TOPOLOGICAL SEPARATION VERSUS WEIGHT SHARING 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

Recent advances in neural networks application development for real life problems have drawn attention to network optimization. Most of the known optimization methods rely heavily on a weight sharing concept for pattern separation and recognition. The shortcoming of the weight sharing method is attributed to a large number of extraneous weights which play a minimal role in pattern separation and recognition. Our experiments have shown that up to 97% of the connections in the network can be eliminated with little or no change in the network performance.

Topological separation should be used when the size of the network is large enough to tackle real life problems such as fingerprint classification. Our research has focused on the network topology by changing the number of connections as secondary method of optimization. Our findings indicate that for large networks. topological separation yields 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 which 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

Neural net optimization research has achieved some success for real life problems. The focus of this effort has been on error minimization. A standard method of optimization for real world problems is weight sharing [1]. The weight sharing method increases the redundancy of the network while reducing the Vapnick-Chervonenkis (VC) dimension [2]. 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 network will not be stable and the generalization power of the network can not be estimated.

The optimization strategy used in this research focuses on 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 [3].

We have used the Boltzmann method as a secondary method of optimization to prune the networks used here. This method has been applied to both supervised and self organizing networks [4]. The method can be used in conjunction with a primary method of optimization such as Scaled Conjugate Gradient scheme [5]. 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.

2 Recognition Systems

Artificial neural network systems are constructed as interacting subsystems that have 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 [6] the input is an image containing a single fingerprint. If the input is filtered, an image of the fingerprint with ridges enhanced is produced. 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) 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.

The process described above is all that is necessary for classification but needs a modification to allow learning. This modification is shown in figure 1. The filter and feature extraction process remain unchanged, as does the idea of calculating class errors, but a switch is introduced into this calculation which decides if the error is low enough to allow classification.

Figure 2 illustrates a method used to perform self-organized learning. The data input path contains any required filters or feature extraction calculations. The switch used to activate learning compares the pattern to classes which have been learned and makes a decision on the basis of pattern similarity; it then either modifies known patterns using the new pattern or creates a new pattern class to accommodate the pattern. Self-organizing methods based on decision trees have existed for some time. The unique feature of the neural network methods is the parallel properties of the algorithms.

Figure 3 illustrates the method used for supervised learning. As in figure 2, data filtering and feature extraction are done at the input. The learning switch operates on a different

principle. In the supervised system, the learning switch is driven by the change in the error of the global system. The latter requires a set of data for which the correct answers are known.

A more powerful version of this method is illustrated in figure 4. It shows how replication of figure 2 and modification of the control process can be used to construct a multi-map method which can learn to classify patterns based on two or more distinct types of features. The multi-map method is of interest because it is known to occur in various kinds of biological systems. The basic building block of the method is the self-organizing structure shown in figure 2. The basic unit has been modified to allow the isolation of learning control from other processes. This unit is replicated for each multi-map subsystem but the learning controls are merged rather than replicated. This results in the structure shown in figure 4.

Previous work by Linsker [7] and by Rubner [8] has shown that simple learning rules that update a neural network without using feedback are capable of generating layers of neural processors. These maps have sensitivity profiles similar to sensitivity profiles found in the visual system of mammals and to the Gabor basis functions [9] used in [10]. This leads to the inquiry: How can these learning processes and the maps that they generate be combined directly into a self-organizing system that allows the smoothing of the input data to occur in parallel with pattern recognition and feature extraction?

The FAUST architecture provides a self-organizing method of feature extraction and classification [4]. The FAUST architecture is one of several neural networks which provide self-organizing multi-map capabilities. The structure used is a multi-map procedure similar to those known to exist in the mid-level visual cortex [11]. As in previous work [12, 13, 14] the method must provide a parallel, multi-map, self-organizing, pattern classification procedure. This is achieved using a feed-forward architecture which allows multi-map features stored in weights acting as associative memories to be accessed in parallel and to trigger a symmetrically controlled parallel learning process. A diagram of the FAUST system is shown in figure 4. This method allows features of different data types, such as binary image patterns and multi-bit statistical correlations, to be updated in parallel. This capability is provided by the parallel pattern association and relevance paths shown in figure 4 and by the existence of separate input modules for each path.

A pattern comparison method is used to form a centralized learning control which is contained in the symmetric triggering learning control block. The triggering block gates data into the learning block on the right of figure 4. This combined architecture is described by the acronym FAUST (Feed-forward Association Using Symmetrical Triggering). The three essential features of FAUST shown in this figure are: 1) Different feature classes use individual association rules in the pattern comparison blocks. 2) Different feature classes use individual learning rules as illustrated by the pattern modification blocks. 3) All feature classes contribute symmetrically to learning as illustrated by the functional symmetry of the pattern and relevance paths. The number of feature classes is shown as two in figure 4 for graphic clarity, but the architecture is not restricted to any number or type of feature classes.

3 Learning is an Optimization Problem

Assume that a set of features has been chosen and the number of hidden neurons has been selected based on the expected complexity of the regions involved. The number of inputs is the size of the set of features, and the number of output neurons is the number of classes. To specify the network completely, values must be specified for all the weights; the choice of the

weights determines how well the network classifies data. The desired criterion for choosing weights is:

> For all possible images, choose the weights so as to make the fewest mistakes in classifying.

This is both vague and impossible. A more reasonable criterion is:

> For a given *training set* of images, choose the weights so as to make the fewest mistakes in classifying.

If the the training set is sufficiently representative of the set of all the images to be classified by the network, the *test set*, this method can be adequate. However, even this method is too difficult, and the actual criterion used is usually:

> For a given *training set* of imagess, choose the weights so as to minimize the sum of squares of the errors in all the output values.

This, at last, is a fairly well-defined mathematical optimization process. It may not, however, produce the desired network, as will be seen in the next section. Accomplishing this optimization process can be done in many ways, and is a subject of ongoing research, since optimizing a function of several hundred or thousand variables is not trivial. Finding *the* optimum set of weights is usually impossible, but several different optimization attempts may be made, each starting with a different random guess for the weights. Each attempt (in general) reaches a different local minimum, and the best set of weights attained is chosen. However, the *best* set may not be the set that reduces the sum of squares of the errors to the lowest value.

4 Network Capacity Optimization

The networks, shown in figure 3 for a supervised system and in figure 4 for a self-organizing network, can be modified to concurrently optimize information content by including a self-organizing network that is used exclusively for information capacity mininzation. A network of this type is shown in figure 5. This network is placed in the learning loop of the network shown in figure 3 and this results in the network shown in figure 6. A similar network can be added to the self-organizing system as a modification of the learning control block. This modification results in the network shown in figure 7.

In both cases the additional optimization changes the objective of the optimization from:

> For a given *training set* of images, choose the weights so as to minimize the sum of squares of the errors in all the output values.

To:

> For a given *training set* of images, choose the weights so as to minimize the network information capacity and concurrently the sum of squares of the errors in all the output values.

The minimum information capacity determination is made by comparing the generalization capacity of the network with a specified entropy with the information needed to classify a *testing set* of comparable entropy.

5 Pruning and Information Capacity Reduction

A fully connected network is optimized using the Scaled Conjugate Gradient method (SCG) developed by [15] and modified by Blue and Grother [5]. The SCG method is used as a starting network for the Boltzmann weight pruning algorithm. The network has an input layer with 128 input nodes, a hidden layer with 128 nodes and an output layer with five 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:

$$P_i = \exp(-|w_i|/T)$$

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.

During this optimization process three important measures of information content are calculated [16]. The information capacity of the network, C, is given by:

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

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 magnitude. The entropy is given by:

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

and the Shannon redundancy is given by:

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

The dynamic effects of weight removal are shown in figure 8 for nine temperatures between 0.001 and 0.2 starting from a fully converged but unpruned network. The dynamic effects of weight removal are shown in figure 9 for the same nine temperatures between 0.001 and 0.2 starting from a fully converged and fully pruned network. The changes in capacity and entropy starting with a fully connected network are shown figure 10. The change in capacity starting with a fully pruned network is shown figure 11. As the size of the temperature change increases the number of weights removed initially increases, but the effect of later iterations of optimization is to decrease the rate at which weights are removed. The number of weights in the initial network was 17157, 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 phenomena 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 two curves plotted in figure 8 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. In figure 8, at the critical temperature of 0.125, read

from figure 8 by extropolating the low temperature crossing point. chaotic behavior occures the vicinity of T_c due to critical weight removal.

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. The effect of the number of weights, N_{wts}, can be seen in figure 10. which shows the capacity reduction and entropy reduction. As the temperature is increased. the recognition accuracy of the network decreases slowly for temperatures up to 0.15 as shown in figure 8. As the temperature approaches 0.2. the rate of weight removal shown in figure 10 slows. and the rate of accuracy decay accelerates.

The effect of the near-zero weights is more important when viewed as information content. The VC dimension and the information content are both 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 VC dimension.

6 Conclusions

A method of network optimization has been developed which reduces the number of weights required for moderately accurate fingerprint classification by 97%. 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 usung 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] 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 396–404. Morgan Kaufman. 1990.

[2] I. Guyon. V. N. Vipnick. 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. Morgan Kauffman. 1992.

[3] 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.

[4] C. L. Wilson. A new self-organizing neural network architecture for parallel multi-map pattern recignition - FAUST. *Progress in Neural Networks*. 4. 1992. to be published.

[5] J. L. Blue and P. J. Grother. Training feed forward nerworks using conjugate gradients. In *Proceedings Febuary 1992*. SPIE. National Institute of Standards and Technology Gaithersburg Mayland. 1992.

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

[7] R. Linsker. Self-organization in a perceptual network. *Computer*, 21:105–117, 1988.

[8] J. Rubner and K. Schulten. Development of feature detectors by self-organization. *Biological Cybernetics*, 62:193–199, 1990.

[9] J. G. Daugman. Representational issues and local filter models of 2d spatial visual encoding. In D. Rose and V. G. Dobson, editors, *Models of the Visual Cortex*, pages 96–107. J. Wiley and Sons, 1985.

[10] J. G. Daugman. Complete discrete 2-d Gabor transform by neural networks for image analysis and compression. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 36:1169–1179, 1988.

[11] A. Rojer and E. Schwatz. Multi-map model for pattern classification. *Neural Computation*, 1:104–115, 1989.

[12] L. D. Jackel, H. P. Graf, W. Hubbard, J. S. Denker, D. Henderson, and Isabelle Guyon. An application of neural net chips: Handwritten digit recognition. In *IEEE International Conference on Neural Networks, volume II*, pages 107–115, San Diego, 1988.

[13] A. Rajavelu, M. T. Musavi, and M. V. Shirvaikar. A neural network approach to character recognition. *Neural Networks*, 2:387–393, 1989.

[14] C. L. Wilson, R. A. Wilkinson, and M. D. Garris. Self-organizing neural network character recognition on a massively parallel computer. In *Proc. of the IJCNN, volume II*, pages 325–329, June 1990.

[15] M. F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, to be published.

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

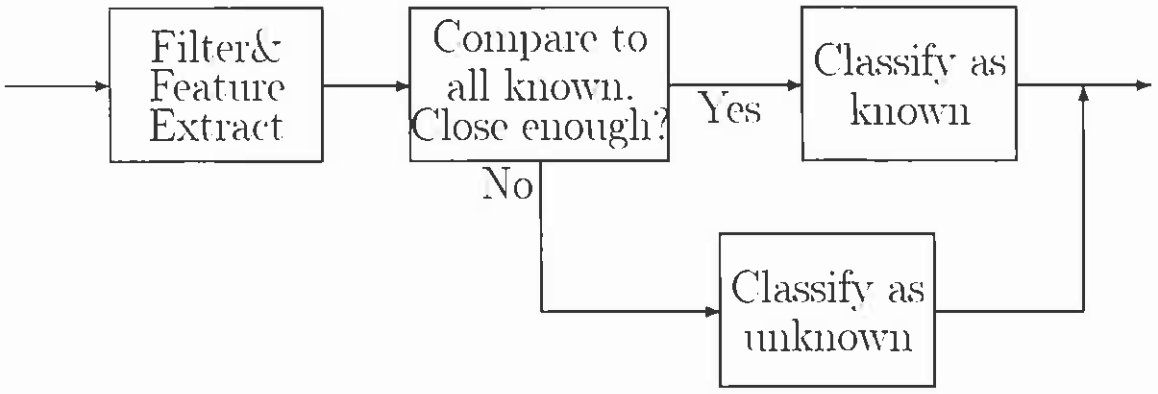

Figure 1: Data flow in a simple classification system.

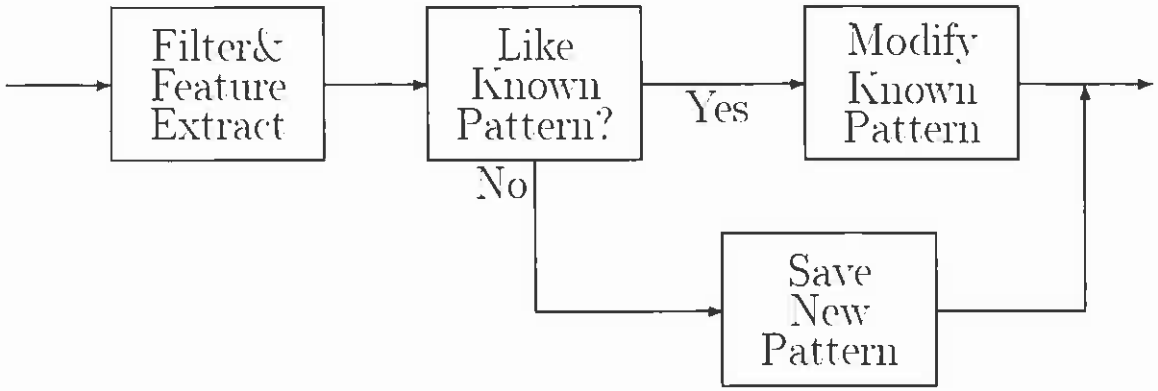

Figure 2: Data flow in a simple self-organizing system.

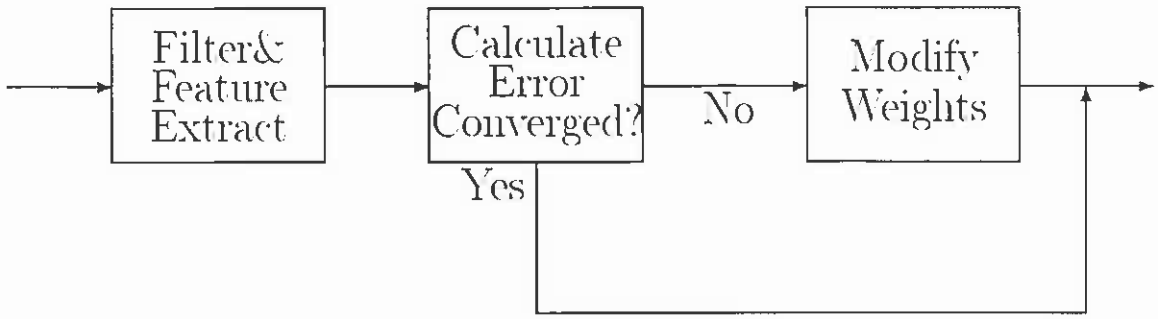

Figure 3: Data flow in a simple supervised learning algorithm.

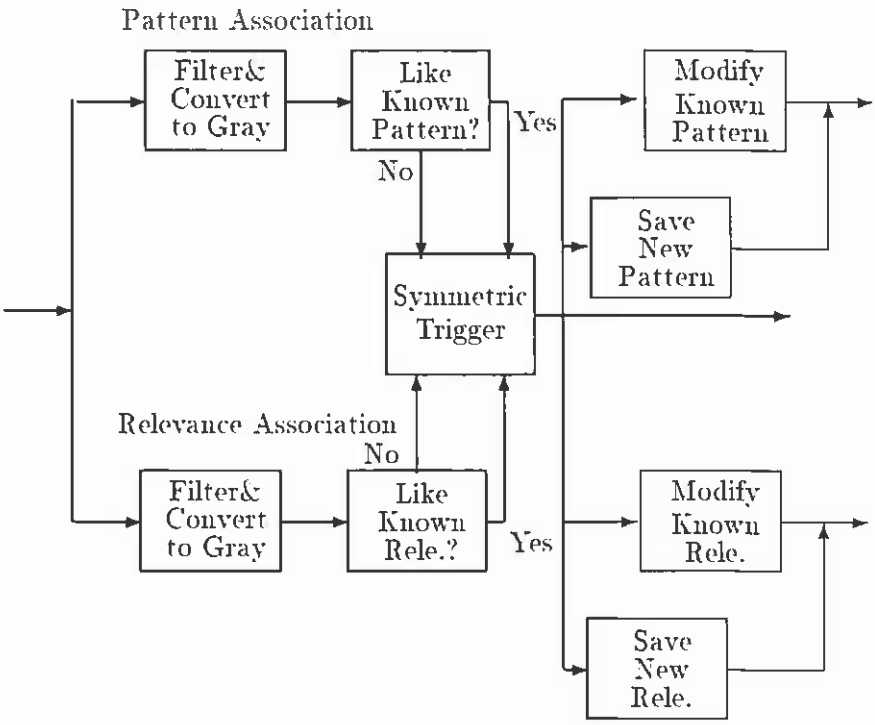

Figure 4: FAUST architecture disgram.

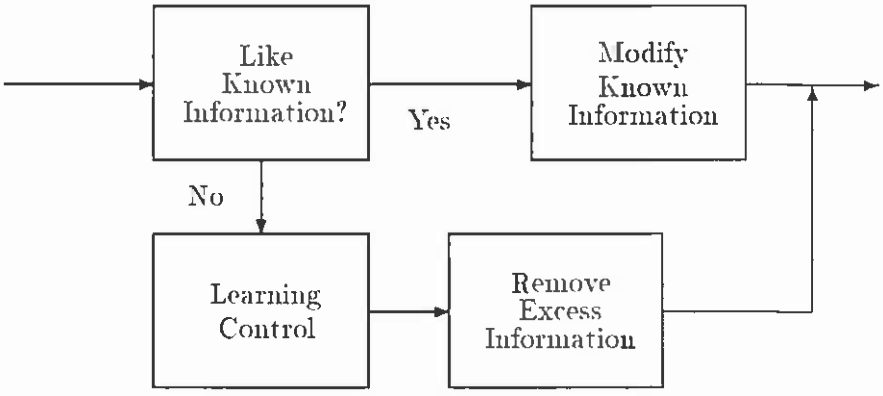

Figure 5: Data flow for a self-organizing information optimization network.

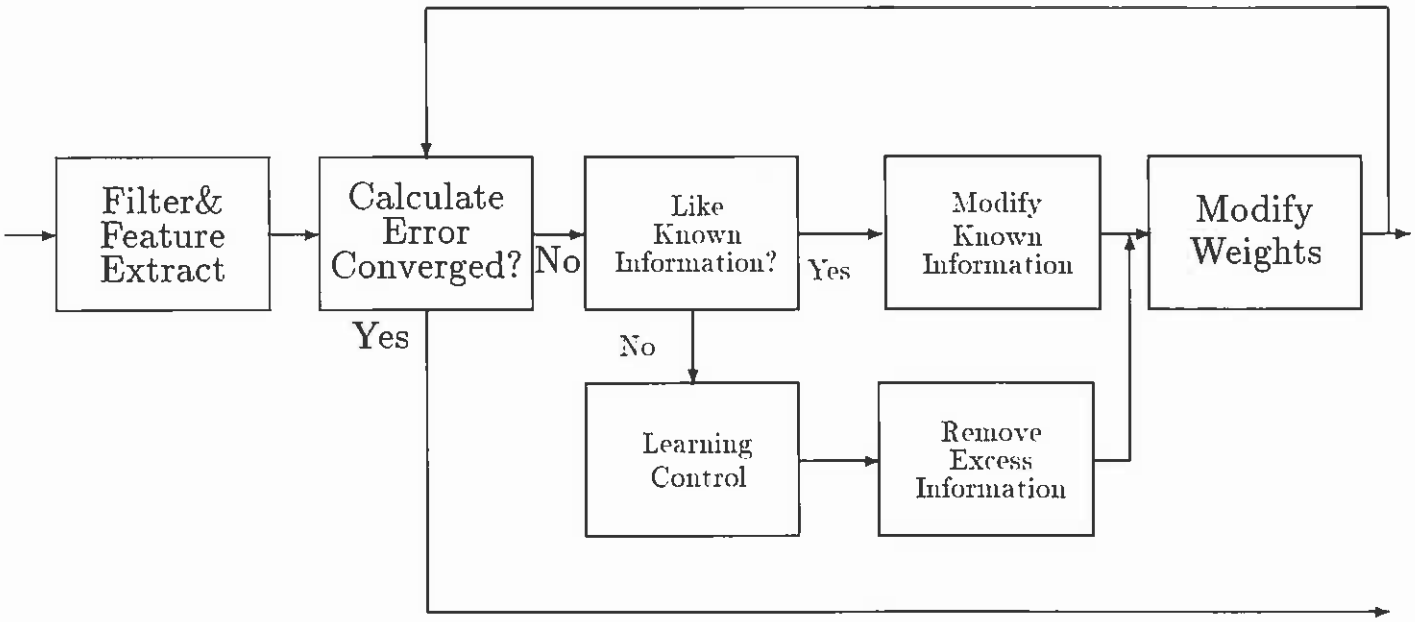

Figure 6: Data flow for size optimization in a supervised system.

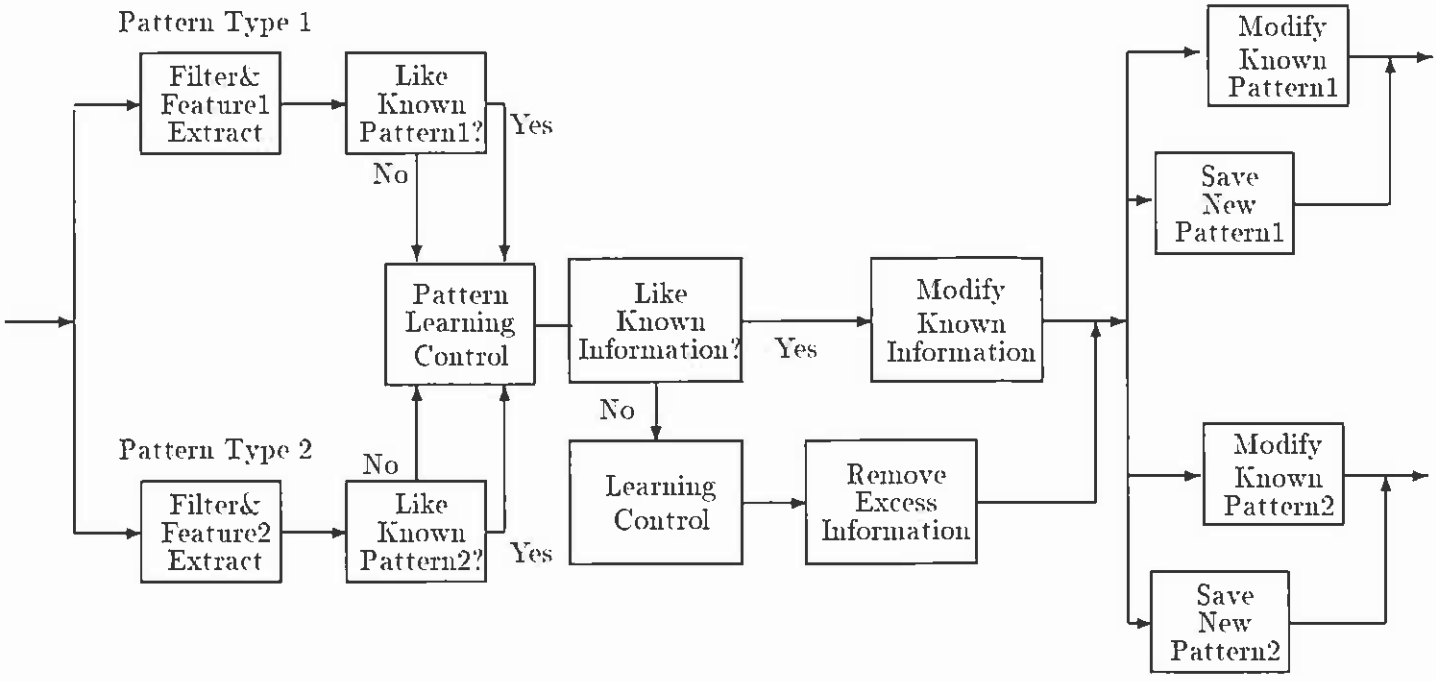

Figure 7: Data flow for size optimization in a multi-map network.

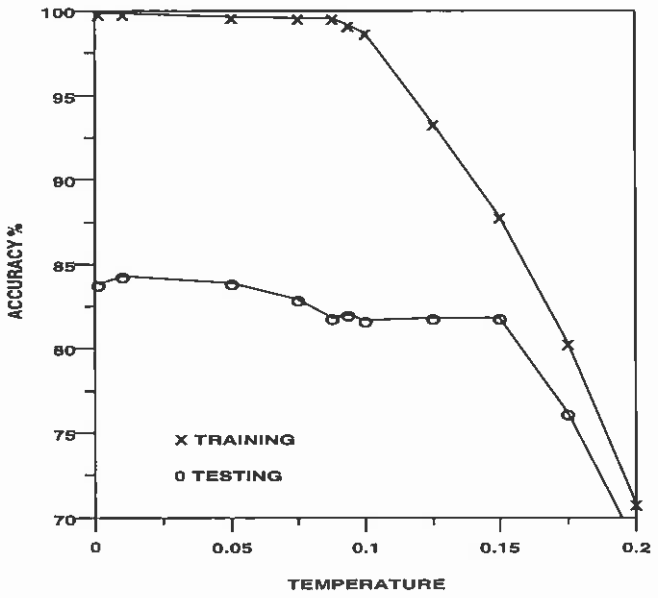

Figure 8: Network testing and training accuracy as a function of temperature for $T = 0.001,\ 0.01, 0.05,\ 0.075,\ 0.0875,\ 0.9375,\ 0.1,\ 0.125,\ 0.15$. The capacity initially was that of an unpruned network.

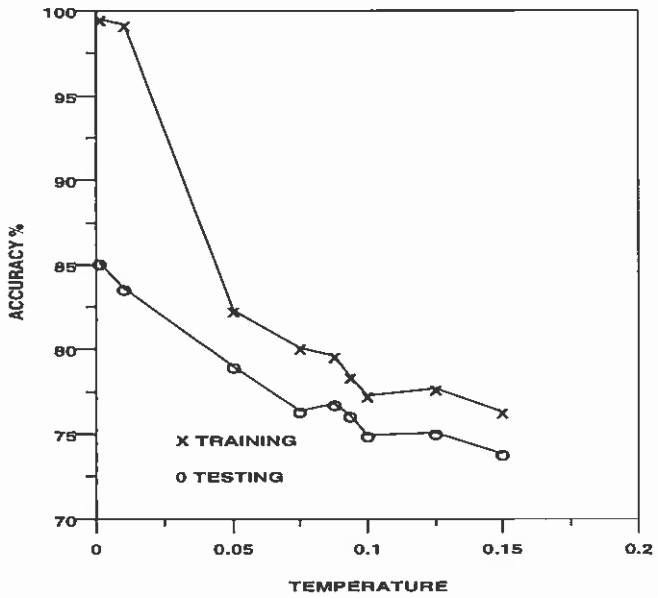

Figure 9: Network testing and training accuracy as a function of temperature for $T = 0.001,\ 0.01, 0.05,\ 0.075,\ 0.0875,\ 0.9375,\ 0.1,\ 0.125,\ 0.15$. The capacity initially was reduced by annealing the network at a temperature of 0.2.

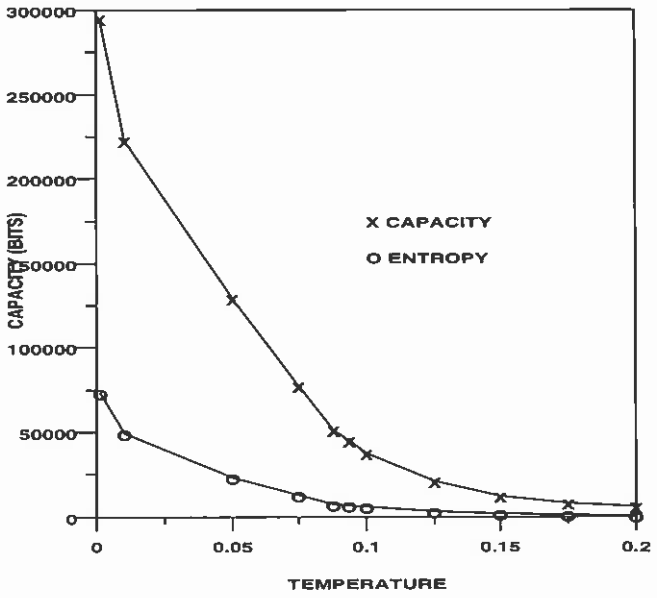

Figure 10: Change in capacity and entropy as a function of temperature for a 128-128-5 network after 300 iterations at each temperature starting with a network at $T = 0.001$.

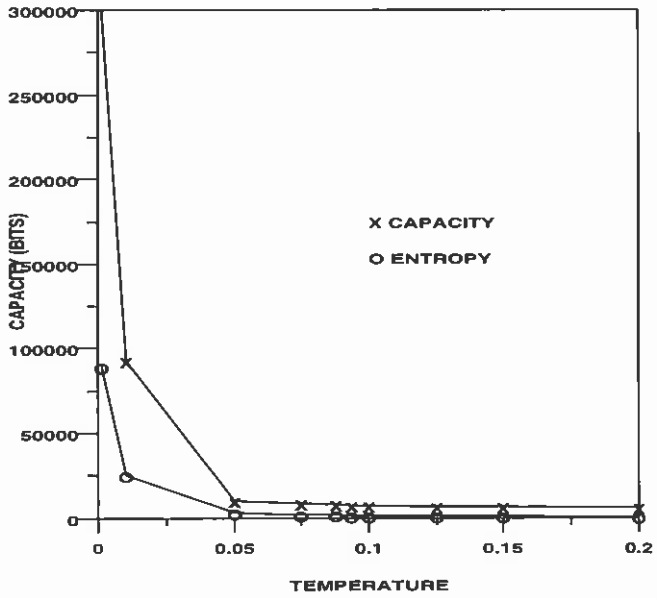

Figure 11: Change in capacity and entropy as a function of temperature for a 128-128-5 network after 300 iterations at each temperature starting with a network at $T = 0.2$.