INFORMATION CONTENT IN NEURAL NET OPTIMIZATION

O. M. Omidvar, University of the District of Columbia
Washington, DC 20008
C L. Wilson, National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

Reduction in the size and complexity of neural networks is essential to improve generalization, reduce training error, and improve network speed. Most of the known optimization methods heavily rely on weight sharing concepts for pattern separation and recognition. In weight sharing method the redundant weights from specific areas of input layer are pruned and the value of weights and their information content play a very minimal role in the pruning process. The method presented here focuses on network topology and information content for optimization. We have studied the change in the network topology and its effects on information content dynamically during the optimization of the network. The primary optimization is Scaled Conjugate Gradient (SCG) and the secondary method of optimization is a Boltzmann method. The conjugate gradient optimization serves as a connection creation operator and the Boltzmann method serves as a competitive connection annihilation operator. By combining these two methods its is possible to generate small networks which have similar testing and training accuracy, i.e. good generalization. from small training sets. In this paper we have also focused on network topology. Topological separation is achieved by changing the number of connections in the network. This method should be used when the size of the network is large enough to tackle real life problems such as fingerprint classification. Our findings indicate that for large networks, topological separation yields a smaller network size that is more suitable for VLSI implementation. Topological separation is based on the error surface and information content of the network. As such it is an economical way of size reduction that leads to overall optimization. The differential pruning of the connections is based on the weight contents rather than number of connections. The training error may vary with the topological dynamics but the correlation between the error surface and recognition rate decreases to a minimum. Topological separation reduces the size of the network by changing its architecture without degrading its performance.

1 Introduction

The Boltzmann methods have been used as a statistical method for combinatorial optimization and for the design of learning algorithms [1].[2]. This method can be used in conjunction

with a supervised learning method to dynamically reduce network size . The strategy used in this research is to remove the weights using Boltzmann criteria during the training process. Information content is used as a measure of network complexity for evaluation of the resulting network [1].

The Cross Validation technique have been used for optimization by Moody [3], in which the information content is removed before the training. In the Optimal Brain Damage method the optimization takes place after the training is done, see Le Cun [4]. The type of optimization described in this paper is simultaneous, which means the optimization takes place concurrent with the training process. During the optimization process the Boltzmann method is used in conjunction with Scaled Conjugate Gradient (SCG) mechanism. The Boltzmann method works as a connection annihilator while SCG is the connection creator. There are several points where these two methods can be compared. The Boltzmann method is self-organizing while the SCG method is a supervised learning method. The Boltzmann method seeks to minimize the the number of weights while maintaining the information content of the network. The SCG method seeks to minimize an error function on the training set. The information in the network during the iteration time t, as $t \rightarrow \infty$ is used as the control parameter for Boltzmann method. The control parameter for the SCG method is the information provided at $t = 0$ in the initial weights. The algorithmic control in the Boltzmann method is the temperature sequence applied during the iteration. The equivalent controlling parameter for the SCG method is the restart sequence.

A standard method of optimization for real world problems is weight sharing [5]. The weight sharing method increases the redundancy of the network while reducing the Vapnick-Chervonenkis (VC) dimension [6]. Weight sharing lowers the network capacity and decreases the network entropy. The increase in redundancy and decrease in network entropy lead to larger size networks with minimal information capacity. A very large training set is needed to train such a network. Even after the training the generalization power of the network can not be estimated.

The optimization strategy used in this research focuses on network topology as it effects information content and the quality of information represented in the network. This results in a smaller network with a very high information content that allows the use of a reasonably small training set. We have also done the topological separation to verify that the method is successful for networks with different number of neurons in the hidden layer. The information content for topologically equivalent networks is basically the same and the change in the number of neurons in the hidden layer has little or no effect in the information content and the generalization power of the optimized network [7].

We have used the Boltzmann method as a secondary method of optimization to prune the networks. This method has been applied to both supervised and self organizing networks [8]. The method can be used in conjunction with a primary method of optimization such as Scaled Conjugate Gradient scheme [9]. The resulting optimized network has been used for both fingerprint and handwritten character recognition. The recognition system is briefly described. The optimization method is explained, the information content and capacity are discussed and the results are presented.

[1] A US patent is applied for by NIST for the optimization method presented in this paper.

2 Recognition Systems

Artificial neural network systems are constructed as interacting subsystems with parallel data flow between the layers and parallel processing of data in each subsystem. For example, all pixels of the image are simultaneously applied to the input of the network so that all parts of the input are filtered in parallel. In fingerprint classification [10] the input is an image containing a single fingerprint. If the input is filtered, an image of the fingerprint with ridges enhanced is produced [11]. The input to the system is initially converted to a more compact representation in terms of ridge direction data; this conversion is called ridge-valley feature extraction. After the ridge-valley feature extraction is performed, a set of numbers which represents the input data in a more compact form, ridge direction data, is produced. In the next calculation the Karhunen-Loève (K-L) [12] transform is used to filter the ridge-valley data by expanding it in terms of a set of characteristic image components which are the eigenfunctions of the image covariance. This representation of the data is then used for classifying the input in each of the learned classes providing an estimate of the probability of the input being in each of the known classes. In the final calculation the input is assigned to one or more of the known classes.

2.1 Training and Testing Data

The recognition system described in this paper were trained and tested using feature vectors derived from the fingerprint images of NIST Special Database 4 [13]. This database consists of 8 bit per pixel gray level raster images of two inked impressions ("rollings") of each of 2000 different fingers. The 300 feature vectors used to train the classifiers were made from the 2000 first-rollings, and 300 feature vectors used to test the classifiers were made from the 2000 second-rollings. Every fingerprint in the database has an associated class-label, assigned by experts. The two rollings of any finger have the same class, since the class of a fingerprint is not affected by variations that occur between different rollings of the finger.

Fingerprints as they naturally occur are not distributed equally into the five classes. The estimated probabilities are .037, .029, .338, .317, and .279, for the classes Arch, Tented Arch, Left Loop, Right Loop, and Whorl respectively. The 2000 fingers represented in Special Database 4 are equally divided among the five classes. The database was produced this way, rather than by using a natural distribution, so as to increase the representation of the relatively rare, and also difficult, Arch and Tented Arch classes. This provides trainable classifiers with more examples with which to learn these problematical classes. The training set (first rollings) therefore has equally many prints of each class, so the testing set (second rollings) also has equally many prints of each class, since the database consists of rolling pairs.

2.2 KL Features

The feature extractor performs a K-L transform directly on the fingerprint image. The fingerprint image is a raster of 512 by 512 8-bit grayscale pixels, produced by scanning the fingerprint card with a CCD camera. The fingerprint classifiers described in this report take as their input a small vector of numerical features derived from a fingerprint raster image. The fingerprint is reduced to 112 features (not all of which need be used) as follows. First, it is subjected to an FFT-based filter that increases the relative power of dominant

frequencies, increasing the ratio of signal (fingerprint ridges) to noise. The local orientations of the ridges at 840 equally-spaced locations (a 28 by 30 grid) are then measured, using an orientation finder described in [11]. It computes an orientation at the location of each pixel, then averages these basic orientations in nonoverlapping 16x16-pixel squares to produce the grid of 840 orientations. The feature vector is intended to represent most of the information relevant to classification in a compact form.

3 Scaled Conjugate Gradient Network

A fully connected multilayer network was implemented on a parallel machine with 1024 processor. The weights in the network are updated for each step of iteration using the following technique.

$$\Delta \mathbf{W}_k = \alpha_k(\mathbf{g}_k + \beta_k \Delta \mathbf{W}_{k-1}) \tag{1}$$

Where β_k is calculated by the algorithm to make $\Delta \mathbf{W}_k$ and $\Delta \mathbf{W}_{k-1}$ conjugate, or orthogonal in a generalized meaning of the word. The factor α_k is often determined by some kind of line search, the line is drawn between your current position and a potential minimum in the $\mathbf{g}_k$ direction. Initially a temporary small value used for α_k to perform function evaluation. This is done to approximate the second derivative in the search direction.then we use the second derivative to select a final α_k [14].

4 Boltzmann Methods

A fully connected multi layer neural network is used as a starting network for the Boltzmann weight pruning algorithm. The network has an input layer with thirty-two input nodes, a variable size hidden layer with sixteen, thirty-two or sixty-four nodes and an output layer with ten nodes. The initial network is a fully connected network. The pruning was carried out by selecting a normalized temperature, T, and removing weights based on a probability of removal:

$$P_i = \exp(-|w_i|/T) \tag{2}$$

The values of P_i are compared to a set of uniformly distributed random numbers, R_i, on the interval $[0, 1]$. If the probability P_i is greater than R_i then the weight is set to zero. The process is carried out for each iteration of the SCG optimization process and is dynamic. If a weight is removed it may subsequently be restored by the SCG algorithm; the restored weight may survive if it has sufficient magnitude in subsequent iterations.

The dynamic effect of this is shown in figure 1 for five temperatures between 0.1 and 0.5, starting from a fully converged and fully connected network. The initial iterations removes many weights but later iterations of optimization and pruning remove fewer weights. The rate of weight removal is the function of iteration count, while the number of weights removed is related to the temperature change. The number of weights in the initial network was 1386, including bias weights. At all temperatures the initial iterations are very effective in reducing the weights. The decrease in the rate of pruning is the result of a critical phenomenon characterized by a critical temperature, T_c, at which the new information added by the SCG training balances the information removed by pruning. At this critical point networks trained

on small training sets will achieve identical testing and training accuracy even when tested on large test sets.

The effect of change in the number of hidden nodes can be seen in figures 2, 3 and 4. The change in the number of hidden units has minimal effect on the optimization process in general and the network performance changes very little. With the increase in tempreture the accuracy of the network for recognition decreases slowly for temperatures up to 0.4. With higher number of iterations the rate of weight removal slows and the rate of accuracy decay accelerates. The comparison of the training set and testing set accuracy shows that the training set accuracy is initially greater than the testing accuracy. At a critical temperature, T_c, the testing accuracy and training accuracy are identical. In figure 2, at the critical temperature of approximatly 0.58, chaotic behavior is observed in the vicinity of T_c due to the effects of weight removal. The behavior of the 32-64-10 network in figure 4 is similar to the 32-32-10 network. The 32-16-10 network in figure 3 shows an increase in the critical temperature, T_c, and a significant decrease in accuracy at T_c. This increase in T_c is caused by the reduced set of possible pruned configurations in the 32-16-10 network: the initial 32-16-10 network is too small, eventhough it has many more weights than the pruned 32-32-10 network.

5 Information Capacity Reduction

The mechanism involved in the collapse of the information capacity reduction during testing and training accuracy near T_c is caused by the large increase in weights near zero created by the most recent SCG iteration. In a given training cycle some weights are removed. If these weights are redundant they will be compensated for by other weights in the network. If these weights are critical they will be restored by the SCG optimization. The peak in the distribution near zero in both figures 5 and 6 is caused by this process. At T_c the SCG creation process is just balanced by the Boltzmann pruning.

To evaluate the generalization capability of the pruned network the network associated with a temperature $T = 0.55$ was tested. The predicted accuracy from T_c data was 75.5%; the accuracy achieved in the test was 72.6%. In this region the change in accuracy of the network is about 5% for each ΔT of 0.001 so that this agreement is consistent with an accuracy of T_c of $\pm.0005$ with a value of $T_c = 0.582$.

During this optimization process three important measures of information content are calculated [15]. The information capacity of the network, C, is given by:

$$C = N_{wts}((\log_2(|w_{max}| - \log_2(|w_{min}|) + 1))) \tag{3}$$

where N_{wts} is the number of non-zero weights, w_{max} is the weight with the largest magnitude, and w_{min} is the weight with the smallest nonzero magnitude. The distribution of information content during the optimization process are shown figures 7 and 8. The entropy is given by:

$$H = C - (\sum_{i=1}^{N_{wts}} \log_2 |w_i| + N_{wts}(1 - \log_2(w_{min}))) \tag{4}$$

and the Shannon redundancy is given by:

$$R = (\sum_{i=1}^{N_{wts}} \log_2 |w_i| + N_{wts}(1 - \log_2 w_{min}))/C \tag{5}$$

The dynamic effects of weight removal for nine temperatures between 0.001 and 0.2, starting with weights from a fully converged but unpruned network are shown in figure 9. The two curves plotted in this figure are the training set and testing set accuracy of the network. The training set accuracy is initially greater than the testing accuracy. At a critical temperature, T_c, the testing accuracy and training accuracy are identical. At the critical temperature of 0.125, by extra polating the low temperature crossing point, chaotic behavior occurs in the vicinity of T_c due to critical weight removal. For the same nine temperatures and starting with a fully converged and fully pruned network the effects of weight removal are shown in figure 10. The changes in capacity and entropy starting with a fully connected unpruned network are shown figure 11. The change in the number of weights, N_{wts}, results in capacity reduction and entropy reduction. The change in capacity starting with a fully pruned network is shown figure 12.

The effect on the information content of the network can be evaluated by examining the distribution of weights in the network as a function of temperature or by evaluation of the information capacity of the network. As the temperature is increased, the recognition accuracy of the network decreases slowly for temperatures up to 0.15. As the temperature approaches 0.2, the rate of weight removal slows, and the rate of accuracy decay accelerates. The accuracy collapse is caused by the large increase in weights near zero created by the most recent SCG iteration. In a given training cycle some weights are removed. If these weights are redundant they will be compensated for by other weights in the network. If these weights are critical they will be restored by the SCG optimization. The effect of the near-zero weights is more important when viewed as information content. The information content is approximately $\sum(\log_2 |w_i| + 1)$. When large numbers of near-zero weights exist, their contribution to the sum dominates the network information. Under these conditions the network is dominated by recently created weights that have not been optimized by SCG iterations. This lowers network accuracy without reducing genralization power of the network.

6 Conclusions

A method of network optimization has been developed which reduces the number of weights required for moderately accurate fingerprint classification by 87%. The method is based on achieving equilibrium between the information in the training set and the information capacity of the neural network by concurrent weight creation using SCG optimization and Boltzmann weight removal. These reductions allow smaller training sets and smaller classification networks to be used since the information capacity of the network and the information capacity of the training and testing sets are matched.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Botlzmann machines. *Cognitive Science*, 9:147-169, 1985.

[2] S. Kirkpatrick. C. D. Gelatt, and M. P. Vacchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[3] J. E. Moody. The effective numbers of parameters: an analysis of generalization and regularization in nonlinear systems. In R. Lippmann. editor, *Advances in Neural Information Processing System*, volume 4, pages 847–854. Morgan Kauffman, 1992.

[4] Y. Le Cun. J. S. Denker. and S. A. Solla. Optimal Brain Damage. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 396–404. Morgan Kaufman. 1990.

[5] Y. Le Cun. B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 598–605. Morgan Kaufman. 1990.

[6] I. Guyon, V. N. Vapnick, B. E. Boser, L. Y. Botton, and S. A. Solla. Structural risk minimization for character recognition. In R. Lippmann, editor, *Advances in Neural Information Processing System*, volume 4. pages 471–479. Morgan Kauffman, 1992.

[7] O. M. Omidvar and C. L. Wilson. Optimization of Neural Network Topology and Information Content Using Boltzmann Methods. In *Proceedings of the IJCNN, volume IV*, pages 594–599. June 1992.

[8] O. M. Omidvar and C. L. Wilson. Optimization of Adaptive Resonance Theory Network with Boltzman. In Dennis W. Ruck. editor. *Science of Artificial Neural Networks II*, volume 1966. SPIE, Orlando, FL, 1992.

[9] J. L. Blue and P. J. Grother. Training Feed Forward Networks Using Conjugate Gradients. In *Conference on Character Recognition and Digitizer Technologies*, volume 1661, pages 179–190. San Jose California, February 1992. SPIE.

[10] C. L. Wilson. Massively parallel neural network recognition. In *Proceedings of the IJCNN, volume III*, pages 227–232, June 1992.

[11] C. L. Wilson. G. T. Candela. P. J. Grother. C. I. Watson. and R. A. Wilkinson. Massively Parallel Neural Network Fingerprint Classification System. Technical Report NISTIR 4880, National Institute of Standards and Technology. July 1992.

[12] Anil K. Jain. *Fundamentals of Digital Image Processing*, chapter 5.8. pages 155–157. Prentice Hall Inc.. Prentice Hall International edition. 1989.

[13] C. I. Watson and C. L. Wilson. Fingerprint database. *National Institute of Standards and Technology*. Special Database 4. FPDB. April 18. 1992.

[14] M. F. Moller. A scaled conjugate gradient algorithm for fast supervised learning. Technical Report PB-339. Aarhus University. 1990.

[15] J. J. Atick. Could information theory provide an ecological theory of sensory processing? *Networks*. 3(2):213–251. 1992.

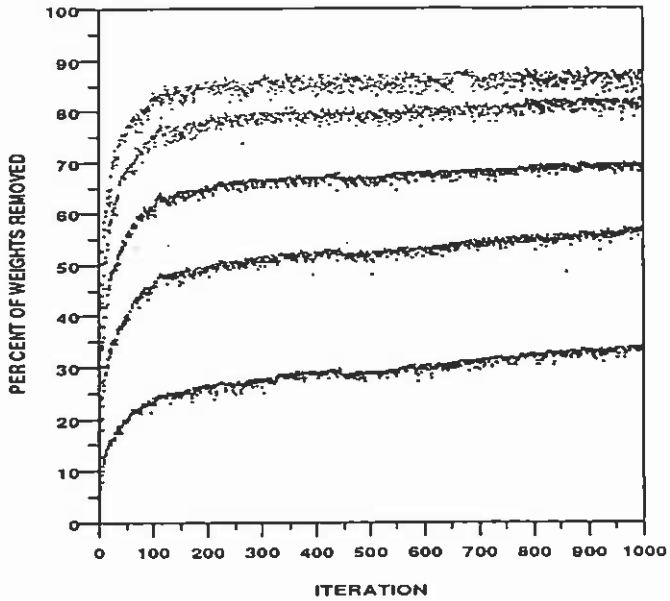

Figure 1: Weights removed as a function of iteration and temperature for $T = 0.1, 0.2, 0.3, 0.4, 0.5$. The lower curve is for $T = 0.1$; the upper curve is for $T = 0.5$.

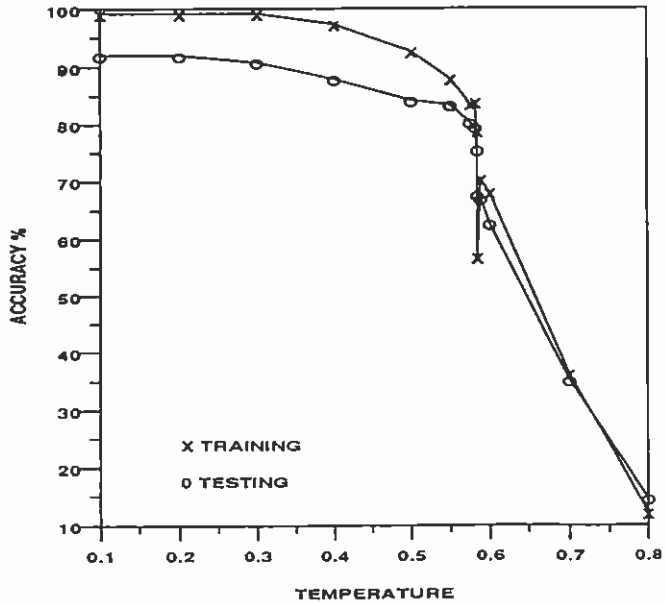

Figure 2: Change in testing and training accuracy as a function of temperature for a 32-32-10 network after 1000 iterations at each temperature.

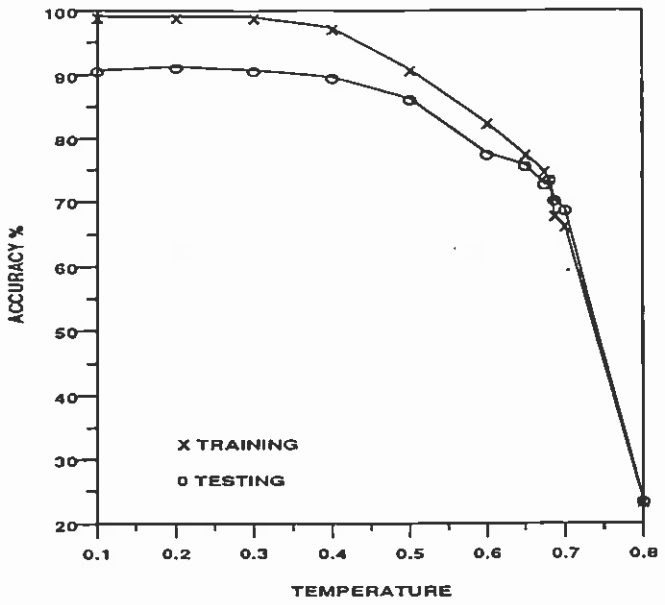

Figure 3: Change in testing and training accuracy as a function of temperature for a 32-16-10 network after 1000 iterations at each temperature.

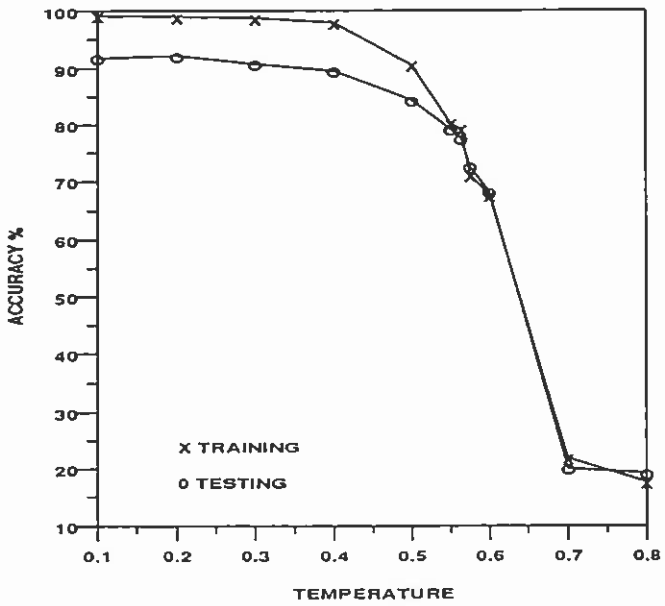

Figure 4: Change in testing and training accuracy as a function of temperature for a 32-64-10 network after 1000 iterations at each temperature.

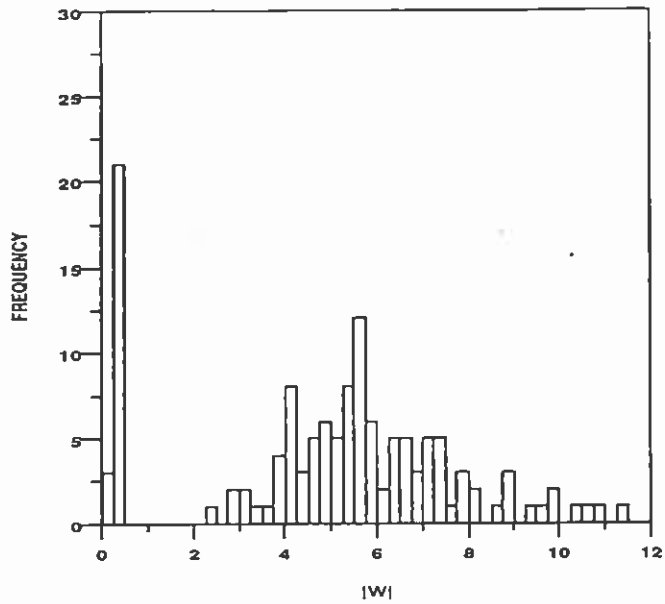

Figure 5: Weight distribution below T_c at $T = 0.55$.

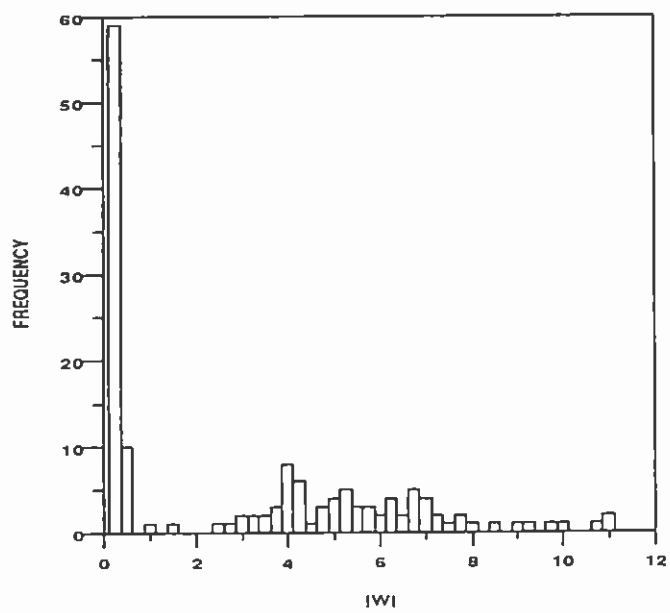

Figure 6: Weight distribution above T_c at $T = 0.6$

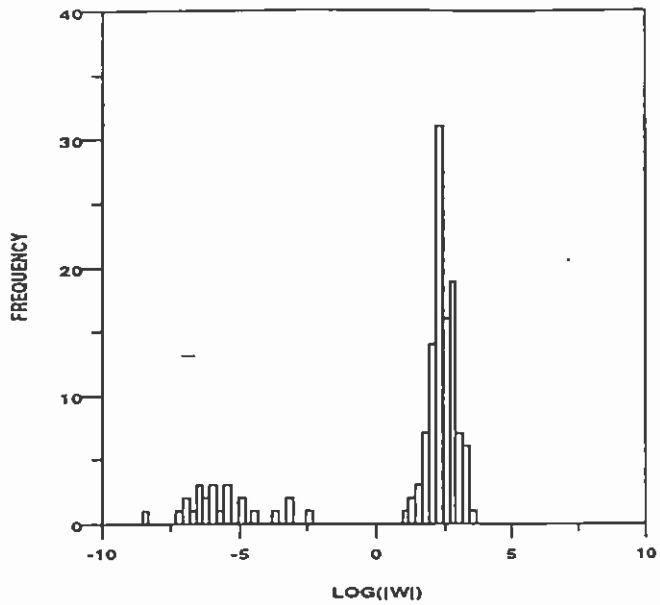

Figure 7: Information content distribution, $\sum \log_2(|w_i|)$, below T_c at $T = 0.55$.

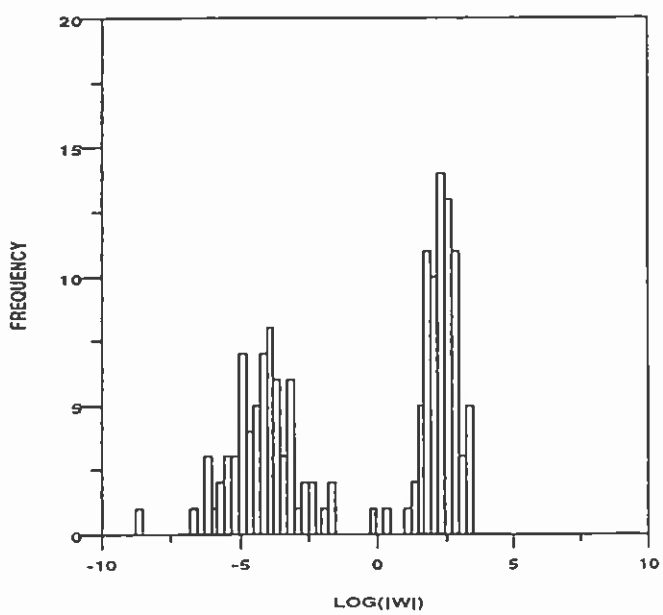

Figure 8: Information content distribution, $\sum \log_2(|w_i|)$, above T_c at $T = 0.6$.

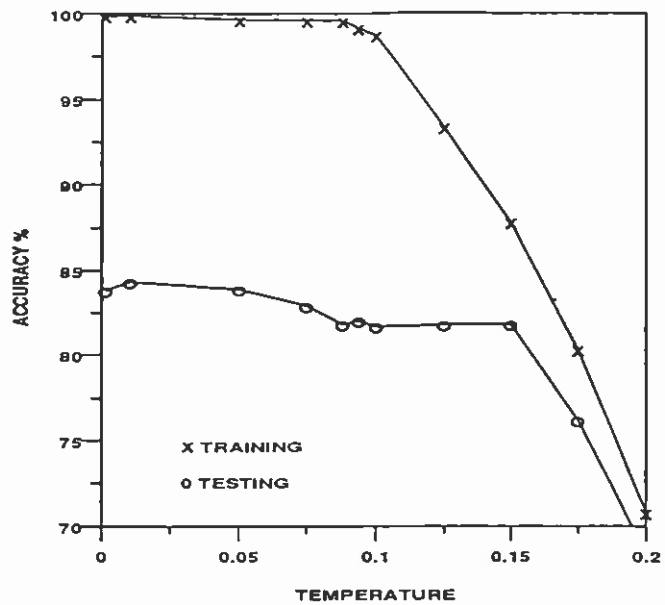

Figure 9: Network testing and training accuracy as a function of temperature for T = 0.001, 0.01, 0.05, 0.075, 0.0875, 0.9375, 0.1, 0.125, 0.15. The capacity initially was that of an unpruned network.

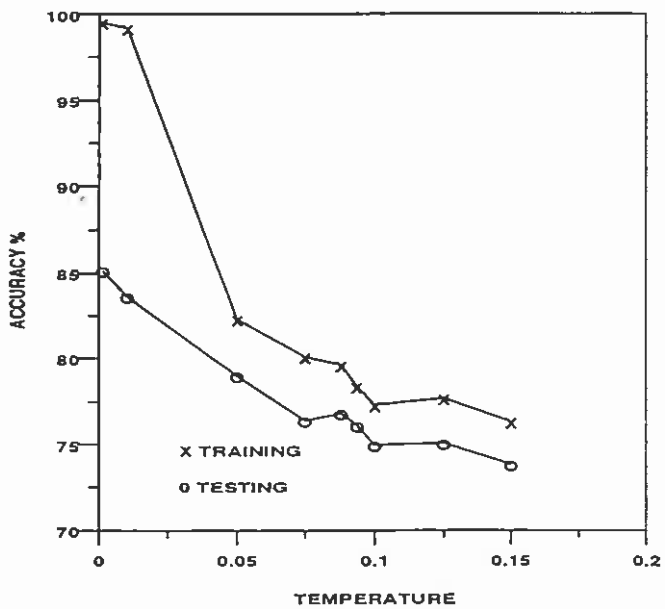

Figure 10: Network testing and training accuracy as a function of temperature for T = 0.001, 0.01, 0.05, 0.075, 0.0875, 0.9375, 0.1, 0.125, 0.15. The capacity initially was reduced by pruning the network at a temperature of 0.2.

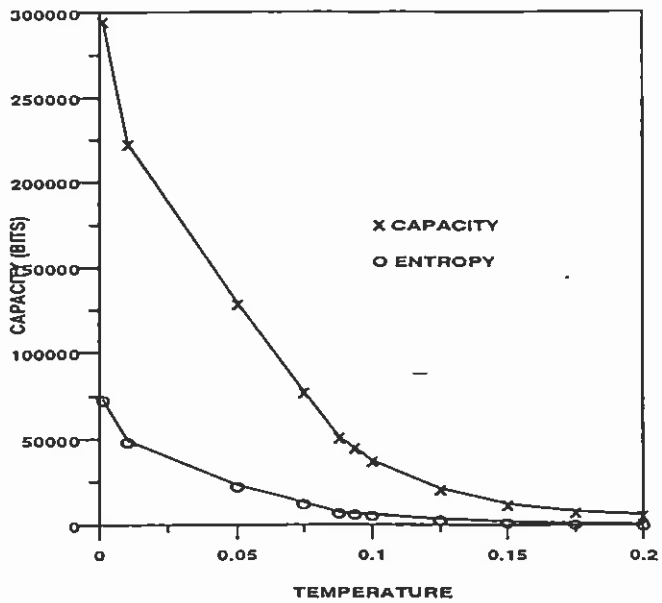

Figure 11: Change in capacity and entropy as a function of temperature for a 128-128-5 fingerprint recognition network after 300 iterations at each temperature starting with a network at $T = 0.001$.

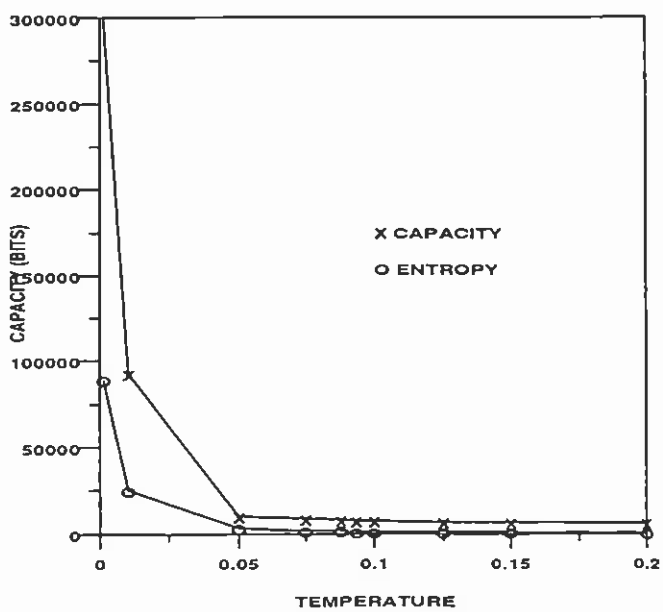

Figure 12: Change in capacity and entropy as a function of temperature for a 128-128-5 network after 300 iterations at each temperature starting with a network at $T = 0.2$.