From Proceedings of World Congress of Neural Networks - San Diego, June 5-9, 1994.

PARALLEL CONSENSUAL NEURAL NETWORKS WITH OPTIMALLY WEIGHTED OUTPUT *

J.A. Benediktsson, J.R. Sveinsson
Laboratory for Information Technology and Signal Processing
University of Iceland
Hjardarhagi 2-6, 107 Reykjavik, Iceland.

O.K. Ersoy and P.H. Swain
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47907, U.S.A.

Abstract

A recently proposed neural network architecture, the parallel consensual neural network, is applied in classification of data from multiple data sources. The parallel consensual neural network (PCNN) architecture is based on statistical consensus theory and involves using stage neural networks with either non-linearly transformed input data or different initializations for the stage networks. When non-linear transformations are applied, the input data are transformed several times and the different transformed data are used as if they were independent inputs. The independent inputs are classified using stage neural networks and the outputs from the stage networks are then weighted and combined to make a decision. Optimization methods are proposed to compute the weights for the stage networks. The given experimental results show the superiority of the optimization approach as compared to conjugate-gradient backpropagation in classification of test data.

1 Introduction

The recent resurgence of research in neural networks has resulted in the development of new and improved neural network models. These new models have been trained successfully to classify complex data. In pattern recognition applications, the question of how well neural network models perform as classifiers is very important. In previous papers [1],[2], it has been shown that neural networks compared well to statistical classification methods in classification of multisource remote sensing/geographic data and very-high-dimensional data. The neural network models were superior to the statistical methods in terms of overall classification accuracy of training data. However, statistical methods based on consensus from several data sources outperformed the neural networks in terms of overall classification accuracy of test data. Thus it would be very desirable to combine certain aspects of the statistical consensus theory approaches and the neural network models. However, it is very difficult to implement statistics in neural networks. In [3] parallel consensual neural networks (PCNNs) were proposed and implemented as stage-wise neural network algorithms. The network models in [3] do not use prior statistical information but are somewhat analogous to the statistical consensus theory approaches. In this paper the methods proposed in [3] are extended to include optimal weights for the stage networks. The paper begins with a short overview of consensus theory followed by a discussion of the PCNNs. Finally, experimental results are given.

*This research is supported in part by the Icelandic Council of Science, the National Aeronautics and Space Administration Contract No. NAGW-925 and the Research Fund of the University of Iceland

2 Consensus Theory

Consensus theory [4],[5] is a well-established research field involving procedures for combining single probability distributions to summarize estimates from multiple data sources with the assumption that the data sources are Bayesian. In most consensus theoretic methods, the data from each source are at first classified into a number of source-specific data classes [1]. The information from the sources is then aggregated by a global membership function and the data are classified according to the usual maximum selection rule into a number of user-specified information classes. The combination formula obtained in consensus theory is called a consensus rule. Several consensus rules have been proposed. Probably the most commonly used consensus rule is the linear opinion pool which has the following form for the information class if n data sources are used:

$$C_j(Z) = \sum_{i=1}^{n} \alpha_i p(\omega_j|z_i) \tag{1}$$

where $Z = [z_1, \ldots, z_n]$ is a pixel, $p(\omega_j|z_i)$ is a source-specific posterior probability and α_i's $(i = 1, \ldots, n)$ are source-specific weights which control the relative influence of the data sources. The weights are associated with the sources in the global membership function to express quantitatively our confidence in each source [4]. The linear opinion pool is simple but has several shortcomings, e.g., it is not externally Bayesian since it is not derived from class-conditional probabilities using Bayes' rule. Another consensus rule which overcomes the shortcomings associated with the linear opinion pool is the logarithmic opinion pool:

$$L_j(Z) = \prod_{i=1}^{n} (p(\omega_j|z_i))^{\alpha_i}. \tag{2}$$

The logarithmic opinion pool has performed well in classification of data from multiple sources [4].

It is desirable to implement consensus theoretic approaches in neural networks since consensus theory has the goal of combining several opinions and a collection of different neural networks should be more accurate than a single network in classification. It is important to note that neural networks have been shown to approximate class-conditional probabilities, $p(\omega_j|z_i)$, at the output in the mean square sense [6]. Using this property of neural networks it becomes possible to implement consensus theory in the networks.

3 Neural Networks with Parallel Stages

A block diagram of the parallel consensual neural network (PCNN) architecture is shown in Figure 1. Each stage neural network (SNN) has the same number of outputs neurons as the number of information classes and is trained for a fixed number of iterations or until the training procedure converges. When the training of the first stage has finished, the classification error is computed. Then another stage is created. The input data to the second stage are obtained by non-linearly transforming (NLT) the original input vectors. That stage is trained in a fashion similar to the first stage. When the training of the second stage has finished, the consensus for the SNNs is computed. The consensus is obtained by taking class-specific weighted averages of the output responses of the SNNs using source-specific weights [4], similar to the ones in equations (1) and (2). Error detection is then performed and the consensual classification error is computed. In neural networks it is very important to find the "best" representation of input data and the consensual method attempts to average over the results from several input representations or different initializations for the stages. Also, in the consensual neural networks, classification of test data can be done in parallel with all the stages receiving data simultaneously, which makes this method attractive for implementation on parallel machines.

The PCNN is self-organizing in the following sense: If the consensual classification error is lower than the classification error for the first stage, another stage is created and trained in a way similar to the second stage, but with another non-linear transformation of the input data or another initialization of the stage neural network. Stages are added in the consensual neural network as long as the consensual classification error decreases or a tolerance limit is reached. If the consensual classification error is not decreasing or is lower than the tolerance limit, the training is stopped. Using this architecture it can be guaranteed that the PCNNs should do no worse that single stage networks, at least in training. To be able to guarantee such

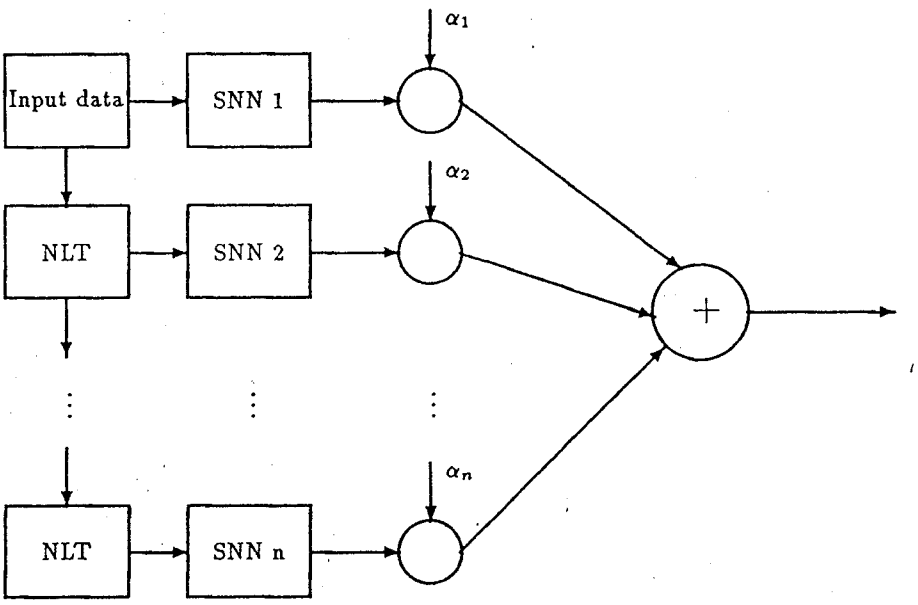

Figure 1: Parallel consensual neural network architecture

performance in classification of test data, cross-validation methods can be used. Also, it has been shown [7] that if all the networks in a collection of neural networks arrive at the correct classification with a certain likelihood $1-p$ and the networks make independent errors, the chances of seeing exactly k errors among N copies of the network is:

$$\binom{n}{k} p^k (1-p)^{n-k}$$

which gives the following likelihood of a sum of network outputs being in error:

$$\sum_{k>n/2} \binom{n}{k} p^k (1-p)^{n-k}$$

which is monotonically decreasing in N if $p < 1/2$. This implies that using a collection of networks reduces the expected classification error if the networks have equal weights and make independent errors. It has also been shown [8] that the standard deviation of the classification of a portfolio of neural networks (such as the PCNN) decreases as the number of stage networks increase.

In [3] two versions of the PCNN were proposed. Both PCNNs combine the information from separate inputs and can be considered neural network implementations of the consensus rules in equations (1) and (2). Here we concentrate on the PCNNS, the consensual neural network version of the linear opinion pool which will be referred to below as the PCNN.

Related neural network architectures to the PCNN have been proposed by Hansen and Salamon [7], Ersoy and Hong [9], Deng and Ersoy [10], Valafar and Ersoy [11], Alpaydin [12], and Nilson [13]. However, the PCNN architecture is different from all of these. It uses non-linear transformation between stages and weights the output from all the SNNs.

4 Optimal Weights

The weight selection schemes in the PCNN should reflect the goodness of the separate input data, i.e., relatively high weights should be given to input data that can be classified with good accuracy. There are at least two possible weight selection schemes. The first one is to select the weights such that they weight

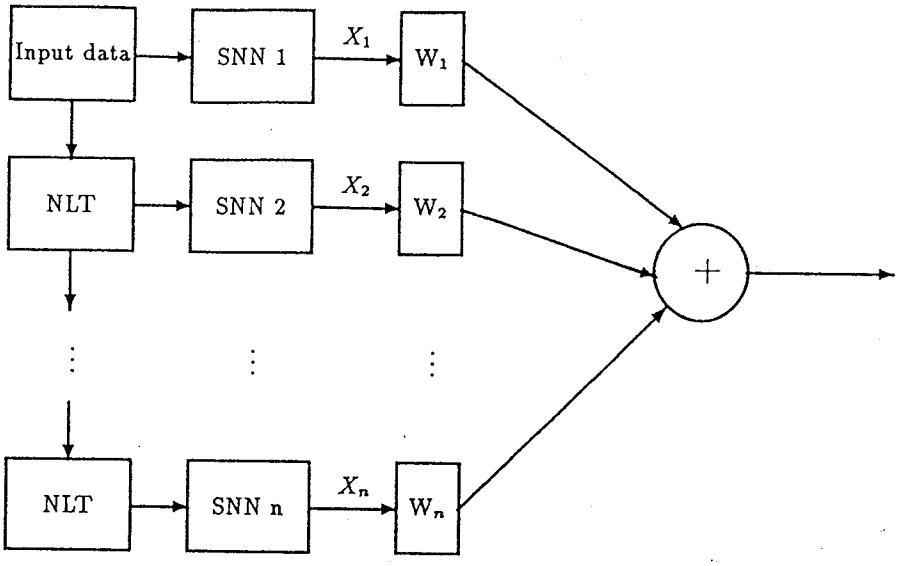

Figure 2: PCNN with weighted individual stages

the individual stages but not the classes within the stages. This scheme is shown in Figure 1. In this case one possibility is to use equal weights for all the outputs of the SNNs and effectively take the average of the outputs from the SNNs. Another possibility is to use reliability measures which rank the SNNs according to their goodness. These reliability measures are, e.g., stage-specific classification accuracy of training data, overall separability and equivocation [1].

The second scheme is to choose the weights such that they not only weight the individual stages but also the classes within the stages. This scheme is depicted in Figure 2. In the case of the PCNN the combined output response Y can be written in a matrix form as

$$Y = XW$$

where X is a matrix containing the output of all the SNNs and W contains all the weights. Assuming that X has full column rank, the above equation can be solved for W using the pseudo-inverse of X or a simple delta rule.

Let's now look at the problem of choosing the weights such that they not only weight the individual stages but also the classes within the stages. In order to find the optimal weights in Figure 2 we define

$$X = [X_1\ X_2\ \ldots\ X_n],$$

$$W = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_n \end{bmatrix}$$

where X_i, $i = 1, \ldots, n$ are $m \times p$ matrices. Each row of X_i represents an output vector of each stage network SNN i. W_i, $i = 1, \ldots, n$ are $p \times p$ matrices representing the weight of each stage network SNN i. If $Y = D$ is the desired output of the whole network we have

$$XW = D.$$

W is an unknown matrix, and its least square estimate W_{opt} is sought to minimize the square error

$$\|XW - D\|^2.$$

This is a well known problem from linear regression, signal processing and adaptive filters. The formula for W_{opt} uses the pseudo-inverse of W, i.e.,

$$W_{opt} = (X^T X)^{-1} X^T D$$

where X^T is the transpose of X, and $(X^T X)^{-1} X^T$ is the pseudo-inverse of X if $X^T X$ is non-singular. In the case that X is not of full column rank this solution becomes ill-conditioned. In that case one can use dummy augmentation to make W a full column rank matrix in a higher dimensional space and then solve the problem. There are at least two other suboptimal methods for solving the optimization problem above. The rest of this section will be denoted to these methods.

The first method is to use sequential formulas to compute the optimal W. Let the ith row vector of the matrix X be x_i^T and the ith row of the matrix D be d_i^T, then W can be calculated iteratively using the sequential formula

$$\begin{aligned} W_{i+1} &= W_i + P_{i+1} x_{i+1} \left(d_{i+1}^T - x_{i+1}^T \right) \\ P_{i+1} &= P_i - \frac{P_i x_{i+1} x_{i+1}^T P_i}{1 + x_{i+1}^T P_i x_{i+1}} \qquad i = 0, 1, \ldots, m \end{aligned}$$

where W_m is the least square estimate of W_{opt}. The initial conditions to the sequential formula are $W_0 = 0$ and $P_0 = \beta I$, where β is a positive large number.

The second method solving the least square error problem is to choose unitary W which minimizes $\|D - XW\|$. We compute

$$\|D - XW\|^2 = \|D\|^2 - 2 < D, XW > + \|X\|^2$$

where $< D, XW >= \mathrm{tr}(DW^T X^T)$ and tr returns the trace of of its argument matrices. If

$$X^T D = V \Sigma U^T$$

is a singular value decomposition (SVD) of $X^T D$ then

$$\begin{aligned} \mathrm{tr}(DW^T X^T) &= \mathrm{tr}(X^T D W^T) \\ &= \mathrm{tr}(V \Sigma U^T W^T) \\ &= \mathrm{tr}(\Sigma U^T W^T V) \\ &= \sum_{i=1}^{p} \sigma_i (X^T D) t_{ii} \end{aligned}$$

where $T = [t_{ij}] = U^T W^T V$ is a unitary matrix. This sum is maximized when all $t_{ii} = 1$, that is when $W_{opt} = V U^T$.

5 EXPERIMENTAL RESULTS

The PCNNs were used to classify a data set consisting of the following 4 data sources:

1. Landsat MSS data (4 data channels)
2. Elevation data (in 10 m contour intervals, 1 data channel)
3. Slope data (0-90 degrees in 1 degree increments, 1 data channel).
4. Aspect data (1-180 degrees in 1 degree increments, 1 data channel)

Each channel comprised an image of 135 rows and 131 columns, all channels were co-registered. The area used for classification is a mountainous area in Colorado. It has 10 ground-cover classes which are listed in Table 1. One class is water; the others are forest types. It is very difficult to distinguish between the forest types using the Landsat MSS data alone since the forest classes show very similar spectral response [1]. Reference data were compiled for the area by comparing a cartographic map to a color composite of

Table 1: Training and Test Samples for Information Classes in the Experiments on the Colorado Data Set

Class #	Information Class	Training Size	Test Size
1	Water	301	302
2	Colorado Blue Spruce	56	56
3	Mountane/Subalpine Meadow	43	44
4	Aspen	70	70
5	Ponderosa Pine 1	157	157
6	Ponderosa Pine/Douglas Fir	122	122
7	Engelmann Spruce	147	147
8	Douglas Fir/White Fir	38	38
9	Douglas Fir/Ponderosa Pine/Aspen	25	25
10	Douglas Fir/White Fir/Aspen	49	50
Total		1008	1011

the Landsat data and also to a line printer output of each Landsat channel. By this method 2019 reference points (11.4% of the area) were selected comprising two or more homogeneous fields in the imagery for each class. It has been shown [2],[3] that neural networks are sensitive to having representative training samples. In order to see how well the PCNNs compared to a backpropagation neural network with a representative training sample, the training samples were selected uniformly spaced apart in the experiments. Around 50% of the samples were used for training and the rest to test the neural networks (see Table 1). Two versions of the PCNN were applied in classification of the Colordado data, i.e., PCNN with equal weights and optimized weights. (The optimal approach reported here was the inverse method but the suboptimal methods gave similar results.) The PCNN algorithms were implemented using one-layer conjugate-gradient delta rule neural networks (CGLC) [2],[14] as its SNNs. The conjugate-gradient versions of the feedforward neural networks are computationally more efficient than conventional gradient descent neural networks. The original input data were Gray-coded but that representation has previously given the best results for this particular data set [2]. Using the Gray-code and 8 bits for each input stage expanded the dimensionality of input data to 56 dimensions. Therefore, each SNN had 57 inputs (one extra input for the bias), and 10 outputs. In these experiments the Gray-code of the Gray-code was the non-linear transformation selected. This is the same non-linear transformation used in [3]. Each SNN was trained for 200 iterations. In order to get comparison to the results of the PCNN, the single-stage conjugate-gradient backpropagation (CGBP) algorithm with two layers [14] was trained on the same data with a variable number of hidden neurons. The CGBP neural networks had 57 inputs, 8, 16, 24 and 32 hidden neurons and 10 output neurons. Eleven experiments were run for the PCNN with different numbers of stages. The highest number of SNNs used in each PCNN was fifteen. All the neural networks used the sigmoid activation function. The experiments were run on a SUN SPARCstation 10/41.

The average results of the experiments with the PCNN are shown in Figure 3 for the two weight selection schemes and the standard deviation of the training accuracy for the PCNNs is shown in Figure 4. The results with the CGBP (for different number of hidden neurons) are shown in Figure 5 as a function of the number of training iterations. From these figures it is clear that the PCNN methods outperformed the single stage CGBP in terms of classification accuracy of test data. Also, the difference between the equal weight selection and the optimal weighting method became very clear in the experiments. The optimal approach clearly outperformed the equal weighting approach in terms of training accuracy. In fact, for training data, the optimal weighting approach did show monotonically increasing overall accuracy as a function of the number of stages. This result was expected since the weights in the PCNN were optimized based on the training data. On the other hand, the PCNN methods showed very similar test accuracies after 15 stages. On the average, the optimal approach achieved 80.77% overall accuracy for test data as compared to 80.74% for the equal weighting approach. In comparison, the CGBP method achieved the maximum accuracy of 77% for test data. It is also important to note that the test results with both PCNNs are better than the

best statistical result achieved in [4].

As Figure 4 displays, the standard deviation of the classification went down as a function of the number of stages used. Overall, the PCNN results in these experiments were very satisfying.

6 CONCLUSIONS

In this paper optimized weights were computed for the stage networks in the PCNN architecture. The results obtained showed the PCNN architecture to be a desirable alternative to conjugate-gradient backpropagation for multisource classification when representative training data are available. The results for the PCNN outperformed all other methods (applied now and previously on the data set used) in terms of classification accuracy of test data. The results using the optimized weights were very promising and it is important to note that the new optimized weighting approach can also be used for the networks proposed in [11] and [12]. Although binary input data were used in the experiments, the PCNN with optimized weights works both for analog and binary input data.

At this point, the PCNNs require to be tested extensively. Different non-linear transformations and the various weight-selection schemes proposed here need to be explored more thoroughly. Also, different types of PCNN architectures are being investigated. These architectures include PCNNs with different non-linear transforms for each stage and different number of iterations for the stages. The most important remaining problem in the research concerning the PCNN architecture is the selection of the non-linear transformations. In this paper we did not concentrate on that problem but used somewhat an ad hoc method, i.e. the Gray-code of the Gray-code. Using an optimal non-linear transformation could be critical to the performance of the PCNN.

ACKNOWLEDGEMENT

The Colorado data set was originally acquired, preprocessed and loaned to us by Dr. Roger Hoffer of Colorado State University. Access to the data set is gratefully acknowledged.

References

[1] J.A. Benediktsson, P.H. Swain and O.K. Ersoy, "Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. GE-28, no. 4, pp. 540-552, July 1990.

[2] J.A. Benediktsson, P.H. Swain and O.K. Ersoy, "Conjugate-Gradient Neural Networks in Classification of Multisource and Very- High Dimensional Remote Sensing Data," *International Journal of Remote Sensing*, vol. 14, no. 15, pp. 2883-2903, October 1993.

[3] J.A. Benediktsson, J.R. Sveinsson, O.K. Ersoy and P.H. Swain, " Parallel Consensual Neural Networks", *Proceedings of the 1993 IEEE International Conference on Neural Networks*, vol. 1, pp. 27-32, San Francisco, 1993.

[4] J.A. Benediktsson and Philip H. Swain, "Consensus Theoretic Classification Methods," *IEEE Transactions on Systems Man and Cybernetics*, vol. 22, no. 4, pp. 688-704, July/August 1992.

[5] C. Genest and J.V. Zidek, "Combining Probability Distributions: A Critique and Annotated Bibliography," *Statistical Science*, vol. 1. no. 1, pp. 114-118, 1986.

[6] D. W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, and B.W. Suter, "The Multilayer Perceptron as an Approximation to a Bayes Optimal Discrimination Function," *IEEE Transactions on Neural Networks*, vol. 1, no. 4, pp. 296-298, 1990.

[7] L.K. Hansen and P. Salamon, "Neural Network Ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993-1001, 1990.

[8] G. Mani, "Lowering Variance of Decisions by Using Artificial Neural Network Portfolios," *Neural Computation*, vol. 3, pp. 484-486, 1991.

[9] O.K. Ersoy and D. Hong, "Parallel, Self-Organizing, Hierarchical Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 2, pp. 167-178, 1990.

[10] S-W. Deng and O.K. Ersoy, "Parallel, Self-Organizing, Hierarchical Neural Networks with Forward-Backward Training," *Circuits, Systems and Signal Processing*, vol. 12, no. 2, 1993.

[11] H. Valafar and O.K. Ersoy, Parallel, Self-Organizing, Consensual Neural Network, Report No. TR-EE 90-56, School of Electrical Engineering, Purdue University, 1990.

[12] E. Alpaydin, "Multiple Networks for Function Learning", *Proceedings of the 1993 IEEE International Conference on Neural Networks*, vol. 1, pp. 9-14, San Francisco, 1993.

[13] N. Nilsson, Linear Machines, McGraw-Hill, New York, 1965.

[14] E. Barnard, "Optimization for Training Neural Nets,"*IEEE Transactions on Neural Networks*, vol. 3, no. 2, pp. 232-240, March 1992.

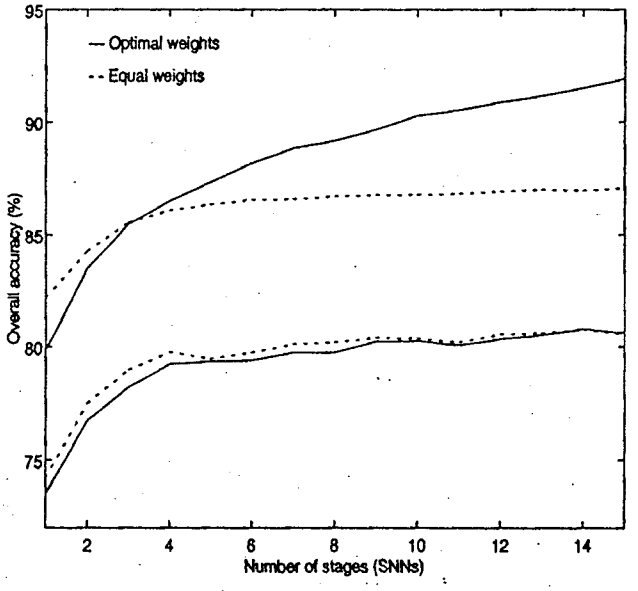

Figure 3: Average results for the PCNN with equal and optimal weights as a function of the number of SNNs. The upper curves represent training results and the lower curves test results.

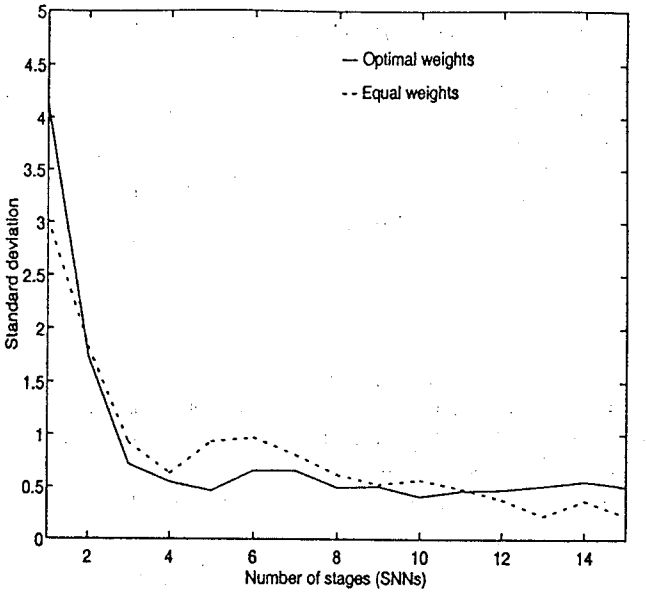

Figure 4: Standard deviation for the training results of the PCNN methods.

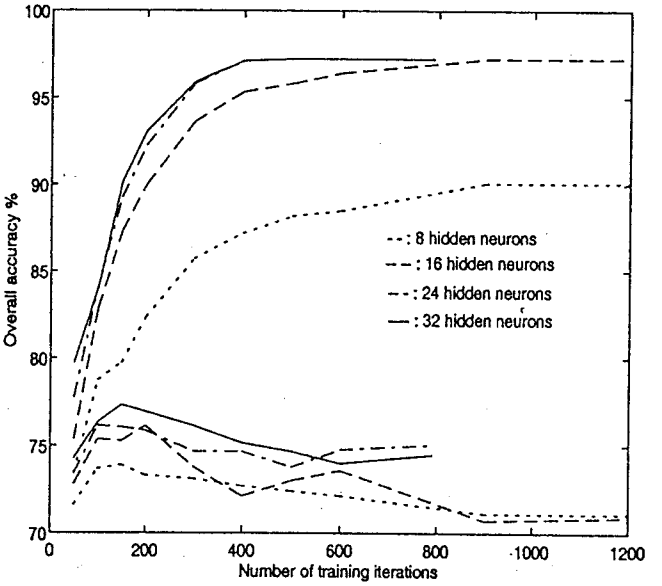

Figure 5: Experimental results for the CGBP with a variable number of hidden neurons. The upper curves represent training results and the lower curves test results.