Binary Decision Clustering for Neural Network Based OCR

C. L. Wilson
P. J. Grother
and
C. S. Barnes

National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

A neural network method of handprint character recognition is presented which consists of an input network which is trained to make binary decisions on character classes ie., to distinguish a "1" from a "0"," and an output network which combines the signals from the input networks into a digit recognition decision. For a ten digit OCR problem this results in 45 binary decision machines (BDMs) in the input network. The output of these machines are connected to an output structure which is trained separately to provide the character recognition decision. The neural network classifiers used in these input and output networks were multi-layer perceptrons (MLP), radial basis function networks (RBF), and probabilistic neural networks (PNN). A simple majority vote rule was also tested in place of the output network. The system was tested on OCR data consisings of 7,480 digit images for training and 23,140 digit images for testing. K-L tranforms were used for each BDM to transform the input images into feature vectors. Similar accuracy was obtained from several different combinations of neural network input and output structures. Minimum classification error obtained was 2.5%. The best reject accuracy performance was obtained by combining a PNN input structure with a RBF output structure. This combined network had an error rate of 0.7% with 10% rejection.

keywords: OCR, neural networks, pattern recognition, K-L transform, dynamic systyems.

1 Introduction

In a previous study, the accuracies of statistical and neural network OCR methods were compared [1]. In this study, methods which used clustering to generate an initial state for learning or statistical analysis such as RBF or Qudratic Minimum Distance (QMD) had consistently better performance than MLP based methods. Methods that were local and used no learning, PNN and KNN, had the best performance. This paper presents a clustering method based on the construction of binary decision machines. BDMs. which allows MLP based methods to become as accurate as the best method presented in the previous study. Clustering has been presented as an essential component in many biologically based methods. ART [2. 3. 4] clusters data using a leader clustering method [5] prior to learning. DYSTAL [6] clusters data into patches during the learning process. cluster formation is a critical element of the learning discussed in [7. 8]. and FAUST [9] uses clustering of data to selectively control the learning process. In MLP based character recognition. the weight sharing methods which have been used [10. 11. 12. 13] provide a method that effectively clusters the input feature space. In neighbor based methods. the local intrinisic dimensionality [14] has been recognized as a critical factor in the pattern recognition capabilities of these methods.

In this paper we present an alternate method for clustering the feature data for OCR which is more readily adapted to vector based feature sets than weight sharing is and which can give high accuracy classification with only a simple winner take all voting method as an outpout process. When a more complex combination of input and output networks is used the error reject performance of BDM based OCR is comparable to the best systems presented at the First Census OCR Systems Conference [15] and the improved PNN system in the NIST form based OCR system [16]. A simple analysis of the speed of the various methods shows that the run time in a serial computers for PNN BDM recognition increases by a factor of nine since each prototype is used nine times and that the cost of MLP and RBF methods is typically increased by a factor of about 22 since half as many features are required for 45 machines. In a parallel system with 45 processors all operations could be carried out in parallel in all methods so the smaller size of the BDM networks would result in a 2-5 times speed up in recognition.

In [1] and [15] many different OCR systems were presented which achieved 5% error rates and several were presented which has 2%-3% error rates. By reducing the problem to a series of BDMs we show that several different neural network methods can achieve 3% error rates. These systems also can exhibit error reject behavior comparable to the best presented in [15]. Analysis of the digit by digit performance with respect to feature set size and network type will show that much of this improvement is associated with local rank reduction in the feature set.

The next section of the paper the network structures of the input and output networks are described as are the specific classifiers. In section 3 the training and testing data is described. In section 4 the classification accuracy. digit recognition accuracy. and reject accuracy of the system are described. In section 5 the results of the experiments are discussed and conclusions are drawn.

2 Network Structures

2.1 K-L Network Input

The Karhunen Loève expansion of digit images is used as reduced dimensionality optimally compact representation for the BDMs. The use of such features in OCR has been described in. for example [17] [18] [1]. The handwritten binary characters are size and orienation normalized and represented as the ±1 elements of a vector by some consistent ordering of the square image. The mean vector of P such images is subtracted from each and an ensemble matrix. $\mathbf{U}$ is formed with these P vectors as its columns. The symmetric covariance matrix. $\mathbf{R}$. gives the mean of of all the interpixel correlations.

$$\mathbf{R} = \frac{1}{P}\mathbf{U}\mathbf{U}^T \tag{1}$$

The covariance matrix $\mathbf{R}$ has eigenvectors as the columns of Ψ defined as:

$$\mathbf{R}\Psi = \Psi\Lambda \tag{2}$$

where the only non zero elements of Λ are the eigenvalues on its diagonal. The eigenvectors are the directions of maximum variance in the image space and form a complete orthonormal basis. They are the principal axes of a hyperellipse in that space. The eigenvalues define the statistical "length" of these axes: thus the first column of Ψ corresponding to the largest eigenvalue is the major axis. The eigensolution of the covariance matrix provides an ordered variance expansion of the image ensemble. The latter eigenvectors, describing very little variance in the images, are discarded thus affording reduced dimensionality.

The Karhunen Loève transforms, $\mathbf{V}$, are just the projection of the zero mean images onto the principal axes:

$$\mathbf{V} = \Psi^T\mathbf{U} \tag{3}$$

2.2 Input and Output Networks

The Network is divided into two sections, the Input Network and the Output Network. The Input Network consists of 45 BDMs using two digit pairs for each classifier. Each BDM is trained on two digits which are distinct for that BDM. Each class of digits is used for the training of 9 machines in combination with the other nine digits with no combination being repeated. The training and testing of the Input Network was done using 8 to 32 K-L features in increments of 4 features. The decision machines are made using the MLP, the type 1 RBF, type 2 RBF, and the PNN. We use two different kinds on MLPs. MLP1 uses a distinct target value for each digit class. The topology for the MLP2 classifier has been modified to eliminate the second target value in the output, leaving only one target value. The modification causes the output of the MLP2 BDM to have only one resulting signal. This is used to select one of the training digit pairs. The RBF1 and RBF2 classifiers use 2 cluster for each class in the Input Network. The results of the Input Network are normalized for MLP and RBF classifiers. The PNN classifier results are the logarithms of absolute signals.

The Output Network uses the results from the Input Network and converts the results using a voting rule or one of several different types of networks. The network types used in the Output Network are PNN, MLP, RBF1 and RBF2. The 24 feature PNN BDMs Output Network for RBF1 were tested over 1 to 6 starting RBF values, and the rest of the Output Networks RBF1s and all the RBF2s were tested over 3 to 6 starting clusters.

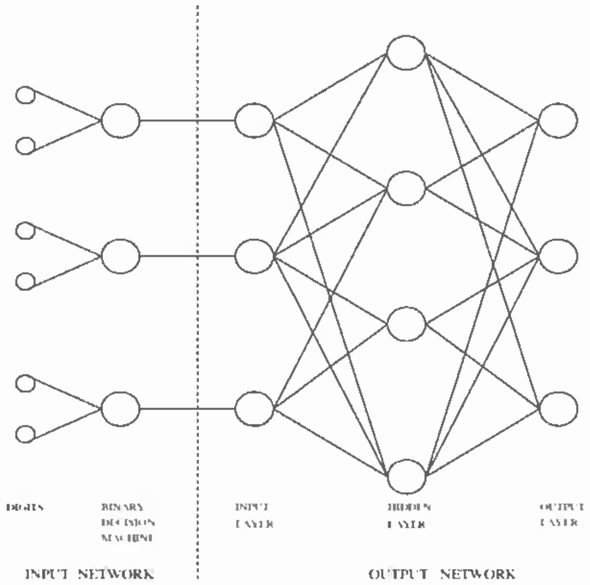

Figure 1: Diagram of the Relation of Input Network and Output Network

Figure 1 shows the basic form of the BDM Network. The dashed line is the division between the Input Network and the Output Network. The output signals of the Input Network are combined in a set order and become the input signals to the Output Network. In the Output Network, the hidden layer is used in the MLP and the two RBF types, but not in the PNN network.

2.3 Classifiers

Four different types of classifiers, MLP, RBF1, RBF2, and PNN, were used in the experiment.

2.3.1 Multi-Layer Perceptron

This classifier is also known as a feedforward neural net. We have used an MLP with three layers (counting the inputs as a layer). It will be convenient to define the following notation:

$$\begin{aligned}
N^{(i)} &= \text{number of nodes in } i^{\text{th}} \text{ layer } (i = 0, 1, 2),\ N^{(0)} = n,\ N^{(2)} = L \\
f(x) &= 1/(1 + e^{-x}) = \text{sigmoid function} \\
b_i^{(k)} &= \text{bias of } i^{\text{th}} \text{ node of } k^{\text{th}} \text{ layer } (k = 1, 2) \\
w_{ij}^{(k)} &= \text{weight connecting } i^{\text{th}} \text{ node of } k^{\text{th}} \text{ layer to } j^{\text{th}} \text{ node of} \\
&\quad\ (k-1)^{\text{th}} \text{ layer } (k = 1, 2; 1 \le i \le N^{(k)}; 1 \le j \le N^{(k-1)})
\end{aligned}$$

The discriminant functions are then of the form

$$D_i(\mathbf{x}) = f\left(b_i^{(2)} + \sum_{j=1}^{N^{(1)}} w_{ij}^{(2)} f\left(b_j^{(1)} + \sum_{k=1}^{N^{(0)}} w_{jk}^{(1)} x_k\right)\right).$$

For the training of the weights of this network, a reasonable procedure is to use an optimization algorithm to minimize the mean-squared-error over the training set between the discriminant values actually produced and "target discriminant values" consisting of the appropriate strings of 1's and 0's as defined by the actual classes of the training examples. For example, if a training feature vector is of class 2, then its target vector of discriminant values is set to (0, 1). It is more feasible to minimize this kind of an "error function" than to attempt to directly minimize the number of incorrectly classified training examples, since the latter number will take on only relatively few values and is a discontinuous "step function". The error function is modified by the addition of a scalar "regularization" term [19]. This equals a tunable constant, λ, multiplied by the mean square weight, $\overline{w_{ij}^2}$. This term prevents large weights which are associated with overtraining i.e. the overfitting of the weights to the training data. This has been shown to increase the generalization ability of the network [20].

Networks of the MLP type are the most commonly used "neural nets" in use today, and they are usually trained using a "backpropagation" algorithm [21]. A "scaled conjugate gradient" training method [22, 23, 24, 20] has been used in our research instead of the ubiquitous backpropagation method, training speed gains of an order of magnitude being typical.

2.3.2 Radial Basis Functions

Neural nets of the RBF type get their name from the fact that they are built from radially symmetric Gaussian functions of the inputs. Actually, the RBF nets discussed here use

Gaussian functions that are more general than radially symmetric functions: their constant potential surfaces are ellipsoids whose axes are parallel to the coordinate axes, whereas radially symmetric Gaussian functions have spherical constant potential surfaces. However, the name RBF has become customary for any neural net that uses Gaussian functions in its first layer.

We have experimented with RBF networks of two types, which will be denoted RBF1 and RBF2. The following notation will be convenient:

$N^{(i)}$ = number of nodes in i^{th} layer ($i = 0, 1, 2$)

$\mathbf{c}^{(j)}$ = center vector of j^{th} hidden node ($1 \le j \le N^{(1)}$) ($\mathbf{c}^{(j)} \in \mathbf{R}^n$) $= (c_1^{(j)}, \ldots, c_n^{(j)})^{\mathrm{T}}$

$\boldsymbol{\sigma}^{(j)}$ = width vector of j^{th} hidden node ($1 \le j \le N^{(1)}$) ($\boldsymbol{\sigma}^{(j)} \in \mathbf{R}^n$) $= (\sigma_1^{(j)}, \ldots, \sigma_n^{(j)})^{\mathrm{T}}$

$b_j^{(k)}$ = bias to the j^{th} node of the k^{th} layer

$f(x)$ = $1/(1 + e^{-x})$ = sigmoid function

w_{ij} = weight connecting i^{th} output node to j^{th} hidden node ($1 \le i \le N^{(2)}; 1 \le j \le N^{(1)}$)

Each hidden node computes a radial basis function. For RBF1, these functions are unbiased exponentials

$$o_j(\mathbf{x}) = \exp\left(-r^2(\mathbf{x}, \mathbf{c}^{(j)}, \boldsymbol{\sigma}^{(j)})\right),$$

and for RBF2, they are of the biased sigmoidal form

$$o_j(\mathbf{x}) = f\left(-b_j^{(1)} - r^2(\mathbf{x}, \mathbf{c}^{(j)}, \boldsymbol{\sigma}^{(j)})\right).$$

For either type of RBF, the i^{th} discriminant function is the following function of the radial basis functions:

$$D_i(\mathbf{x}) = f\left(b_i^{(2)} + \sum_{j=1}^{N^{(1)}} w_{ij} o_j(\mathbf{x})\right).$$

The centers $\mathbf{c}^{(j)}$, widths $\boldsymbol{\sigma}^{(j)}$, hidden-node bias weights $b_j^{(1)}$ (RBF2 only), output-node bias weights $b_i^{(2)}$, and output-node weights w_{ij} may be collectively thought of as the trainable "weights" of the RBF network. They are trained initially using the cluster means (from a "K-means" algorithm applied to the prototype set) as the center vectors $\mathbf{c}^{(j)}$. The width vectors $\boldsymbol{\sigma}^{(j)}$, are set to a single tunable positive value. More sophisticated methods of determining RBF parameters may be found in [25] [26]. The output layer weights are set such that each output node is connected with a positive weight to hidden nodes of its class (that is, hidden nodes whose initial center vectors are means of clusters from its class), and connected with a negative weight to hidden nodes of other classes. Training proceeds by optimization identical to that described for the MLP.

2.3.3 Probabilistic Neural Net

This classifier is proposed in a recent paper by Specht [27]. Each training example becomes the center of a kernel function which takes its maximum at the example and recedes gradually as one moves away from the example in feature space. An unknown $\mathbf{x}$ is classified by computing, for each class i, the sum of the values of the class-i kernels at $\mathbf{x}$, multiplying these numbers by compensatory factors involving the estimated *a priori* probabilities. and picking the class whose resulting discriminant value is highest. Many forms are possible for the kernel functions: we have obtained our best results using radially symmetric Gaussian kernels. The resulting discriminant functions are of the form

$$D_i(\mathbf{x}) = \frac{\hat{p}(i)}{M_i} \sum_{j=1}^{M_i} \exp\left(-\frac{1}{2\sigma^2} d^2\left(\mathbf{x}, \mathbf{x}_j^{(i)}\right)\right),$$

where σ is a scalar "smoothing parameter" that may be optimized by trial and error.

3 Test and Training Data

The classifiers described in this paper were trained and tested using feature vectors derived from the digit images of NIST Special Database 3 [28]. This database consists of binary 128 by 128 pixel raster images segmented from the sample forms of 2100 writers published on CD as [29]. Other results on segmentation and recognition of this database have been reported [30]. The relative difficulties of the NIST OCR databases have been discussed in [31]. For this study samples are drawn randomly from the first 250 writers to yield a training set of 7480 digits with *a priori* class probabilities all equal to 0.1. Even for digits, depending on the application, certain classes may be more prevalent; in banking tasks, for example, "0" is more common. The test set is similarly constructed from the second 250 writers yielding 23140 samples. The images are size normalized by pixel deletion, stroke width bounded by binary erosion and dilation, and consistent orientation is effected by shearing rows by an amount determined by the leftmost and rightmost pixels in the first and last rows defining a vertical line. The resulting image is 32 pixels high and its width is less than or equal to its height. Covariance matrices are produced for each of the training sets and the first 32 eigenvectors are used. The Karhunen-Loeve (K-L) transform is performed to extract the principle features for 8 to 32 features for the training sets in four feature increments [32]. The training and testing data are identical to the data used in [1].

4 Results

4.1 K-L Feature Extraction

One of the advantages of applying the K-L transform to the BDM data set is that insight into problem difficilty can be obtained directly from the KNN classification accuracy for each BDM as shown in table 1 and by ploting the first two K-L features of some typical problems. The problems chosen were separation of "0" and "1", separation of "6" and "8" and separation of "3" and "8". The "0-1" problem is a visually simple canonical easy prpblem [33], the "6-8" problems is the one which involves the easy ("6") and hard ("8") digits on the next subsections, and the "3-8" problem is one that can be difficult on the test data set even for humans.

Examination of table 1 shows that when optimal feature direction is uses, one K-L feature, even the hard problem can be solved with 10.6% error and the erros of the two easy problems are les than 5%. This demonstrates that very simple low dimensional methods can produce results comparable to those found on the global problem in [1, 15] by complex networks.

There is also substantial difference in error rates for the easy and hard problems for any number of features. The easy problems, "0-1" and "6-8", start with 4% error and fall to 0.4% error. The hard "3-8" problem starts at 10.6% erroe and never falls below 1.44% error. All three ploblem have reached their optimim performance using 32 features well below most of the optimum feature levels in [1].

Examination of figure 2 shows that most of the "0" and "1" points are clearly separated but their are some near neighbor points of the opposite class. The data in table 1 shows that these points are usually separated, if they can be separated, using 5 features. Examination of figure 3 shows that most of the "6" and "8" points are clearly separated and their are few near neighbor points of the opposite class. The data in table 1 shows that these points are wel separated using 2 features. The simplification of this problem using two features shows why it is an easy problem at low dimension but table 1 also shows that at 5 features it is intermediate in difficulty between "0-1" and "3-8". Examination of figure 3 shows that most of the "3" and "8" points are not separated; their are many near neighbor points of the opposite class. The data in table 1 shows that these points are never separated as well as they are for easier problems.

These results show that the K-L transform reduces the difficult OCR problem to a relatively simple BDM problem so long as the BDMs are only asked to classify characters of the optimal class. We will show that the global problems is still difficult when each machine is required to classifly digits from other classes.

Feature	"0-1"	"6-8"	"3-8"
1	4.29	4.58	10.60
2	4.01	1.55	9.09
3	1.42	1.31	9.05
4	0.97	1.21	5.89
5	0.47	1.03	5.66
6	0.43	0.99	4.77
7	0.47	0.54	3.73
8	0.34	0.54	3.43
10	0.43	0.47	2.57
12	0.47	0.43	2.09
16	0.43	0.43	1.90
24	0.41	0.36	1.46
32	0.38	0.30	1.44
40	0.36	0.41	1.49
48	0.38	0.38	1.44

Table 1: Error in separating the test digits for several BDM machine using KNN as the feature set size was increased.

4.2 Input Network Training

The MLP1, MLP2, PNN, RBF1 and RBF2 classifiers were used to train 45 BDM Input Networks. The training of the each of the networkd has been done on 8, 12, 16, 20, 28, and 32 K-L features. The output signals of each network becomes the input signals for the

Output Network. The individual training set for each digit is run on all 45 BDMs in order to create the training feature set for the Output Network. The output signals are combined in a set order to produce the feature set for the input to the Output Network.

4.3 Output Network Training

The voting rule for the MLP1 Binary Decision Machines consists of a winner takes all approach. If the resulting signal is greater than 0.5 then the first class of the BDM receives the vote. If the resulting signal is less than 0.5 then the second class of the BDM receives the vote. If the resulting signal is equal to 0.5 the the BDM is rejected. The votes for each pattern are tallied and the class with the greatest number of votes is the winner. In the case of a tie, the pattern is rejected. With this rule, no class is able to receive more than 9 votes. The maximum possible of 9 votes is the result of each class being one of the base set for only 9 of the BDMs. This condition holds true for all the voting rules.

The voting rule for the MLP2, PNN, RBF1 and RBF2 classifiers consists of the greatest signal for the BDM winning the machine's vote for its class. The vote is tallied for each pattern and the class with the greatest number of votes wins. Any pattern which has a tie vote is rejected.

The MLP1, MLP2, PNN, RBF1 and RBF2 Output Networks were trained using the Input Network output signals from the training set as inputs.

4.4 Full Input - Voting Output

The Table 2. shows the results of each of the classifiers over 8, 12, 16, 20, 24, 28 and 32 features. It shows the percent error and the percent rejected due to ties. The table shows that the PNN BDMs have both the lowest percent error and percent reject when the classifying is done with the voting rules. The table also shows a slight decrease in the percent error for the PNN BDMs as the number of features increase with the lowest error at 32 features.

Classifier	Number of features						
	8	12	16	20	24	28	32
MLP1-ERR	3.9	3.2	3.0	2.9	3.0	2.8	3.2
MLP1-REJ	1.6	1.1	3.0	0.9	1.0	0.9	1.0
MLP2-ERR	3.9	3.2	3.2	2.6	2.9	2.9	2.9
MLP2-REJ	1.6	1.0	0.9	0.9	0.9	0.9	0.9
PNN-ERR	3.7	3.0	2.8	2.7	2.7	2.6	2.6
PNN-REJ	0.4	0.3	0.2	0.1	0.1	0.1	0.1
RBF1-ERR	6.8	6.3	6.1	5.9	5.8	5.8	6.4
RBF1-REJ	1.5	1.5	1.5	1.6	1.6	1.6	1.5
RBF2-ERR	6.4	5.2	5.0	5.1	4.7	4.6	3.4
RBF2-REJ	1.4	1.5	1.5	1.6	1.7	1.6	1.6

Table 2:
Voting Rule Error and Reject Percentages for Classifiers and Number
of Features. Reject Percentages based on the number of tied votes.

4.5 Full Input - Full Output

Table 3 shows the percent error for each class of digits for the PNN based BDMs. The combined output signals of the Input Network are presented to the PNN. MLP. RBF1. and RBF2 networks for 8. 12 16. 20. 24. 28 and 32 features used in the Input Network. The MLP network used 48 hidden nodes. As seen in Table 3. the 8 feature input network was unable to train using the RBF2. The RBF1 and RBF2 reported in Table 3 are from the cluster pattern which had the least percent error. Generally in Table 3. class "8' has the highest error rate. and class "6" has the lowest error rate. The bold face entrys in the table indicate the combination of features and networks in the output network which had the lowest error for that digit. For the"0" class this is an output PNN netwotk using 32 features for the input network. For the "8" class this is a PNN network using 16 features for the input network. The wide range of feature sizes and optimal output networks shows that the optimal decision critera for classification and the optimal input feature dimension vary with the type of character being classified for PNN BDMs.

Table 4 shows the percent error for each class of digits for the MLP2 based BDMs. As with the PNN BDMs. MLP2 BDMs generally achieve the best error rate for class "6" and the worst error rate for class "8". As in the previous table the bold face entrys in the table indicate the combination of features and networks in the output network which had the lowest error for that digit. For the"0" class this is an output RBF1 netwotk using 20 features for the input network. For the "8" class this is a RBF2 network using 20 features for the input network. Unlike the PNN case the 20 feature case contains most of the optimal output networks and RBF output networks have the highest digit by digit accuracy.

Table 5 shows the percent error for each class of digits for the MLP2 based BDMs. As with the PNN BDMs. MLP2 BDMs generally achieve the best error rate for class "6" and the worst error rate for class "8". When using the MLP2 Input Networks and a PNN Output Network the error rate for class "8" appears to at least double. As in the previous table the bold face entrys in the table indicate the combination of features and networks in the output network which had the lowest error for that digit. For the"0" class this is an output MLP netwotk using 20 features for the input network. For the "8" class this is a MLP network using 28 features for the input network. Like the MLP1 case the 20 feature case contains most of the optimal output networks but MLP output networks have the highest digit by digit accuracy.

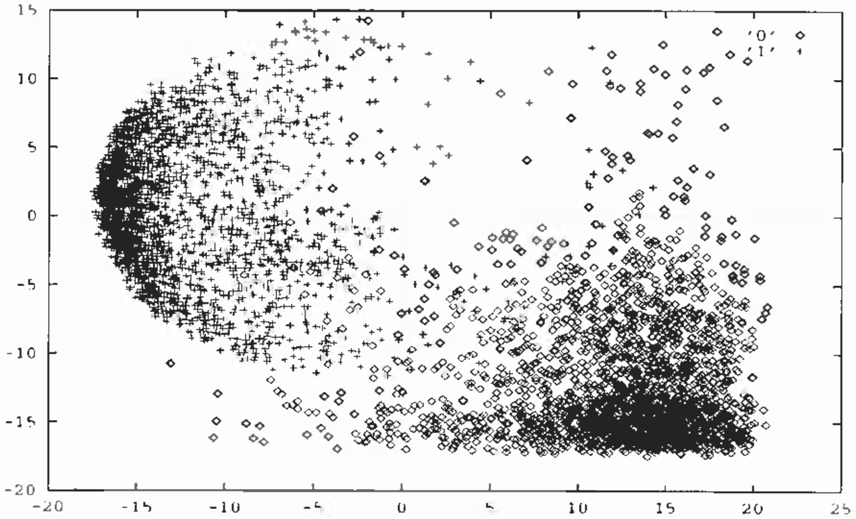

Figure 2: The separation of "0" and "1" testing data using the first two K-L features from "0" and "1" training data. 2314 examples of each digit are shown.

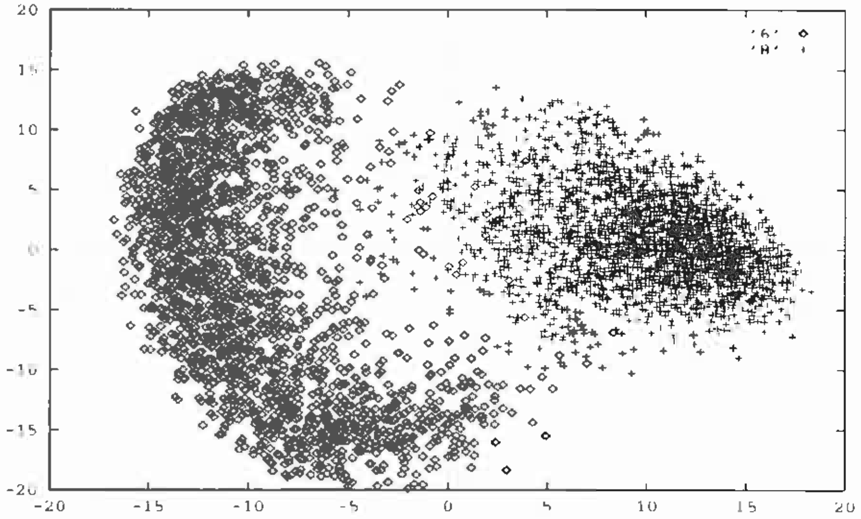

Figure 3: The separation of "6" and "8" testing data using the first two K-L features from "6" and "8" training data. 2314 examples of each digit are shown.

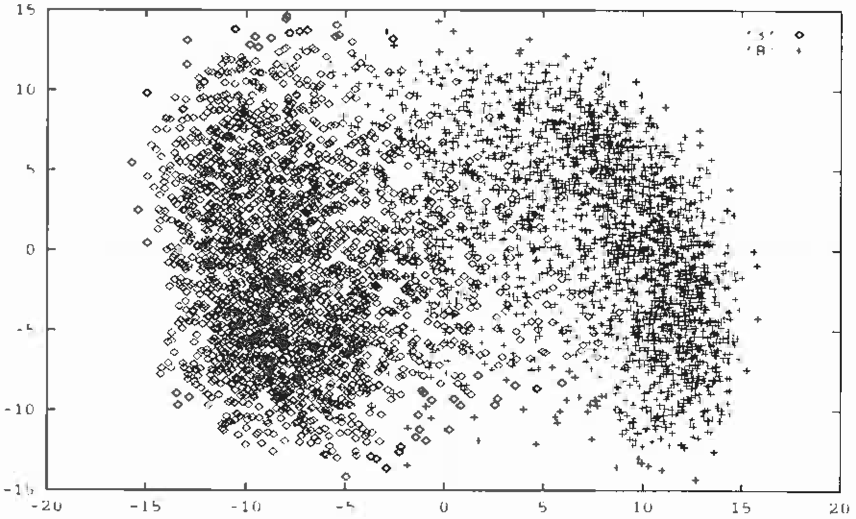

Figure 4: The separation of "3" and "8" testing data using the first two K-L features from "3" and "8" training data. 2314 examples of each digit are shown.

DIGIT	PNN BINARY DECISION MACHINES OUTPUT NETWORK									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
	8 FEATURES IN INPUT NETWORK									
MLP	7.1	4.1	7.4	14.5	23.3	5.5	8.0	7.7	8.0	13.0
PNN	1.5	1.4	3.6	4.6	2.8	4.0	1.0	2.8	7.1	4.6
RBF1	0.9	1.7	7.5	4.2	4.4	4.4	2.1	2.2	9.4	4.3
RBF2										
	12 FEATURES IN INPUT NETWORK									
MLP	4.3	1.3	4.1	**2.2**	2.8	3.2	2.0	2.1	7.7	4.1
PNN	1.6	1.4	3.0	4.8	2.8	6.9	1.2	2.5	**6.91**	3.9
RBF1	1.5	2.0	3.8	3.0	3.4	3.5	0.7	2.2	10.2	**3.4**
RBF2	1.2	1.8	4.1	2.5	3.8	4.4	0.9	2.2	11.7	3.5
	16 FEATURES IN INPUT NETWORK									
MLP	2.0	**1.0**	4.0	2.2	4.2	2.9	2.5	2.8	9.4	4.4
PNN	1.5	1.5	3.4	4.3	3.3	3.7	1.1	4.8	7.0	39.3
RBF1	1.5	1.6	4.2	3.1	3.3	3.8	0.7	2.6	8.5	3.8
RBF2	1.8	2.1	4.0	2.7	2.8	4.4	**0.5**	2.2	6.9	4.1
	20 FEATURES IN INPUT NETWORK									
MLP	2.5	1.3	5.1	5.0	4.0	2.9	2.6	**1.0**	9.5	11.0
PNN	1.4	1.3	4.0	4.2	3.2	3.8	1.1	2.4	7.4	3.8
RBF1	1.4	3.1	5.0	2.5	3.1	3.9	0.5	1.6	8.9	4.3
RBF2	1.5	1.8	4.2	2.8	2.9	3.9	0.5	2.2	10.4	4.3
	24 FEATURES IN INPUT NETWORK									
MLP	2.8	2.2	7.8	3.2	2.8	3.6	0.7	1.6	9.7	5.4
PNN	1.2	1.4	4.8	4.2	3.3	3.9	1.2	2.4	8.2	3.6
RBF1	1.3	2.5	4.0	2.9	2.3	4.0	0.6	1.5	7.7	5.2
RBF2	1.3	1.8	3.6	2.8	3.3	3.3	0.5	2.0	11.7	4.1
	28 FEATURES IN INPUT NETWORK									
MLP	1.9	1.2	5.2	6.0	2.7	**2.2**	1.8	2.2	8.7	7.1
PNN	1.1	1.6	6.4	4.9	3.5	4.3	1.4	2.3	9.0	3.9
RBF1	1.7	1.6	**2.4**	3.5	**1.8**	3.5	0.5	1.7	8.3	6.3
RBF2	1.3	2.1	2.9	2.8	2.9	3.5	0.7	2.1	12.8	4.4
	32 FEATURES IN INPUT NETWORK									
MLP	2.8	1.2	4.3	7.3	2.5	2.2	1.0	1.8	11.5	7.5
PNN	**0.1**	1.6	7.5	5.2	3.8	5.4	1.7	2.5	10.1	4.1
RBF1	1.5	1.6	3.1	4.4	3.3	2.8	0.6	2.0	8.4	4.1
RBF2	1.3	2.2	3.0	2.5	2.0	3.1	0.6	1.5	12.5	5.0

Table 3:
The error rates for each digit using the PNN BDMs.

DIGIT	MLP1 BINARY DECISION MACHINES OUTPUT NETWORKS									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
	8 FEATURES IN INPUT NETWORK									
MLP	4.0	2.8	6.5	4.9	2.7	5.1	1.9	2.7	9.3	5.1
PNN	8.6	3.6	26.1	9.5	24.1	19.5	1.2	7.6	34.1	7.1
RBF1	2.1	2.6	6.8	5.4	3.5	5.2	2.2	3.0	9.2	5.1
RBF2	2.1	2.8	6.6	5.4	3.4	5.0	2.0	3.0	9.3	5.1
	12 FEATURES IN INPUT NETWORK									
MLP	1.8	2.2	4.9	3.8	2.4	4.4	1.5	2.5	8.1	5.1
PNN	10.3	5.0	20.2	6.9	16.2	12.8	0.9	6.9	29.1	6.5
RBF1	3.9	4.2	4.8	3.9	3.2	4.5	1.6	2.4	8.1	5.0
RBF2	1.6	2.3	4.9	4.0	3.1	4.4	1.5	2.6	8.2	4.8
	16 FEATURES IN INPUT NETWORK									
MLP	1.6	2.1	4.7	**3.4**	2.2	4.2	1.7	2.2	7.3	5.1
PNN	12.0	5.0	20.0	6.0	9.5	12.3	1.3	4.7	26.7	7.7
RBF1	1.4	2.0	4.7	3.5	2.4	4.3	1.6	2.5	7.3	4.7
RBF2	1.5	2.2	4.9	3.5	2.4	4.5	1.4	2.3	7.3	5.1
	20 FEATURES IN INPUT NETWORK									
MLP	1.6	1.9	4.1	3.6	**1.9**	3.7	1.6	2.1	7.2	**4.2**
PNN	10.2	8.3	24.6	6.4	22.1	13.6	**0.4**	9.8	27.1	8.0
RBF1	**1.2**	2.0	4.3	3.5	2.4	**3.8**	1.4	**2.0**	7.3	4.5
RBF2	1.6	**1.9**	4.2	3.7	2.4	3.9	1.7	2.2	**7.0**	4.6
	24 FEATURES IN INPUT NETWORK									
MLP	1.5	2.2	4.0	3.6	2.3	4.6	1.3	2.8	7.3	4.7
PNN	10.5	7.2	25.6	7.0	20.1	16.5	0.4	8.3	30.7	8.4
RBF1	1.6	2.3	**4.0**	3.8	2.2	4.8	1.0	2.5	7.2	4.4
RBF2	1.5	2.2	4.1	3.8	2.0	4.5	1.5	2.6	7.4	4.4
	28 FEATURES IN INPUT NETWORK									
MLP	1.3	2.2	4.2	3.7	2.1	3.8	1.3	2.3	7.7	4.3
PNN	10.2	7.3	28.0	7.5	19.0	13.4	0.4	8.8	31.0	8.6
RBF1	1.2	2.3	4.1	3.7	2.3	3.9	1.3	2.3	7.4	4.3
RBF2	1.3	2.2	4.0	3.6	2.3	4.0	1.4	2.3	7.7	4.3
	32 FEATURES IN INPUT NETWORK									
MLP	1.6	2.3	4.7	4.2	2.5	4.7	1.4	2.3	8.4	4.9
PNN	8.2	7.3	27.7	7.8	19.2	15.5	0.5	8.9	27.9	7.5
RBF1	1.6	2.4	5.0	4.0	2.5	4.4	1.2	2.4	8.5	4.6
RBF2	1.6	2.5	5.4	3.9	2.5	4.6	1.1	2.3	8.7	4.6

Table 4:
The error rates for each digit using the MLP1 BDMs.

	MLP2 BINARY DECISION MACHINES OUTPUT NETWORK									
DIGIT	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
	8 FEATURES IN INPUT NETWORK									
MLP	5.0	3.0	7.0	4.7	2.9	4.3	1.7	2.7	9.2	4.7
PNN	2.9	2.1	10.4	6.7	2.6	6.0	1.5	2.5	18.4	6.6
RBF1	2.0	3.1	7.0	5.4	3.5	5.2	2.0	3.0	9.1	5.1
RBF2	2.0.	2.9	6.7	5.3	3.3	4.9	1.9	3.0	9.4	5.1
	12 FEATURES IN INPUT NETWORK									
MLP	1.8	2.5	4.5	3.9	2.8	4.1	1.6	2.2	7.9	4.9
PNN	3.0	2.3	7.3	4.9	2.6	4.3	1.0	2.8	17.5	5.9
RBF1	1.8	2.2	4.1	3.6	3.0	4.4	1.6	2.6	8.0	5.2
RBF2	1.8	2.2	4.7	3.6	2.8	4.3	1.6	2.6	8.2	5.1
	16 FEATURES IN INPUT NETWORK									
MLP	1.5	2.0	4.7	3.9	2.6	4.3	1.5	2.3	8.1	4.8
PNN	2.5	2.2	8.3	5.0	1.9	5.0	0.9	2.3	16.5	6.1
RBF1	1.6	2.3	4.4	3.8	2.5	3.9	1.6	2.5	8.2	5.0
RBF2	1.4	2.2	4.6	3.9	2.3	4.0	1.5	2.5	8.2	4.9
	20 FEATURES IN INPUT NETWORK									
MLP	**0.3**	1.9	3.5	**3.5**	2.2	**3.7**	1.1	1.9	8.1	**4.1**
PNN	1.2	2.3	7.9	4.7	2.4	4.7	**0.8**	**1.8**	16.5	6.6
RBF1	0.1	**1.8**	3.2	3.7	2.3	3.8	1.2	2.1	8.0	4.6
RBF2	0.1	1.9	**3.1**	3.6	2.5	3.7	1.2	2.1	8.2	4.5
	24 FEATURES IN INPUT NETWORK									
MLP	1.5	2.4	4.2	3.6	2.3	4.1	1.4	2.1	7.3	4.6
PNN	2.5	2.0	8.2	4.4	**1.7**	5.2	1.0	2.6	16.9	6.0
RBF1	1.6	2.2	4.1	3.8	2.1	4.1	1.2	2.3	**6.1**	4.4
RBF2	1.6	2.8	4.2	3.5	2.3	3.9	1.2	2.2	7.6	4.3
	28 FEATURES IN INPUT NETWORK									
MLP	1.4	2.5	4.4	3.5	2.2	3.8	1.3	2.2	7.1	4.6
PNN	2.8	2.0	8.1	4.6	1.8	4.7	1.1	2.3	16.2	7.0
RBF1	1.3	2.0	4.3	3.5	2.3	3.8	1.3	2.2	7.8	4.4
RBF2	1.3	2.6	3.9	3.5	2.3	3.8	1.2	2.2	7.4	4.5
	32 FEATURES IN INPUT NETWORK									
MLP	1.2	2.3	4.0	3.9	2.4	4.1	0.9	2.0	7.7	4.3
PNN	2.5	2.0	7.9	5.1	2.0	4.6	0.9	2.2	15.6	6.0
RBF1	1.2	2.2	4.0	3.8	2.2	4.3	1.0	2.2	7.6	4.4
RBF2	1.5	2.2	4.2	3.7	2.2	4.3	1.0	2.2	7.7	4.4

Table 5:
The error rates for each digit using the MLP2 BDMs.

DIGIT	RBF1 BINARY DECISION MACHINES OUTPUT NETWORKS									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
	8 FEATURES IN INPUT NETWORK									
MLP		2.3	4.3	4.5	3.0	4.5	2.1	2.8	8.0	5.4
PNN		3.6	13.5	6.2	9.7	9.6	2.1	5.6	26.9	10.4
RBF1		1.9	5.4	4.7	3.6	4.2	2.3	2.8	9.8	6.4
RBF2		2.7	4.8	4.4	3.5	4.5	1.9	2.5	9.2	6.0
	12 FEATURES IN INPUT NETWORK									
MLP	2.0	2.1	4.6	4.5	3.4	4.1	1.4	2.5	**7.9**	5.3
PNN	3.0	3.8	13.1	5.7	10.0	9.9	2.2	5.5	27.5	9.8
RBF1	2.5	2.4	4.8	4.8	3.4	4.4	2.8	2.8	10.1	5.8
RBF2	2.5	2.5	4.2	**3.9**	3.3	4.2	2.4	**2.4**	8.9	5.9
	16 FEATURES IN INPUT NETWORK									
MLP	2.1	**1.9**	4.6	4.4	2.7	3.7	1.8	2.9	8.9	5.0
PNN	2.8	3.5	12.9	5.9	8.8	9.8	2.2	4.8	27.8	8.9
RBF1										
RBF2										
	20 FEATURES IN INPUT NETWORK									
MLP	**2.0**	2.2	**4.2**	4.4	2.6	3.8	**1.4**	3.2	8.4	5.0
PNN	2.7	3.4	13.4	5.8	9.5	8.5	2.4	5.4	27.6	8.9
RBF1	2.7	2.5	5.3	4.6	3.4	4.1	2.1	2.6	9.7	5.6
RBF2	2.3	2.4	5.2	4.5	3.0	4.4	2.1	2.9	8.8	5.0
	24 FEATURES IN INPUT NETWORK									
MLP	2.1	2.2	4.4	5.0	3.0	**3.4**	1.5	3.0	8.7	5.7
PNN	2.8	3.3	13.3	5.6	7.9	9.1	2.2	5.4	28.3	8.2
RBF1	2.6	2.2	5.1	4.9	2.9	4.5	1.9	3.2	10.1	5.9
RBF2	2.1	2.2	4.7	4.7	2.5	4.3	2.1	2.9	9.2	5.3
	28 FEATURES IN INPUT NETWORK									
MLP	2.0	2.0	4.8	4.9	**2.3**	4.1	1.7	3.3	8.4	5.2
PNN	2.8	3.4	13.3	6.0	8.0	9.6	2.2	5.1	28.4	8.1
RBF1										
RBF2										
	32 FEATURES IN INPUT NETWORK									
MLP	2.2	1.9	4.4	4.8	2.5	4.0	2.2	3.7	8.7	**4.4**
PNN	3.4	3.2	13.9	6.7	9.5	8.2	3.1	5.1	33.6	8.6
RBF1										
RBF2										

Table 6:
The error rates for each digit using the RBF1 BDMs
with 8, 12, 16, 20, 24, 28, and 32 features in the Input Network.

DIGIT	RBF2 BINARY DECISION MACHINES OUTPUT NETWORKS									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
	8 FEATURES IN INPUT NEWTWOK									
MLP	2.1	3.1	5.1	3.9	4.8	4.4	2.0	3.9	9.9	5.6
PNN	3.0	2.7	12.7	6.3	9.5	8.2	2.0	5.0	26.4	9.6
RBF1	2.5	2.3	5.9	5.2	3.5	4.3	2.5	3.4	11.1	6.2
RBF2	2.3	2.6	5.2	4.4	3.4	4.2	2.5	4.0	11.3	5.3
	12 FEATURES IN INPUT NETWORK									
MLP	1.9	2.4	5.2	4.6	2.9	4.1	2.0	2.8	15.3	5.2
PNN	3.1	2.3	**2.0**	5.4	6.6	8.8	1.8	3.7	20.8	8.7
RBF1										
RBF2	2.5	2.7	4.5	4.6	3.3	4.4	2.7	2.6	7.9	5.5
	16 FEATURES IN INPUT NETWORK									
MLP	**1.8**	2.1	5.0	4.3	2.6	**4.3**	2.0	3.2	**7.8**	5.1
PNN	2.7	2.2	10.9	5.3	6.1	7.8	1.9	3.1	23.0	7.2
RBF1										
RBF2										
	20 FEATURES IN INPUT NETWORK									
MLP	1.9	2.5	5.0	4.0	3.0	4.7	1.9	3.3	8.7	5.3
PNN	2.8	1.9	12.3	4.9	6.1	7.8	1.9	3.5	22.0	7.7
RBF1										
RBF2										
	24 FEATURES IN INPUT NETWORK									
MLP	1.9	2.8	4.6	4.2	**2.2**	4.1	2.2	3.0	8.1	5.0
PNN	2.6	2.0	12.4	4.5	6.8	6.6	**1.8**	4.3	23.8	6.8
RBF1										
RBF2										
	28 FEATURES IN INPUT NETWORK									
MLP	2.0	2.3	5.7	4.4	2.3	4.4	1.9	**2.7**	8.3	5.2
PNN	2.6	**2.0**	12.4	4.9	6.5	7.9	1.9	3.7	22.8	6.1
RBF1										
RBF2										
	32 FEATURES IN INPUT NETWORK									
MLP	1.7	2.3	5.6	**3.8**	2.7	**3.6**	3.2	4.1	12.4	**4.6**
PNN	2.5	2.0	11.6	4.8	5.4	7.6	1.9	3.7	16.6	6.2
RBF1										
RBF2										

Table 7:
The error rates for each digit using the RBF2 BDMs
with 8, 12, 16, 20, 24, 28, and 32 features in the Input Network.

Table 6 shows the percent error for each class of digits for the RBF1 based BDMs. As with the PNN BDMs, MLP2 BDMs generally achieve the best error rate for class "6" and the worst error rate for class "8". As in the previous table the bold face entries in the table indicate the combination of features and input networks which, in the output network, had the lowest error for that digit. For the"0" class this is an output MLP netwotk using 20 features for the input network. For the "8" class this is a MLP network using 12 features for the input network. Unlike the MLP cases the number of features changes from digit to digit for the optimal output networks. MLP output networks have the highest digit by digit accuracy.

Table 7 shows the percent error for each class of digits for the RBF2 based BDMs. As with the other BDMs, RBF2 BDMs generally achieve the best error rate for class "6" and the worst error rate for class "8". As in the previous table the bold face entrys in the table indicate the combination of features and networks in the output network which had the lowest error for that digit. For the"0" class this is an output MLP netwotk using 16 features for the input network. For the "8" class this is a MLP network using 16 features for the input network. Unike the MLP1 case several different feature set sizes contains most of the optimal output networks. MLP output networks have the highest digit by digit accuracy.

PNN Binary Decision Machines							
	Number of features						
Classifier	8	12	16	20	24	28	32
VOTE	3.7	3.4	2.8	2.7	2.7	2.6	**2.6**
MLP	8.5	3.2	3.6	4.5	4.9	3.9	4.2
PNN	3.4	3.2	3.2	3.3	3.4	3.8	4.3
RBF1							
3 clusters	4.9	3.7	3.5	3.5	3.5	3.2	3.3
4 clusters	5.3	3.6	3.4	3.3	3.4	3.4	3.3
5 clusters	4.5	3.6	3.4	3.3	3.5	3.3	3.2
6 clusters	4.1	3.4	3.3	3.3	3.3	**3.1**	3.2
RBF2							
3 clusters		3.8	3.8	4.3	3.8	3.9	3.9
4 clusters		3.7	3.8	3.7	3.6	3.4	3.5
5 clusters		3.6	3.4	3.5	3.7	3.5	3.6
6 clusters		3.6	3.3	3.4	3.4	3.5	3.4

Table 8:
Percent Error for PNN BDMs for the Voting Rule, MLP, PNN, RBF1, and RBF2. Output Networks using different numbers of features from 8 t0 32.

Table 8 contains the global percent error rates for the PNN BDMs. The table shows that using the voting rule for PNN that the best error rate is 2.57% for a 32 feature Input Network. This should be comparable with the best PNN accuracy achived in [1] of 2.5% with 40 features. For the neural networks, the best error rate of 3.09% was achieved by the 28 feature Input Network RBF1 using 90 input nodes, 60 hidden nodes, 10 output nodes. Both these results are shown in bold face in the table. The neural network output network result is important because, as discussed in the next section, both VOTE and PNN output networks provide poor reject accuracy results.

Table 9 contains the global percent error rates for the MLP1 BDMs. This the modified MLP network of 45 BDMs using only 1 target value. The combined output signals of the Input Network are presented to the PNN, MLP, RBF1, and RBF2 networks over the 8 to 32 range of features used in the Input Network. The combined feature set has 45 nodes. The Output Network MLP has 45 input nodes, 32 hidden nodes, and 10 output nodes. The lowest error rate for a neural network of 3.29% was for the MLP with a 20 feature Input Network and a six cluster per class RBF1 network. Again both these results are shown in bold face in the table 9.

Table 10 contains the percent error rates for the MLP2 BDMs. This the unmodified MLP network of 45 BDMs using 2 target values. The combined output signals of the Input Network are presented to the PNN, MLP, RBF1, and RBF2 networks over the 8 to 32 range of features used in the Input Network. As the PNN base BDMs, the MLP Output Network has 90 input nodes and 48 hidden nodes. Both RBFs use the 90 input nodes and various hidden nodes from 30 to 60. In the MLP2 BDM network the best error rate of 2.59% for the voting rule using 20 feature Input Network. The RBF1 with 90 input nodes, 30 hidden nodes, and 10 output nodes achieved error rates of 3.09%. Again both these results are shown in bold face in the table. All of these error rates are substantial improvements over the best MLP error rate in [1] of 4.3% with 52 features.

Table 11 contains the percent error rates for the RBF1 BDMs. The combined output signals of the Input Network are presented to the PNN, MLP, RBF1, and RBF2 networks over the 8 to 32 range of features used in the Input Network. As the PNN base BDMs, the MLP Output Network has 90 input nodes and 48 hidden nodes. Both RBFs use the 90 input nodes and various hidden nodes from 30 to 60. MLP achieved an error rate of 3.69% using 12 input features. This error rate is a small improvements over the best RBF1 error rate in [1] of 4.2% with 48 features.

Table 12 contains the percent error rates for the RBF2 BDMs. The combined output signals of the Input Network are presented to the PNN, MLP, RBF1, and RBF2 networks over the 8 to 32 range of features used in the Input Network. As the PNN base BDMs, the MLP Output Network has 90 input nodes and 48 hidden nodes. Both RBFs use the 90 input nodes and various hidden nodes from 30 to 60. The 90 input nodes, 48 hidden nodes, 10 output nodes of the MLP achieved the best error rate of 3.02%. This error rate has improvements over the best RBF2 error rate in [1] of 3.9% with 44 features.

4.6 Error-Rejection Rates

In addition to forced decision accuracy, The error reject characteristics of the various combination of Input Network and Output networks were examined. The two most successful of these were PNN Input Networks and MLP input networks. The RBF Input Networks produced only marginal improvements in forced decision accuracy and has less effective reject accuracy performance.

Figure 5 shows the percent error versus the percent reject for the 24 feature PNN BDM's. The percent error for the PNN BDM voting rule is based on the output signal size. A machine is rejected if its signal is less then a given threshold value. The pattern is rejected if all 45 BDMs are rejected or if there is a tie. Each of the graphs show that the voting rule starts with a lower error rate, but the RBF1 and RBF2 have a much better rejection rate over the voting rule.

MPL1 Binary Decision Machines							
	Number of features						
Classifier	8	12	16	20	24	28	32
VOTE	3.9	3.2	3.0	2.9	2.9	**2.8**	3.2
MLP	4.3	3.7	3.5	3.3	3.4	3.4	3.7
PNN	14.1	11.5	9.9	13.1	13.5	13.4	13.1
RBF1							
3 clusters	4.6	3.8	3.5	3.3	3.4	3.3	3.7
4 clusters	4.5	3.7	3.5	3.3	3.4	3.3	3.7
5 clusters	4.5	3.7	3.5	3.3	3.4	3.3	3.7
6 clusters			3.5	**3.3**	3.5	3.3	3.7
RBF2							
3 clusters	4.5	3.8	3.5	3.3	3.4	3.3	3.7
4 clusters	4.5	3.8	3.5	3.3	3.4	3.3	3.7
5 clusters	4.5	3.8	3.5	3.3	3.4	3.3	4.7
6 clusters			3.5	3.3	3.4	3.3	3.7

Table 9:
Percent Error for MLP1 BDMs for the Voting Rule. MLP. PNN. RBF1. and RBF2. Output Networks using different numbers of features from 8 to 32.

MPL2 Binary Decision Machines							
	Number of features						
Classifier	8	12	16	20	24	28	32
VOTE	3.9	3.2	3.2	2.6	2.9	2.9	2.9
MLP	4.3	3.6	3.5	3.1	3.3	3.3	3.3
PNN	6.1	5.2	5.1	4.9	5.1	5.0	11.9
RBF1							
3 clusters	4.5	3.8	3.6	**3.1**	3.3	4.3	3.3
4 clusters	4.5	3.7	3.6		3.3	3.3	3.3
5 clusters			3.6		3.2	3.3	3.3
6 clusters			3.6		3.3		
RBF2							
3 clusters	4.5	3.6	3.6	3.1	3.3	3.3	3.4
4 clusters	4.5	3.7	3.5		3.3	3.3	3.4
5 clusters			3.5		3.3	3.3	3.3
6 clusters			3.6		3.3		

Table 10:
Percent Error for MLP2 BDMs for the Voting Rule. MLP. PNN. RBF1. and zRBF2. Output Networks using different numbers of features from 8 to 32.

RBF Type 1 Binary Decision Machines							
	Number of features						
Classifier	8	12	16	20	24	28	32
VOTE	6.8	6.3	6.1	5.9	5.8	5.8	6.4
MLP	13.4	**3.7**	3.8	3.7	3.9	3.9	3.9
PNN	28.7	9.0	8.7	8.8	8.6	8.7	9.6
RBF1 3 clusters	14.6	6.1	5.0	4.9	6.0	4.9	5.1
4 clusters	14.4	4.7	4.7	4.6	4.6	4.5	4.9
5 clusters	14.2	4.5	4.5	4.5	4.5	4.3	
6 clusters	14.0	4.4		4.3	4.3		
RBF2 3 clusters	14.2	4.3	4.4	4.4	4.4	4.5	4.6
4 clusters	14.0	4.1	4.2	4.2	4.2	4.1	4.2
5 clusters	13.8	4.0	4.1	4.2	4.0	4.0	
6 clusters	13.3	4.0		4.0	4.0		

Table 11:
Percent Error for RBF1 BDMs for Voting Rule, MLP, RBF1, and RBF2. Networks used different numbers of features from 8 to 32. Results show percent error found when using different Output Networks.

RBF Type 2 Binary Decision Machines							
	Number of features						
Classifier	8	12	16	20	24	28	32
VOTE	6.4	5.2	5.0	5.1	4.7	4.6	3.4
MLP	4.5	4.6	3.8	4.0	**3.0**	3.9	4.4
PNN	9.5	7.3	7.0	7.0	7.2	7.1	6.2
RBF1 3 clusters	5.2	4.7	4.6	4.5	4.7	4.6	7.2
4 clusters	4.9						
5 clusters	4.7						
6 clusters	4.7						
RBF2 3 clusters	4.7	4.5	4.2	4.3	4.2	4.3	7.8
4 clusters	4.7	4.6	4.2	4.2	4.1	4.2	
5 clusters	4.5	4.4	4.1				
6 clusters	4.6	4.1		4.1			

Table 12:
Percent Error for RBF2 BDMs for Voting Rule MLP, PNN, RBF1 and RBF2. Networks used different numbers of features from 8 to 32. Results show percent error found when using different Output Networks.

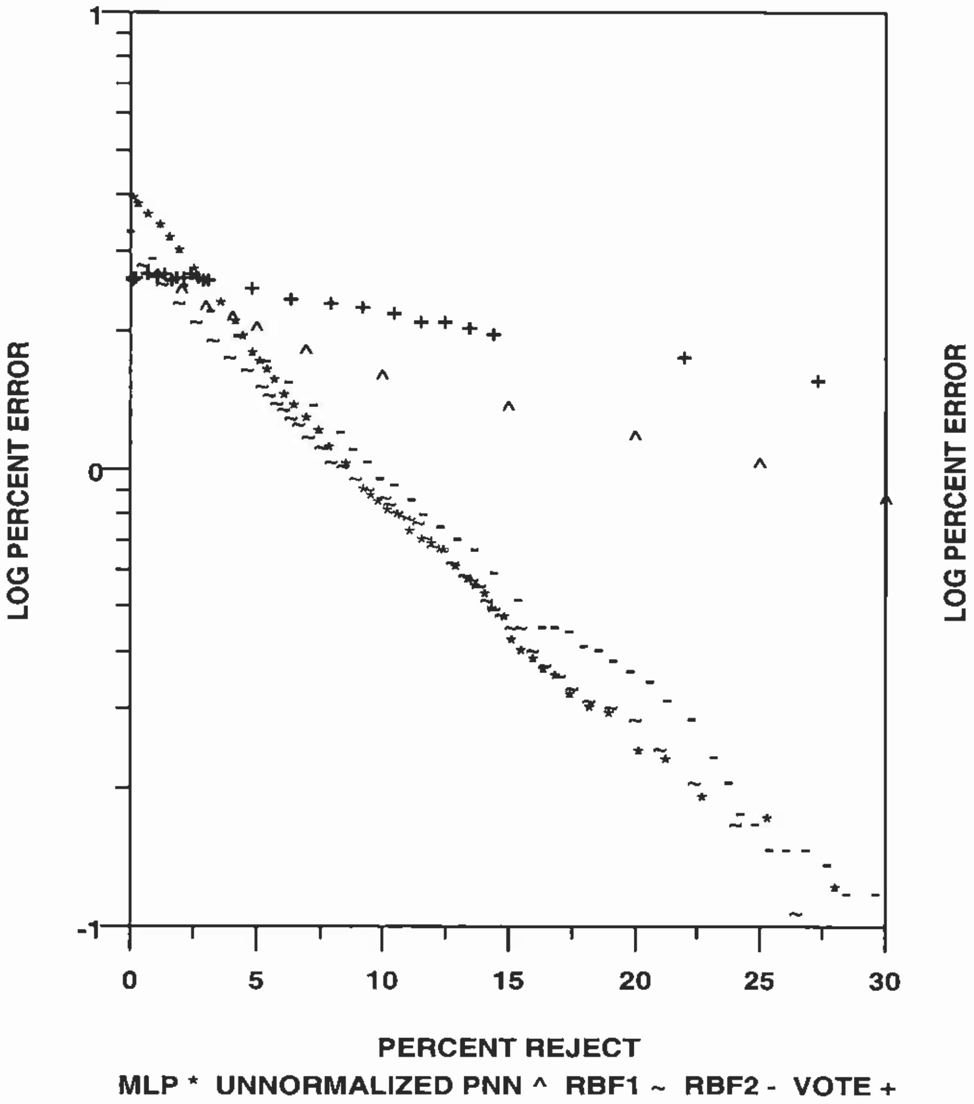

Figure 5: 24 Feature PNN BDM Error vs Reject for different output networks.

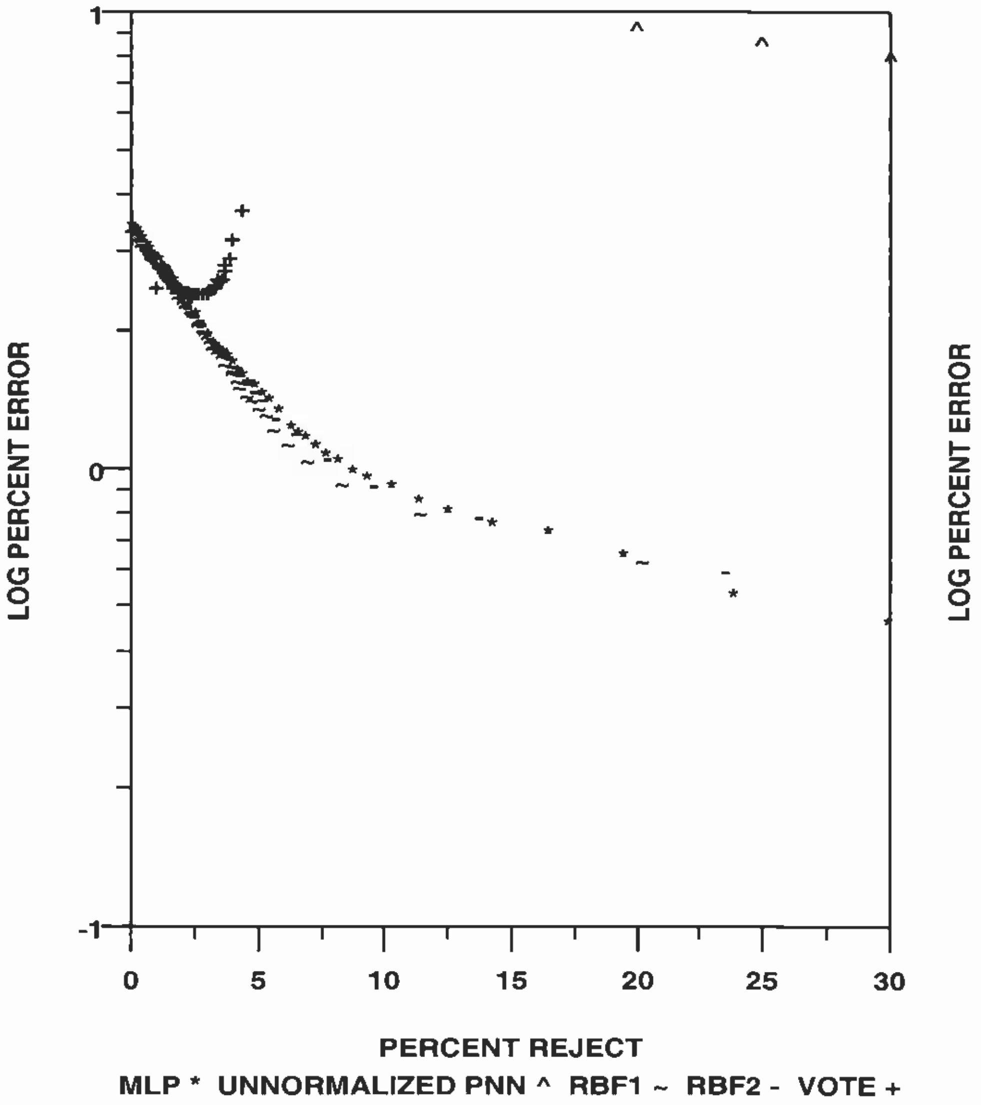

Figure 6: 24 Feature MLP1 BDMs Error vs Reject for different output networks.

Figure 6 shows the log percent error versus the percent reject of the 24 feature MLP1 BDMs. The percent error for the MLP1 BDMs voting rule is based on the output signal's magnitude. A machine is rejected if its signal is less then a given threshold value. The pattern is rejected if all 45 BDMs have been rejected based on the threshold or if the resulting vote is a tie. The curve for the MLP1 voting network is the result of both the number of machines rejected per pattern and the number of tie votes which result.

Although the PNN and voting rule based systems give good forced decision accuracy the provide poor reject accuracy performance. The best combination of input and output networks is one which uses PNN BDMs and a RBF1 output. This provides 3.09% forced decision accuracy and 1% accuracy at about 8% rejection. For rejection rates between 1% and 7% an all MLP based system will provide better reject accuracy since at 7% rejection rate 1% accuracy is achieved.

5 Conclusions

The structure consists of three layers of processing, the K-L feature processing, the input network layer and the output network layer. As we demonstrated in table 1 the K-L layer can provide accuracy approching [1] on the binary digit problem. We also demonstated in table 2 that a simple voting mechanism can get good results using the outputs of the second layer. Unfortunately, neither of these mechanism provides satisfactory estimates of the confidence of its result so that the output layer is required to privide good reject accuracy performance. This high accuracy on forced decision coupled with poor reject performance was also observed in [15].

It was also possible to improve the recognition performance, over that obtained with the same test and training sets in [1], of both the MLP and RBF methods by the use of BDMs. The performance of the local methods, PNN, is not improved by this process. The improvement in MLP performance is greater than the improvement in RBF performance. This clearly indicated that the methods improved in a way which is proportional to the amount that they are converted form global to local mathods. RBF, as used here, is preclustered and is therefore partly local and partly global. RBF is improved but not as much as the MLP networks are. MLPs are usually global but by converting them to local methods, even when it results in a smaller traing set as it does in this case, has made them perform as well as local methods. This is an issue of intrinsic dimensionality of the type discussed in [14]. Local dimensionality has long been recognized as a critial factor in neighbor based methods and we now conclude that it is equally important for neural networks methods.

This intrinsic variability of rank of the feature set is further seen in the distribution of character by character errors seen with feature set size and output network type and in the large difference in errors for the BDMs associated with different digits. In tables 3-7 typical errors for the "6" class are 0.5% and typical errors for the "8" class are 7.0% so that in this sense the classification of "8" is 14 times harder than the classification of "6". This variability of classification accuracy by character type has also been seen using an image based method [34, 9] where the number of memories needed to recognize digits is highly class dependent. This is a clear indication that the features used to select a "6" are much more efficient than the features used to select an "8".

Another indication of the local rank effects in the problem structure for OCR is provided by the clear division of Tables 3-7 into two groups distinguished by the amount of change in feature set size for maximum accuracy for each digit. Tables 3,6, and 7 for PNN, RBF1, and RBF2 networks show large changes in optimum feature set size and output network type for different digits. Tables 4 and 5 for MLP1 and MLP2 networks show a global optimum feature set size of 20 for most digit types. This is less than half the optimum feature set size for the global solution to this same problem given in [1]. The change in optimum feature set size and the sharp decrease in feature set size for BDM networks demonstrates that the global solution is rank deficient and that clustering of the global problem into local problems reduces the rank of the optimum feature set and therefore makes global optimization, used for the MLP networks, more effective.

The need for global rank reduction is further supported by Boltzmann pruning studied of large MLP networks. In [35, 36] it was shown that in these networks up to 80% of the weights can be removed without affecting the network performance and that the remaining weights typically have 9-11 bits of information content. As the weights are removed form the network the removal of low variance weights connected to K-L features associated with smaller eigenvalues is strongly favored. Weight pruning provides strong evidence of rank deficiency in the global problem and the relatively small number of digits present in even the pruned weights demonstrates that calculations using these weight values may experience significant problems.

The networks that are trained using optimization methods [20, 24] are, during training, dynamic systems subject to the same convergence and stability problems which are present in other dynamiccal systems [37, 38, 39]. All of these methods of training nonlinear networks are directly or indirectly dependent on the stability of the Jacobiam matrix of the local linearization of the system. The equivalent linear problem is dependent on the rank of the covariance matrix as is the computation of the K-L transform. If the number of bits used in the variables which form this matrix is lower than is practical for the condition number of the matrix used in the training process the process will be both dynamically and numerically unstable. This does not mean that all solutions found are useless but only that some of these solutions are selected at random based on rounding error not on input data. In problems which are seriously rank deficient the training process is a Boltzmann machine where nemerical rounding error and input signal noise serve as the random number generator. Clustering, using BDMs or other methods, should be used to provide an effective method for rank reduction and improve system stability.

In [1] and [15] many different OCR systems were presented which achieved 5% error rates and several were presented which has 2%-3% error rates. By reducing the problem to a series of BDM we have shown that several different neural network methods can achieve 3% error rates on the relatively small training set used in [1]. These systems also can exhibit error reject behavior comparable to the best presented in [15]. Analysis of the digit by digit performance with respect to feature set size and network type shows that much of this improvement is associated with local rank reduction in the feature set.

These changes the way the problem of character recignition is stated. We now know that the binary decision process for two characters using learning data that consists of other characters from the binary set is relatively easy. The "3-8" data using K-Ls and KNN can get 1.in general get errors of 7.6%. So "3-8" is three times harder than "0-1" and "8"s are 5 times harder than "1"s. The increasing difficulty is caused not by the primary seraration of digits but by the ability of the BDMs to reject characters of other classes. This is particulary important when an OCR system is constructed since the most sucessful methods of segmentation depend of deliberate over segmentation and reconstruction [40]. The over segmentation process generates numerious partial and merged sections of digits which must be rejected for OCR to suceed.

Another way of considering the rejection problem is to consider the number of images that are near any 32 by 32 binary image in image space. If character images should be recognized after reversing m bits of an n bit image then each character has $n! - (n - m)!$ neighbors which must be recognized and $(n - m)!$ derived images that should be rejected. Both these numbers are very large compared to any projected OCR test or training set and indicate the redundancy of even simple character images. The more redundant the image set the more difficult the classification process is since many small variations in the image yield no useful classification data. Larger and larger training set are more rather than less redundant since they will contain more examples of common character types.

From these arguments we would expect neither larger feature sets nor larger training set to eleminate the remaining sources of OCR error. Larger feature set can only be effective if the increase the rank of the feature set. Larger training set can only be effective if they provide new protypes which are not redundant.

Acknowledgement

The authors would like to acknowledge Jack Rabinow for suggesting the BDM approch and Craig Watson, and Jerry Candela for helping with various aspects of the calculations.

References

[1] J. L. Blue, G. T. Candela, P. J. Grother, R. Chellappa, and C. L. Wilson. Evaluation of Pattern Classifiers for Fingerprint and OCR Applications. *Pattern Recognition*, 27(4):485–501, 1994.

[2] G.A. Carpenter and S. Grossberg. A massively parallel architecture for a self organizing neural pattern recognition machine. *Neural Networks*, 2:169–181, 1989.

[3] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115, 1987.

[4] G. A. Carpenter and S. Grossberg. Art 2: self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26:4919–4930, 1987.

[5] J. A. Hartigan. *Clustering Algorithms*, pages 84–108. New York: John Wiley & Sons, Inc., 1975.

[6] D. L. Alkon, K. T. Blackwell, G. S. Barbour, A. K. Rigler, and T. P. Vogl. Pattern-recognition by an artificial network derived from biological neuronal systems. *Biological Cybernetics*, 62:363–376, 1990.

[7] Richard Granger, Jose Ambros-Ingerson, and Gary Lynch. Derivations of encoding characteristics of layer II cerebral cortex. *Journal of Cognitive Neuroscience*, 1(1):67–87, 1989.

[8] Jose Ambros-Ingerson, Richard Granger, and Gary Lynch. Simulation of paleocortex performs hierarchical clustering. *Science*, 247:1344–1348, 1990.

[9] C. L. Wilson. FAUST: a vision based neural network multi-map pattern recognition architecture. In *Proceedings: Applications of Artificial Neural Networks III*, Orlando, SPIE, April 1992.

[10] 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.

[11] G. L. Martin. Centered-object integrated segmentation and recognition for visual character recognition. In J. Moody. S. Hanson. and R. Lippmann, editors. *Advances in Neural Information Processing Systems*. volume 4. pages 504–511. Morgan Kaufmann. Denver. December 1991.

[12] G. Martin and J. Pittman. Recognizing hand-printed letters and digits. In D. Touretzky. editor. *Advances in Neural Information Processing Systems*. volume 2. pages 405–414. Morgan Kaufmann. 1990.

[13] G. Martin and J. Pittman. Recognizing handprinted letters and digits using backpropagation. *Neural Computation*. 3:258–267. 1991.

[14] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. New York: Academic Press. second edition. 1990.

[15] R. A. Wilkinson. J. Geist. S. Janet. P. J. Grother. C. J. C. Burges. R. Creecy. B. Hammond. J. J. Hull. N. J. Larsen. T. P. Vogl. and C. L. Wilson. The First Optical Character Recognition Systems Conference. Technical Report NISTIR 4912. National Institute of Standards and Technology. August 1992.

[16] Michael D. Garris. James L. Blue. Gerald T. Candela. Darrin L. Dimmick. Jon Geist. Patrick J. Grother. Stanley A. Janet. and Charles L. Wilson. NIST Form-Based Handprint Recognition System. Technical Report NISTIR 5469. National Institute of Standards and Technology. July 1994.

[17] P. J. Grother. Karhunen Loève feature extraction for neural handwritten character recognition. In *Proceedings: Applications of Artificial Neural Networks III*. Orlando. SPIE. April 1992.

[18] T. P. Vogl. K. L. Blackwell. S. D. Hyman. G. S. Barbour. and D. L. Alkon. Classification of Japanese Kanji using principal component analysis as a preprocessor to an artificial neural network. In *International Joint Conference on Neural Networks*. volume 1. pages 233–238. IEEE and International Neural Network Society. 7 1991.

[19] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*. 78(9):1481–1497. 1990.

[20] 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.

[21] D. E. Rumelhart. G. E. Hinton. and R. J. Williams. Learning representations by back-propagating errors. *Nature*. 332:533–536. 1986.

[22] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Computer Journal*. 7:149–154. 1964.

[23] E. M. Johansson. F. U. Dowla. and D. M. Goodman. Backpropagation learning for multi-layer feed-forward neural networks using the conjugate gradient method. *IEEE Transactions on Neural Networks*. 1991.

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

[25] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and K. M. Hummels. On the training of radial basis function classifiers. *Neural Networks*, 5:595–603, 1992.

[26] D. Wettschereck and T. Dietterich. Improving the performance of radial basis function networks by learning center locations. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 1133–1140, San Mateo, 1991. Morgan Kaufmann.

[27] Donald F. Specht. Probabilistic neural networks. *Neural Networks*, 3(1):109–118, 1990.

[28] M. D. Garris and R. A. Wilkinson. Handwritten segmented characters database. Technical Report Special Database 3, **HWSC**, National Institute of Standards and Technology, February 1992.

[29] C. L. Wilson and M. D. Garris. Handprinted character database. Technical Report Special Database 1, **HWDB**, National Institute of Standards and Technology, April 1990.

[30] R. G. Casey and H. Takahashi. Experience in Segmenting and Recognizing the NIST Database. In *Proceedings of the International Workshop on Frontiers of Handwriting Recognition*, France, 1991.

[31] Patrick J. Grother. Cross Validation Comparison of NIST OCR Databases. In D. P. D'Amato, editor, , volume 1906, SPIE, San Jose, 1993.

[32] Patrick J. Grother and Gerald T. Candela. Comparison of Handprinted Digit Classifiers. Technical Report NISTIR 5209, National Institute of Standards and Technology, June 1993.

[33] C. L. Wilson and J. L. Blue. Neural network methods applied to character recognition. *Social Science Computer Review*, 10:173–195, 1992.

[34] C. L. Wilson. A New Self-Organizing Neural Network Architecture for Parallel Multi-Map Pattern Recognition - FAUST. *Progress in Neural Networks*, 4, 1993. to be published.

[35] O. M. Omidvar and C. L. Wilson. Information Content in Neural Net Optimization. *Journal of Connection Science*, 6:91–103, 1993.

[36] 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.

[37] Morris W. Hirsch and Stephen Smale. *Differential Equations, Dynamical Systems, and Linear Algebra*. Academic Press, New York, NY, 1974.

[38] John Guckenheimer and Philip Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer-Verlag, New York, NY, 1983.

[39] Morris W. Hirsch. Convergent activation dynamics in continuous time networks. *Neural Networks*, 2:331–349, 1989.

[40] J. Geist, R. A. Wilkinson, S. Janet, P. J. Grother, B. Hammond, N. W. Larsen, R. M. Klear, M. J. Matsko, C. J. C. Burges, R. Creecy, J. J. Hull, T. P. Vogl, and C. L. Wilson. The Second Census Optical Character Recognition Systems Conference. Technical Report NISTIR 5452, National Institute of Standards and Technology, May 1994.