation (NC'2000)* 55-61

[8] Diederich,S. and M.Opper (1987) Learning of Correlated Patterns in Spin-Glass Networks by Local Learning Rules. *Physical Review Letters*, **58**, 949-952

[9] Gardner,E., H.Gutfreund and I.Yekutieli (1989) The Phase Space of Interactions in Neural Networks with definite Symmetry, *Journal of Physics A*, **22**, 1995-2008

[10] Hopfield, J.J (1982) Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences (USA)*, **79**, 2554-2558

[11] Kanter, I. and H. Sompolinsky (1987) Associative Recall of Memory Without Errors. *Physical Review A*, **35**(1), 380-392

[12] López,B. and W. Kinzel (1997) Learning by dilution in a neural network. *Journal of Physics A*, **30** 7753-7764

[13] Sompolinsky, H. (1986) Neural Networks with nonlinear synapses and a static noise, *Physics Review* A **34**, L519-L523

[14] Turvey, S.P., S.P.Hunt, N.Davey and R.J.Frank (2001) An experimental assessment of the performance of several associative memory models. *Proceedings of the 5th International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA 2001)*, 70-73