Artificial Neural Networks for Feature Extraction and Multivariate Data Projection

Jianchang Mao and Anil K. Jain, *Fellow, IEEE*

Abstract— **Classical feature extraction and data projection methods have been well studied in the pattern recognition and exploratory data analysis literature. In this paper, we propose a number of networks and learning algorithms which provide new or alternative tools for feature extraction and data projection. These networks include a network (SAMANN) for Sammon's nonlinear projection, a linear discriminant analysis (LDA) network, a nonlinear discriminant analysis (NDA) network, and a network for nonlinear projection (NP-SOM) based on Kohonen's self-organizing map. A common attribute of these networks is that they all employ adaptive learning algorithms which makes them suitable in some environments where the distribution of patterns in feature space changes with respect to time. The availability of these networks also facilitates hardware implementation of well-known classical feature extraction and projection approaches. Moreover, the SAMANN network offers the generalization ability of projecting new data, which is not present in the original Sammon's projection algorithm; the NDA method and NP-SOM network provide new powerful approaches for visualizing high dimensional data. We evaluate five representative neural networks for feature extraction and data projection based on a visual judgement of the two-dimensional projection maps and three quantitative criteria on eight data sets with various properties. Our conclusions based on analysis and simulations can be used as a guideline for choosing a proper method for a specific application.**

I. Introduction

FEATURE extraction and multivariate data projection are important issues in pattern recognition and exploratory data analysis [8], [17]. Feature extraction can avoid the "curse of dimensionality," improve the generalization ability of classifiers, and reduce the computational requirements of pattern classification. Data projection enables us to visualize high dimensional data to better understand the underlying structure, explore the intrinsic dimensionality, and analyze the clustering tendency of multivariate data. As a result, feature extraction and multivariate data projection have received considerable attention for the past 20 years [4], [8], [17].

Recently, a large number of artificial neural networks and learning algorithms have been proposed for feature extraction and data projection [1], [2], [5], [12], [19], [20], [23]–[28], [31], [33], [34], [36]. These research efforts can be loosely classified into two groups. The first group contains new neural network designs or existing neural network models for feature extraction and data projection. Examples of this type of efforts include Kohonen's self-organizing maps [20]–[22], the nonlinear projection methods proposed by Kraaijveld *et al.* [23], and nonlinear discriminant analysis based on the functionality of hidden units in feedforward network classifiers [28], [37]. These networks exhibit some nice properties which are different from classical approaches. Therefore, they can be used as new tools or supplementary approaches to classical feature extraction and data projection algorithms. The second group of researchers have investigated the properties of neural networks and learning rules and have established several links between classical approaches and neural networks. As a result, it has been found that some of these neural networks actually perform many of the well-known feature extraction and data projection algorithms [9], [10], [30]. In an attempt to link the fields of pattern recognition and artificial neural networks, several classical feature extraction and data projection approaches have been implemented using neural network architectures [1], [2], [5], [12], [19], [24]–[26], [28], [31], [33], [34], [36]. Although such efforts do not necessarily provide new approaches to feature extraction and data projection (from the view point of functionality performed by the networks), such networks do possess some advantages over the traditional approaches. i) Most learning algorithms and neural networks are adaptive in nature, thus they are well-suited for many real environments where adaptive systems are required. ii) For real-time implementation, neural networks provide good, if not the best, architectures which are relatively easily implemented using current VLSI and optical technologies. iii) Neural network implementations can overcome the drawbacks inherent in the classical algorithms. For example, Jain and Mao [19] proposed a neural network design for Sammon's nonlinear projection algorithm which offers the generalization ability of projecting new data, a property not present in the original algorithm.

We have proposed several neural networks and learning algorithms for feature extraction and data projection. Some preliminary results of our work have appeared in several recent conferences [19], [23], [28]. This paper will present a more thorough analysis and more complete results. Although a variety of neural networks and learning algorithms have been proposed in the literature, as far as we are aware, no comparative study of the performance of these approaches for feature extraction and data projection has been reported yet. In this paper, we also provide a performance evaluation of a number of representative networks based on a visual inspection of two-dimensional projection maps and on three quantitative criteria over eight data sets with various properties.

Manuscript received May 22, 1993; revised Oct. 14, 1993. This work was supported in part by NSF Grant IRI 9002087.

J. Mao is with IBM Almaden Research Center, San Jose, CA 95120 USA.

A. K. Jain is with the Department of Computer Science at Michigan State University, East Lansing, MI 48824 USA.

IEEE Log Number 9214507.

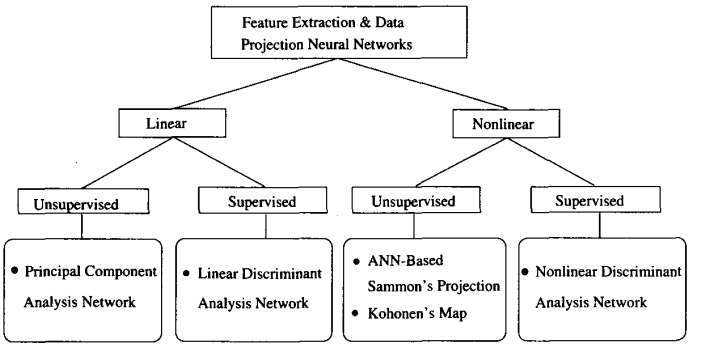


Fig. 1. A taxonomy of neural networks for feature extraction and data projection.

The rest of this paper is organized as follows. Section II presents our proposed neural networks for feature extraction and data projection. The network of Rubner *et al.* is also discussed because it will be used as a building block in our own network for linear discriminant analysis. Section III provides a comparative study of five representative neural networks for feature extraction and data projection. Conclusions are given in Section IV.

II. Feature Extraction and Data Projection

Feature extraction and data projection can be formulated as a mapping Ψ from a d-dimensional input space to an m-dimensional output space (map space)

$$\Psi : R^d \rightarrow R^m,\ m \leq d \tag{1}$$

such that some criterion, $\mathcal{C}$, is optimized. This formulation is similar to a function approximation problem. However, unlike the function approximation problem where the mapping function is estimated from training patterns which are input–output pairs (desired outputs are known), in feature extraction and data projection the desired outputs are often not available even for supervised approaches (e.g., linear discriminant analysis). Feature selection, which is used for choosing an optimal subset of features, can be viewed as a special case of feature extraction where Ψ is a linear $m \times d$ permutation matrix. For data visualization purpose, the value of m is usually set to two or three.

A large number of approaches for feature extraction and data projection are available in the pattern recognition literature [4], [17]. These approaches differ from each other in the characteristics of the mapping function Ψ, how Ψ is learned, and what optimization criterion $\mathcal{C}$ is used. The mapping function can be either linear or nonlinear, and can be learned through either supervised or unsupervised methods. The combination of these two factors results in the following four categories of feature extraction and data projection: i) unsupervised linear, ii) supervised linear, iii) unsupervised nonlinear, and iv) supervised nonlinear. Within each category, the methods differ in the criterion function $\mathcal{C}$ being used. This taxonomy is shown in Fig. 1. In general, linear methods are attractive because they require less computation than nonlinear methods. Analytical solution is often available for linear methods. On the other hand, nonlinear methods are more powerful than linear methods. Since the analytical solution is often not available, however, a numerical optimization method must be used to obtain the solution, which is usually computationally demanding. Furthermore, the optimization procedure often gets trapped into a local minimum. It is also generally true that supervised methods have better performance than unsupervised methods in situations where the category information is available [4].

Various neural networks including those proposed in this paper for feature extraction and data projection can also be grouped into the above four categories. The boxes at the bottom of Fig. 1 list a number of representative networks in each category which will be described and evaluated in this

Choice of a proper feature extraction and data projection method (both the traditional and neural network approaches) for a given problem is driven by i) the available information (supervised versus unsupervised), ii) *a priori* knowledge about the underlying distribution of the data (linear versus nonlinear, and $\mathcal{C}$), and iii) the task requirement (classification or visualization). Knowing the details and the characteristics of all the available approaches is crucial to making a good choice. In this paper, we will investigate the characteristics of a number of neural networks through analysis and evaluation experiments.

A. Principal Component Analysis (PCA) Networks

Principal component analysis, also called Karhunen–Loeve transform, is a well-known statistical method for feature extraction, data compression, and multivariate data projection and has been widely used in communication, signal and image processing, pattern recognition, and data analysis. It is a linear orthogonal transform from a d-dimensional input space to an m-dimensional space, $m \leq d$, such that the coordinates of the data in the new m-dimensional space are uncorrelated and maximal amount of variance of the original data is preserved by only a small number of coordinates.

Projection pursuit is a more general approach in this category which includes principal component analysis as a special case. Projection pursuit was first introduced by Friedman and Tukey [7] in 1974 as a technique for exploratory analysis of multivariate data sets. Later in 1985, Huber [14] provided a unified framework of projection pursuit which included

principal component analysis, multidimensional scaling, factor analysis, nonparametric regression (projection pursuit regression), density estimation (projection pursuit density estimation) and deconvolution. The standard two-layer (one hidden layer) feedforward network is closely related to projection pursuit regression in approximating functions although the backpropagation algorithm used for training the feedforward network is different from the algorithms used in projection pursuit regression. A number of researchers have explored using projection pursuit regression and density estimation in training or constructing feedforward networks [3], [15], [38].

Let $\boldsymbol{\xi}_k = (\xi_{k1}, \xi_{k2}, \cdots, \xi_{kd})^T$ denote the kth d-dimensional input vector, $k = 1, 2, \cdots, n$. Assume zero-mean vectors without a loss of generality. Let Σ be the covariance matrix of the data, with eigenvalues, $\lambda_1, \lambda_2, \cdots, \lambda_d$, in the decreasing order. The m principal components of the data can be obtained by a linear transform

$$\mathbf{o}_k = \Phi^T \boldsymbol{\xi}_k \tag{2}$$

where $\mathbf{o}_k = (o_{k1}, o_{k2}, \cdots, o_{km})^T$, and $\boldsymbol{\Phi}$ is a $d \times m$ matrix whose columns are m eigenvectors corresponding to the m largest eigenvalues of the covariance matrix Σ. It is easy to show that the covariance matrix of the transformed data is a diagonal matrix, $\mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_m\}$, which means that the components of the transformed data are uncorrelated. The variance of the original data retained in the new m-dimensional space is $\sum_{i=1}^{m} \lambda_i$, which is the largest retained value among all linear orthogonal transforms of the same output dimensionality. Principal component analysis is optimal in the mean-square error sense among all linear orthogonal transforms.

A large number of neural networks and learning algorithms have been proposed for principal component analysis [1], [2], [5], [12], [24], [26], [31], [33], [34], [36]. The PCA network proposed by Rubner *et al.* [33], [34] will be used in the comparative study and in our proposed discriminant analysis network and thus will be discussed in detail.

The PCA network architecture proposed by Rubner *et al.* [33], [34] is shown in Fig. 2. It consists of d input nodes and m output units. Assume $m = d$ without any loss of generality. Each input node i is connected to each output unit j with connection strength w_{ij}. We denote the weight vector associated with output unit j from the d inputs by $\mathbf{w}_j = (w_{1j}, w_{2j}, \cdots, w_{dj})^T$. All the output units are hierarchically organized in such a way that the output unit i is connected to the output unit j with connection strength u_{ij} if and only if $i < j$. Let $\{\boldsymbol{\xi}_k | k = 1, 2, \cdots, n\}$ be the set of n input vectors with zero-mean, and $\{\mathbf{o}_k | k = 1, 2, \cdots, n\}$ be their corresponding output vectors produced by the network

$$o_{kj} = \mathbf{w_j} \cdot \boldsymbol{\xi}_k + \sum_{l<j} u_{lj} o_{kl}. \tag{3}$$

The weights on connections between the input nodes and the output layer are adjusted upon presentation of an input vector $\boldsymbol{\xi}_k = (\xi_{k1}, \xi_{k2}, \cdots, \xi_{kd})$ according to the Hebbian rule, i.e.,

$$\Delta w_{ij} = \eta \xi_{ki} o_{kj}. \qquad i, j = 1, 2, \cdots, d \tag{4}$$

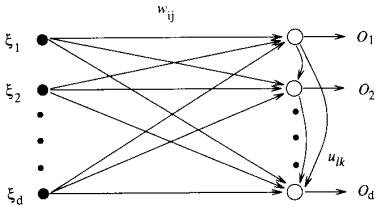


Fig. 2. PCA network proposed by Rubner *et al.*

where η is the learning rate. The lateral synaptic weights, on the other hand, are updated according to the anti-Hebbian rule

$$\Delta u_{lj} = -\mu o_{kl} o_{kj}, \qquad l < j \tag{5}$$

where μ is another positive learning parameter.

Rubner and Tavan [34] proved that if the learning parameters η and μ are properly chosen according to

$$\frac{\eta(\lambda_1 - \lambda_d)}{\lambda_1(1 + \eta\lambda_d)} < \mu < 2/\lambda_1 \tag{6}$$

then all the lateral weights will rapidly vanish and the network will converge to a state in which the d weight vectors associated with the d output units are the d eigenvectors of the covariance matrix of input patterns with eigenvalues, $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_d$. Although, in practice, it is difficult to determine the values of η and μ according to the inequalities in (6) without first computing the eigenvalues, (6) does provide a range for the values of η and μ if λ_1 and λ_d can be somehow estimated. The asymptotically vanishing connection strengths also provide a stopping criterion for the learning procedure.

We have found that introducing the momentum term and letting the learning rate and momentum decay with time can speed up the convergence. In our experiments, we use

$$\Delta w_{ij}(t+1) = \eta(t)\xi_{ki}o_{kj} + \beta(t)\Delta w_{ij}(t) \tag{7}$$

and

$$\Delta u_{lj}(t+1) = -\mu(t)o_{kl}o_{kj} + \beta(t)\Delta u_{lj}(t) \tag{8}$$

where $\eta(t+1) = \max(\alpha\eta(t), 0.001)$, $\mu(t+1) = \max(\alpha\mu(t), 0.002)$, $\beta(t+1) = \max(\alpha\beta(t), 0.001)$, t is the iteration index and α is the decay factor. The nonzero minimum values for η, μ and β enable the network to retain the plasticity so that it can adapt itself to the new input data.

We summarize the learning algorithm for principal component analysis proposed by Rubner *et al.* [33], [34] as follows.

PCA Algorithm

1) Initialize all connection weights to small random values and choose the values of the learning parameters.
2) Randomly select a d-dimensional pattern and present it to the network.
3) Update the connection weights between the input nodes and the output layer according to the Hebbian rule, (7).
4) Normalize each weight vector to a unit vector.
5) Update the lateral weights according to the anti-Hebbian rule, (8).
6) Modify parameters, η, μ and β.

7) If all the lateral weights are sufficiently small for a given number of presentations (their absolute values are below some threshold), then stop; else go to step 2.

Our computer simulations of Runber's PCA network have shown that the precision of the computed eigenvalues and eigenvectors is very high (of the order of 10^{-5} compared to values obtained by the commercial Eispack software) [28]. However, we have experienced a very slow convergence of the PCA algorithm when the input dimensionality d is high and all the d eigenvectors need to be computed, especially when some eigenvalues are very small.

B. Linear Discriminant Analysis (LDA) Network

If the category information of patterns is known, then it is more appropriate to use supervised learning. Linear Discriminant Analysis incorporates the *a priori* category information into the projection or transform by maximizing the between-class scatter while holding the within-class scatter constant.

Let $\boldsymbol{\xi}_i^{(l)} = (\xi_{i1}^{(l)}, \xi_{i2}^{(l)}, \cdots, \xi_{id}^{(l)})^T$ denote the ith pattern in class l, $i = 1, 2, \cdots, n_l$, $l = 1, 2, \cdots, c$, where c is the number of categories. Let $n = \sum_{l=1}^{c} n_l$ denote the total number of patterns. Define the within-class covariance matrix, Σ_W, as

$$\Sigma_W = \frac{1}{n} \sum_{l=1}^{c} \sum_{i=1}^{n_l} (\boldsymbol{\xi}_i^{(l)} - \mathbf{m}^{(l)})(\boldsymbol{\xi}_i^{(l)} - \mathbf{m}^{(l)}) \tag{9}$$

where $\mathbf{m}^{(l)}$ is the mean vector of class l, $l = 1, 2, \cdots, c$. Similarly, define the between-class covariance matrix, Σ_B, as

$$\Sigma_B = \frac{1}{n} \sum_{l=1}^{c} n_l (\mathbf{m}^{(l)} - \mathbf{m})(\mathbf{m}^{(l)} - \mathbf{m})^T \tag{10}$$

where $\mathbf{m}$ is the mean vector of the pooled data. The total scatter matrix is, therefore

$$\Sigma_T = \frac{1}{n} \sum_{l=1}^{c} \sum_{i=1}^{n_l} (\boldsymbol{\xi}_i^{(l)} - \mathbf{m})(\boldsymbol{\xi}_i^{(l)} - \mathbf{m})^T = \Sigma_W + \Sigma_B. \tag{11}$$

The goal of linear discriminant analysis is, then, to find a $d \times m$ transform $\boldsymbol{\Phi}$ such that $|\Phi^T \Sigma_B \Phi| / |\Phi^T \Sigma_W \Phi|$ is maximized, where $|\cdot|$ denotes the determinant. It can be proved that such a transform, $\boldsymbol{\Phi}$, is composed of m eigenvectors corresponding to the m largest nonzero eigenvalues of $\Sigma_W^{-1} \Sigma_B$. Due to the fact that the matrix Σ_B has a maximum rank of $c - 1$, the value of m must be less than c. Therefore, the dimensionality of the projected space is limited by the number of categories. Several variations of linear discriminant analysis have been reported in the literature. For example, Foley and Sammon [6] and Okada and Tomita [32] derived a set of discriminant vectors by selecting the projection axes one at a time under an orthogonality constraint. We use the total covariance matrix Σ_T instead of the between-class covariance matrix Σ_B in the criterion so that the output dimensionality is not limited by the number of categories.

We have proposed a neural network and a supervised self-organizing learning algorithm for multivariate linear discriminant analysis [28]. The linear discriminant analysis (LDA) network is shown in Fig. 3. It consists of two layers, each of

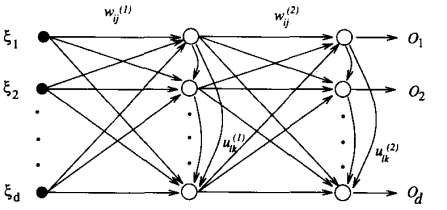


Fig. 3. Architecture of the LDA network.

which is identical to the PCA network proposed by Rubner *et al.* [33], [34] shown in Fig. 2. Linear activation function is used for all the units. Let $w_{ij}^{(1)}$ ($w_{ij}^{(2)}$) be the weights on the interlayer connections from the ith unit in the input (hidden) layer to the jth unit in the hidden (output) layer. Let $u_{ij}^{(1)}(u_{ij}^{(2)})$ denote the lateral weights from the ith unit to the jth unit within the hidden (output) layer. Moreover, let $\boldsymbol{\xi} = (\xi_1, \xi_2, \cdots, \xi_d)^T$ be a d -dimensional input pattern, and $\boldsymbol{\rho} = (\rho_1, \rho_2, \cdots, \rho_d)^T$ and $\mathbf{o} = (o_1, o_2, \cdots, o_d)^T$ be output vectors of the hidden layer and the output layer, respectively. We can write these outputs as

$$\rho_j = \sum_{i=1}^{d} w_{ij}^{(1)} \xi_i + \sum_{l<j} u_{lj}^{(1)} \rho_l,$$

$$o_j = \sum_{i=1}^{d} w_{ij}^{(2)} \rho_i + \sum_{l<j} u_{lj}^{(2)} o_l, \qquad j = 1, 2, \cdots, d. \tag{12}$$

Let $T = \{\boldsymbol{\zeta}_k^{(l)} | k = 1, 2, \cdots, n_l, l = 1, 2, \cdots, c\}$ denote the set of input training patterns. Let $T_{T0} = \{\boldsymbol{\xi}_k^{(l)}, k = 1, 2, \cdots, n_l, l = 1, 2, \cdots, c\}$ denote the normalized training data set obtained by subtracting each pattern in T by the global mean vector of the pooled training data set T, $\boldsymbol{\xi}_k^{(l)} = \boldsymbol{\zeta}_k^{(l)} - \mathbf{m}$. We form another training data set with class-conditional zero-mean vectors, $T_{W0} = \{\boldsymbol{\eta}_k^{(l)}, k = 1, 2, \cdots, n_l, l = 1, 2, \cdots, c\}$ by subtracting each pattern in T by the mean vector of the corresponding class, $\boldsymbol{\eta}_k^{(l)} = \boldsymbol{\zeta}_k^{(l)} - \mathbf{m}^{(l)}$. The category information is only used to form the class-conditional zero-mean training set T_{W0}. Once this is done, the learning algorithm is fully self-organizing.

The proposed supervised self-organizing learning algorithm is given below.

LDA Algorithm

1) Form training sets T_{W0} and T_{T0} from T.
2) Train the first layer using the PCA algorithm using T_{W0}.
3) Project all the patterns in T_{W0} using the first trained layer; compute the standard deviation for each output unit; scale all the weights on connections to each output unit by the standard deviation associated with this unit.

$$w_{ij}^{(1)\prime} = w_{ij}^{(1)} / \sigma_j, \; i, j = 1, 2, \cdots, d.$$

4) Form data set T_H by collecting the output of the hidden layer when the training set T_{T0} is presented to the input layer.
5) Train the second layer using the PCA algorithm on data set T_H.

The learning parameters for the PCA and LDA algorithms in all the experiments in this paper are specified as follows: $\eta(0) = 0.2$, $\mu(0) = 1.5$, $\beta(0) = 0.1$, $\alpha = 0.99999$. The learning process stops when the sum of weights on the lateral connections is below the threshold $t_u = 0.000001$, or the maximum number of iterations, 1 000 000, is reached.[1] All the data sets are normalized by their maximum range so that these learning parameters satisfy the inequalities in (6).

The LDA algorithm actually performs a simultaneous diagonalization of two covariance matrices, Σ_W and Σ_T, of training data sets, T_{W0} and T_{T0}, respectively. In Step 2 of the LDA algorithm, the first layer tries to find the principal vectors of the data T_{W0}. Step 3 "whitens" the data set T_{W0} and, in the meantime, changes the distribution of the data set T_{T0} when it is transformed by the first layer in Step 4. Then, in Step 5 of the LDA algorithm, the second layer attempts to search for the principal vectors of T_H, the transformed version of T_{T0}.

Let $\boldsymbol{\Phi} = [w_{ij}^{(1)}]_{d\times d}$ and $\boldsymbol{\Phi}_s$ be the weight matrix when Steps 2 and 3 are executed, respectively. Let $\boldsymbol{\Psi} = [w_{ij}^{(2)}]_{d\times d}$ be the weight matrix of the second layer after training. The overall mapping that the network performs is $\mathbf{o}_k^{(l)} = A^T\boldsymbol{\xi}_k^{(l)}$, where $A = \boldsymbol{\Phi}_s\boldsymbol{\Psi}$. We have shown that the columns of matrix A are composed of the eigenvectors of $\Sigma_W^{-1}\Sigma_T$ corresponding to the eigenvalues in the decreasing order [28]. When the learning process is finished, these eigenvalues can be obtained by computing the variances of the output units when the training set T_{T0} is presented to the input nodes. The precision of the neural computation is shown to be high enough for feature selection and projection purposes [28]. A significant advantage of these neural networks over conventional approaches is their plasticity which allows the networks to adapt themselves to new input data. Therefore, they can operate in environments where pattern distributions are slowly changing.

Independently of our work, Kuhnel and Tavan [25] have also investigated a network for linear discriminant analysis, but no simulation and comparison results were provided in [25].

Gallinari *et al.* [9] have established a link between linear discriminant analysis and linear feedforward networks. Their theoretical analysis shows that in a linear two-layer feedforward network with $rank(\Sigma_B)$ units in the hidden layer, trained to minimize the square-error criterion, the role of the hidden layer is to realize a linear discriminant analysis that maximizes the criterion $|\Sigma_{B_h}|/|\Sigma_{T_h}|$, where Σ_B is the between-class covariance matrix in the input space, and Σ_{B_h} and Σ_{T_h} are the between-class covariance and total-class covariance matrices, respectively, in the space spanned by the hidden units. This result provides another method to perform a discriminant analysis of the input data by simply collecting the outputs of the hidden units. Comparing our proposed LDA network with this approach, the LDA network has the following two main advantages. First, the LDA algorithm operates in a "forward-propagation" fashion (first train the first layer, then the second layer, thus training of the two layers is independent), while the most commonly used method, backpropagation algorithm, for training the multilayer feedforward is in a "backpropagation" mode. Therefore, the convergence speed of the LDA algorithm is faster than the backpropagation algorithm. Secondly, the weight updating rules (strictly local) for the LDA network are much simpler than the one used in backpropagation algorithm where the updating a weight needs information to be back-propagated from all the units in the next layer. Therefore, the learning circuit of the LDA network is simpler than the one of backpropagation network (feedforword network trained using backpropagation algorithm).

[1] In this paper, one iteration means one presentation of a pattern.

C. A Neural Network for Sammon's Projection (SAMANN)

Sammon [35] proposed a nonlinear projection technique that attempts to maximally preserve all the interpattern distances. Let $\boldsymbol{\xi}(\mu)$, $\mu = 1, 2, \cdots, n$, be the n d-dimensional patterns, and $\mathbf{y}(\mu)$, $\mu = 1, 2, \cdots, n$, be the n corresponding patterns in the m -dimensional projected space, $m < d$. The mapping error, also called Sammon's stress, is defined as

$$E = \frac{1}{\sum_{\mu=1}^{n-1}\sum_{\nu=\mu+1}^{n} d^*(\mu, \nu)} \cdot \sum_{\mu=1}^{n-1}\sum_{\nu=\mu+1}^{n} \frac{[d^*(\mu, \nu) - d(\mu, \nu)]^2}{d^*(\mu, \nu)} \tag{13}$$

where $d^*(\mu, \nu)$ and $d(\mu, \nu)$ are the distances between pattern μ and pattern ν in the input space and in the projected space, respectively. Euclidean distance is commonly used in this projection algorithm.

Sammon's stress E is a measure of how well the interpattern distances are preserved when the patterns are projected from a high dimensional space to a lower dimensional space. Sammon [35] used a gradient descent algorithm to find a configuration of n patterns in the m-dimensional space that minimizes E. This method views the positions of n patterns in the m–dimensional space as nm variables in an optimization problem. There are many local minima on the energy (or error) surface, and it is unavoidable for the algorithm to get stuck in a local minimum. One usually runs the algorithm several times with different random initial configurations and chooses the configuration with the lowest stress. Sammon's algorithm involves a large amount of computation. Since for every step within an iteration, $n(n-1)/2$ distances have to be computed, the algorithm quickly becomes impractical for a large number of patterns.

Sammon's algorithm does not provide an explicit mapping function governing the relationship between patterns in the original space and in the configuration (projected) space. Therefore, it is impossible in Sammon's algorithm to decide where to place new d-dimensional data in the final m-dimensional configuration created by Sammon's algorithm. In other words, Sammon's algorithm has no generalization capability. To project new data, one has to run the program again on pooled data (old data and new data).

We have proposed an unsupervised backpropagation learning algorithm to train a multilayer feedforward neural network to perform the well-known Sammon's nonlinear projection. The proposed learning algorithm, which needs no category

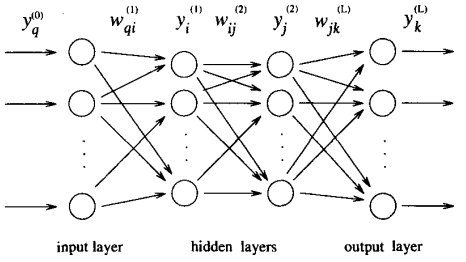


Fig. 4. Three-layer feedforward network for Sammon's projection (SAMANN) and nonlinear discriminant analysis (NDA).

information about patterns, is an extension of the backpropagation algorithm. Fig. 4 shows the neural network architecture. The number of input nodes, d, is set to the input dimensionality of the feature space. The number of output units, m, is specified as the dimensionality of the projected space, $m < d$. In exploratory data analysis, it is customary to use $m = 2$ or 3.

Let $\xi = (\xi_1, \xi_2, \cdots, \xi_d)$ be a d-dimensional input pattern vector. We denote the output of jth unit in layer l by $y_j^{(l)}$, $j = 1, 2, \cdots, n_l$, $l = 0, 1, 2, \cdots, L$, where n_l is the number of units in layer l, L is the number of layers, and $y_j^{(0)} = \xi_j$, $j = 1, 2, \cdots, d$. The weight on connection between unit i in layer $l-1$ and unit j in layer l is represented by $w_{ij}^{(l)}$. We denote $w_{0j}^{(l)}$ as the bias in the jth unit in the lth layer, and $y_0^{(l)} = 1.0$. The sigmoid activation function $g(h)$ whose range is (0.0, 1.0) is used for each unit, where h is the weighted sum of all the inputs to the unit. Therefore, the output of the jth unit in layer l can be written as

$$y_j^{(l)} = g\left(\sum_{i=0}^{n_l} w_{ij}^{(l)} y_i^{(l-1)}\right), \qquad l = 1, 2, \cdots, L. \tag{14}$$

Using the above notation, the $d(\mu, \nu)$ term in (13) can be expressed as

$$d(\mu, \nu) = \left\{\sum_{k=1}^{m} [y_k^{(L)}(\mu) - y_k^{(L)}(\nu)]^2\right\}^{1/2} \tag{15}$$

where μ and ν are the two pattern indices.

Note that the range of each output is between 0.0 and 1.0. If the range of input values is large, it is impossible for the projection algorithm to preserve the interpattern distances no matter which learning rule is used. Therefore, we normalize all the input patterns to equalize the maximum interpattern distances in the original space and in the projected space. Let

$$\lambda = \frac{1}{\sum_{\mu=1}^{n-1} \sum_{\nu=\mu+1}^{n} d^*(\mu, \nu)} \tag{16}$$

which is independent of the network and can be computed beforehand, and

$$E_{\mu\nu} = \lambda \frac{[d^*(\mu, \nu) - d(\mu, \nu)]^2}{d^*(\mu, \nu)} \tag{17}$$

then

$$E = \sum_{\mu=1}^{n-1} \sum_{\nu=\mu+1}^{n} E_{\mu\nu}. \tag{18}$$

Note that $E_{\mu\nu}$ is proportional to the interpattern distance changes between patterns μ and ν, due to the projection from the d-dimensional feature space to the m-dimensional projected space. So, $E_{\mu\nu}$ is more appropriate for the pattern-by-pattern-based updating rules.

We have derived the updating rule for the multilayer feedforward network which minimizes Sammon's stress defined in (13) based on the gradient descent method. For the output layer ($l = L$)

$$\begin{aligned}\frac{\partial E_{\mu\nu}}{\partial w_{jk}^{(L)}} &= \left(\frac{\partial E_{\mu\nu}}{\partial d(\mu, \nu)}\right)\left(\frac{\partial d(\mu, \nu)}{\partial [y_k^{(L)}(\mu) - y_k^{(L)}(\nu)]}\right) \\ &\quad \cdot \left(\frac{\partial [y_k^{(L)}(\mu) - y_k^{(L)}(\nu)]}{\partial w_{jk}^{(L)}}\right) \\ &= \left(-2\lambda \frac{d^*(\mu, \nu) - d(\mu, \nu)}{d^*(\mu, \nu)}\right)\left(\frac{y_k^{(L)}(\mu) - y_k^{(L)}(\nu)}{d(\mu, \nu)}\right) \\ &\quad \cdot \left(g'(h_{k,\mu}^{(L)}) y_j^{(L-1)}(\mu) - g'(h_{k,\nu}^{(L)}) y_j^{(L-1)}(\nu)\right)\end{aligned} \tag{19}$$

where $g'(h_{k,\cdot}^{(L)})$ is the derivative of the sigmoid function of unit k in layer L (output layer) with respect to the net input, $h_{k,\cdot}$, of this unit

$$g'(h_{k,\cdot}) = (1 - y_k^{(L)}(\cdot)) y_k^{(L)}(\cdot). \tag{20}$$

Let

$$\delta_k^{(L)}(\mu, \nu) = -2\lambda \frac{d^*(\mu, \nu) - d(\mu, \nu)}{d^*(\mu, \nu) d(\mu, \nu)} [y_k^{(L)}(\mu) - y_k^{(L)}(\nu)] \tag{21}$$

$$\Delta_{jk}^{(L)}(\mu) = \delta_k^{(L)}(\mu, \nu)[1 - y_k^{(L)}(\mu)] y_k^{(L)}(\mu) \tag{22}$$

$$\Delta_{jk}^{(L)}(\nu) = \delta_k^{(L)}(\mu, \nu)[1 - y_k^{(L)}(\nu)] y_k^{(L)}(\nu) \tag{23}$$

where $\delta_k^{(L)}(\mu, \nu)$ is the change in the output scaled by normalized interpattern distance change when we present patterns μ and ν. As we will see later, $\Delta_{jk}^{(L)}(\mu)$ and $\Delta_{jk}^{(L)}(\nu)$ will be backpropagated to layer $L-1$. Substituting (20)–(23) into (19), we get

$$\frac{\partial E_{\mu\nu}}{\partial w_{jk}^{(L)}} = \Delta_{jk}^{(L)}(\mu) y_j^{(L-1)}(\mu) - \Delta_{jk}^{(L)}(\nu) y_j^{(L-1)}(\nu). \tag{24}$$

The updating rule for the output layer is

$$\begin{aligned}\Delta w_{jk}^{(L)} &= -\eta \frac{\partial E_{\mu\nu}}{\partial w_{jk}^{(L)}} \\ &= -\eta\left(\Delta_{jk}^{(L)}(\mu) y_j^{L-1}(\mu) - \Delta_{jk}^{(L)}(\nu) y_j^{(L-1)}(\nu)\right)\end{aligned} \tag{25}$$

where η is the learning rate.

Similarly, we can obtain the general updating rule for all the hidden layers, $l = 1, \cdots, L-1$

$$\Delta w_{ij}^{(l)} = -\eta \frac{\partial E_{\mu\nu}}{\partial w_{ij}^{(l)}} = -\eta\left(\Delta_{ij}^{(l)}(\mu) y_i^{(l-1)}(\mu) - \Delta_{ij}^{(l)}(\nu) y_i^{(l-1)}(\nu)\right) \quad (26)$$

where

$$\Delta_{ij}^{(l)}(\mu) = \delta_j^{(l)}(\mu)[1 - y_j^{(l)}(\mu)] y_j^{(l)}(\mu) \quad (27)$$

$$\Delta_{ij}^{(l)}(\nu) = \delta_j^{(l)}(\nu)[1 - y_j^{(l)}(\nu)] y_j^{(l)}(\nu) \quad (28)$$

and

$$\delta_j^{(l)}(\mu) = \sum_{k=1}^{m} \Delta_{jk}^{(l+1)}(\mu) w_{jk}^{(l+1)} \quad (29)$$

$$\delta_j^{(l)}(\nu) = \sum_{k=1}^{m} \Delta_{jk}^{(l+1)}(\nu) w_{jk}^{(l+1)}. \quad (30)$$

Analogous to the backpropagation learning algorithm, $\delta_j^{(l)}(\mu)$ and $\delta_j^{(l)}(\nu)$ are changes in layer l backpropagated from its successive layer, $l+1$, for pattern μ and pattern ν, respectively. As we can see from (25) and (26), to update the weights, we need to present a pair of patterns to the network instead of one pattern at a time in the backpropagation learning algorithm. To do this, we can either build two identical networks or just store all the outputs of the first pattern before we present the second pattern. The above learning rule for Sammon's projection makes use of the popular backpropagation algorithm. But, we do not need category information of the patterns. Therefore, we have extended the backpropagation algorithm to perform unsupervised learning.

The SAMANN unsupervised backpropagation algorithm is summarized below.

SAMANN Unsupervised Backpropagation Algorithm

1) Initialize weights randomly in the SAMANN network.
2) Select a pair of patterns randomly, present them to the network one at a time, and evaluate the network in a feedforward fashion.
3) Update the weights using (25) and (26) in the backpropagation fashion starting from the output layer.
4) Repeat steps 2–3 a number of times.
5) Present all the patterns and evaluate the outputs of the network; compute Sammon's stress; if the value of Sammon's stress is below a prespecified threshold or the number of iterations (from steps 2–5) exceeds the prespecified maximum number, then stop; otherwise, go to step 2.

Similar to the backpropagation algorithm, we can add a momentum term in updating the weights to speed up the convergence.

The selection of the number of hidden layers and the number of units in each hidden layer in a multilayer feedforward network is an important yet difficult problem. To achieve the representation power of Sammon's algorithm, we have to use a network with at least mn free parameters, where mn is the number of variables in Sammon's algorithm. So, mn becomes a lower bound for the total number of free parameters. This lower bound becomes very large when the number of patterns is large. However, we have found that it is often not necessary to use such a large number of free parameters to obtain reasonable projection maps. On the other hand, the network should be as small as possible to obtain good generalization capability. A compromise between these two factors is necessary for selecting an appropriate network.

A significant advantage of the SAMANN network is that the network is able to project new patterns after training because of the effect of regularization [29]. Some preliminary simulation results were reported in [19], which demonstrated this good generalization capability.

The local minimum problem in both the methods can be improved by choosing good initial configuration in the original Sammon's algorithm and good initial weights in the SAMANN network. We have found that the values of Sammon's stress of principal component analysis and of Sammon's projection are close for many data sets. This is because when all the interpattern distances in a data set are maximally preserved, the variance of the data is also retained to a very high degree. Therefore, we propose to use the PCA projection map as an initial configuration in Sammon's projection. The PCA projection map can be obtained from the PCA network. Then, the SAMANN network is trained using the standard backpropagation algorithm to approximate principal component analysis. Finally, the proposed unsupervised backpropagation algorithm is used to take advantage of the nonlinearity of the SAMANN network to refine the projection map.

In all the experiments reported in this paper, a two-layer (one hidden layer) network with 20 hidden units is used. The SAMANN network is trained using the standard backpropagation algorithm with a learning rate of 0.7 and momentum value of 0.3 for 200 000 iterations. Then, the unsupervised backpropagation algorithm is used to train the SAMANN network for 60 000 iterations. The learning rate and momentum in the unsupervised backpropagation algorithm are 0.02 and 0.01, respectively, for the artificial data sets, and 0.05 and 0.02, respectively, for the real data sets.

D. A Nonlinear Projection Based on Kohonen's Self-Organizing Map (NP-SOM)

Kohonen's self-organizing map (SOM) [20]–[22] belongs to the category of unsupervised nonlinear projection methods. Kohonen's Self-Organizing map has a very desirable property of topology preserving, which captures an important aspect of the feature maps in human brain. The network architecture is shown in Fig. 5. It basically consists of a two-dimensional array of units, each of which is connected to all the d input nodes. Let $\mathbf{w}_{ij}$ denote the d-dimensional vector associated with the unit at location (i, j) of the 2-D array.

A slightly alternative formulation of the learning rule, which is used in this paper, is the kernel updating rule [22].

$$\mathbf{w}_{ij}(t+1) = \mathbf{w}_{ij}(t) + h_{c_i c_j}(t)[\boldsymbol{\xi}(t) - \boldsymbol{w}_{ij}(t)] \quad (31)$$

where $h_{c_i c_j}(t)$ is a Gaussian weighting function

$$h_{c_i c_j}(t) = h_0(t) \exp\left(-\frac{[(i-c_i)^2 + (j-c_j)^2]}{2\sigma(t)^2}\right). \quad (32)$$

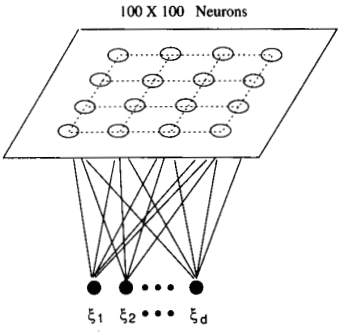


Fig. 5. A sketch of the NP-SOM network.

Here, $h_0(t)$ and $\sigma_0(t)$ are chosen as suitable decreasing functions of time. We use the following parameters in all our experiments: $h_0(0) = 0.05$, $\sigma(0) = 66.666$, $h_0(t+1) = \max\{0.9999 \times h_0(t), 0.0001\}$, $\sigma(t+1) = \max\{0.9999 \times \sigma(t), 1.0\}$, and the maximum number of iterations is 100 000.

Because of the topology-preserving property of the SOM map, some insights into the data can be gained by examining the activation pattern of the 2D array of units when all the input patterns are presented to the network after the training is done. If the category information of the data is available, we can label each unit in the array by the majority class label of the patterns which are projected onto the unit. If the classes in the data are well-separated, the 2D projection map will be divided into regions by classes. Kohonen [21] used this method to obtain the phoneme map which was used in the phonetic typewriter. Since each unit in the network has a trend to encode (become a prototype of) a region in the input space according to the density distribution of the data, all the units have approximately the same chance to be activated. Therefore, patterns in the SOM map are quite uniformly distributed. This makes it very difficult to visualize the structure of the data when the category information is not available. Moreover, we are not able to display high dimensional weight vector associated with each unit. So, Kohonen's SOM maps are not suitable for visualization of high dimensional data with no category information.

Kohonen's Self-Organizing Map Algorithm

1) Initialize weights to small random numbers, set initial learning rate and neighborhood;
2) Present a pattern $\boldsymbol{\xi}$, and evaluate the network outputs;
3) Select the unit (c_i, c_j) with the minimum output:

$$\|\boldsymbol{\xi} - \mathbf{w}_{c_i c_j}\| = \min_{ij} \|\boldsymbol{\xi} - \mathbf{w}_{ij}\|$$

4) Update all the weights according to the kernel-based learning rule;

$$\mathbf{w}_{ij}(t+1) = \begin{cases} \mathbf{w}_{ij}(t) + \alpha(t)\,[\boldsymbol{\xi}(t) - \mathbf{w}_{ij}(t)], & \text{if } (i,j) \in N_{c_i c_j}(t), \\ \mathbf{w}_{ij}(t), & \text{otherwise,} \end{cases}$$

where $N_{c_i c_j}(t)$ is the neighborhood of unit (c_i, c_j) at time t, and $\alpha(t)$ is the learning rate.

5) Decrease the value of $\alpha(t)$ and shrink the neighborhood $N_{c_i c_j}(t)$;
6) Repeat steps 2–5 until the change in weight values is less than a prespecified threshold, or the maximum number of iterations is reached.

We have developed a nonlinear projection method (NP-SOM) based on Kohonen's SOM to facilitate visualization of data, where a 100×100 array of units is used [23]. A sketch of the NP-SOM network is shown in Fig. 5. First, we train the network using Kohonen's self-organizing learning algorithm on a data set. Then, we display the network as a two-dimensional distance image, in which each pixel represents a unit in the 2-D network. The gray value of a pixel is the maximum distance in the weight vector space between the corresponding unit and its four neighbors. In the distance image, dark regions separated by bright curves represent different clusters. This display technique helps us in visualizing the cluster tendency and underlying structure of the data. Experiments in Section III will show that the cluster tendency of many data sets becomes apparent in the 2D distance image. The inter-unit distance is also defined which enables us to compute Sammon's stress for this projection [23].

E. Nonlinear Discriminant Analysis (NDA) Network

Webb and Lowe [37] have investigated a class of multilayer feedforward networks with nonlinear hidden units and linear output units, trained to minimize the squared error between the desired output and actual output of the network. They have found that the nonlinear transformation implemented by the subnetwork from the input layer to the final hidden layer of such networks maximizes the so-called network discriminant function, $\mathrm{Tr}\{S_B S_T^+\}$, where S_T^+ is the pseudo-inverse of the total scatter matrix of the patterns at the output of the final hidden layer, and S_B is the weighted between-class scatter matrix of the output of the final hidden layer (the weights are determined by the coding scheme of the desired outputs). Although this result was derived for the specific class of multilayer feedforward networks with nonlinear hidden units and linear output units, its interpretation can be extended to general multilayer feedforward networks. The role of hidden layers is to implement a nonlinear transformation which projects input patterns from the original space to a space in which patterns are easily separated by the output layer. The first part of the network (from the input layer to the output layer) actually realizes a nonlinear discriminant analysis.

We have proposed to use this theoretical result to perform a nonlinear discriminant analysis (NDA) of input data [28]. The main objective of this method is to visualize high dimensional data to determine whether it is necessary to compute a nonlinear decision boundary in the output layer (e.g., by using another multilayer network, or k-nearest neighbor classifier). This projection method falls into the category of supervised nonlinear approaches. The network architecture is the same as the SAMANN (Fig. 4). We specify the number of input nodes to be the number of features and the number of units in the output layer to be the number of pattern classes in the data set. Furthermore, we fix the number of units in the last hidden layer to m, the dimensionality of the projected space. Thus, from input layer to the last hidden layer, the network

TABLE I
SOME FEATURES OF THE FIVE REPRESENTATIVE NETWORKS

Network	PCA	LDA	SAMANN	NDA	NP-SOM
Class in taxonomy of Fig. 1	unsupervised linear	unsupervised linear	unsupervised nonlinear	supervised nonlinear	unsupervised nonlinear
Architecture	Figure 2	Figure 3	Figure 4	Figure 4	Figure 5
Activation function	linear	linear	sigmoid	sigmoid	Euclidean
No. inputs	d	d	d	d	d
No. outputs	m	m	m	c	100×100
No. layers	1	2	> 2	> 2	1
Learning algorithm	PCA (Box 1)	LDA (Box 2)	unsupervised backpropagation (Box 3)	standard backpropagation	Kohonen's SOM (Box 4)

implements a nonlinear projection from d-dimensional space to m-dimensional space. If the entire network can correctly classify a linearly-nonseparable data set, then this projection actually transforms the linearly-nonseparable data to a linearly-separable data in some "other" space. Therefore, by visualizing the data in this m-dimensional space ($m = 2, 3$), we can gain some insights into the structure of the data. In all our simulations, we use $m = 2$. The backpropagation learning algorithm is used to train the feedforward network with two hidden layers. Sigmoid activation function is used in all the units. In order to avoid squashing the data for the projection map, we replace the sigmoid activation function by a linear function in all the units in the last hidden layer after the training is done.

F. Summary of Networks for Feature Extraction and Data Projection

Table I lists several important features of the five representative networks for feature extraction and data projection: PCA, LDA, SAMANN, NDA, and NP-SOM.

III. COMPARATIVE STUDY

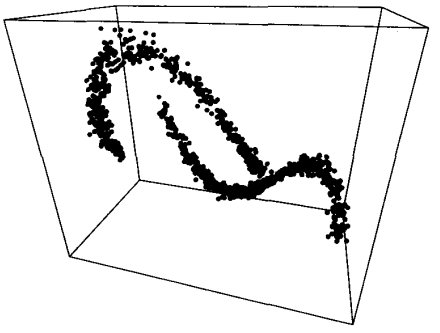
We will evaluate the performance of the five networks for feature extraction and data projection based on a visual judgment of the 2D projection maps and three numerical criteria over eight data sets with various properties. These networks are: i) Rubner's PCA network, ii) LDA network, iii) NDA network, iv) network for Sammon's projection (SAMANN), and v) the nonlinear projection method based on Kohonen's self-organizing map (NP-SOM).

A. Methodology

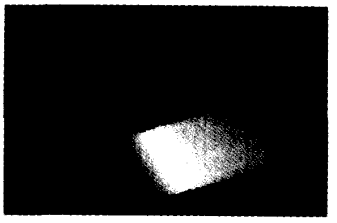
We have generated or collected eight data sets for evaluating feature extraction and data projection approaches. These eight data sets are briefly described below.

1) Two standard normally distributed clusters ($\Sigma_1 = \Sigma_2 = I$ and $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_2 = (1, 1, \cdots, 1)^T$) in a 10-dimensional space with 500 patterns per class. This is a well-separated data set with a Bayes error of 0.078%.
2) Two elongated clusters in a three-dimensional space. A perspective view of this data set is shown in Fig. 6(a). This data set is not linearly separable.
3) Two uniformly distributed sets of points on two 3-dimensional spherical surfaces. One sphere containing 800 patterns is centered at (0, 0, 0) with radius 1, and the other containing 200 patterns is centered at (0, 0, 0.2) with radius 0.1. This data set has a relatively symmetric structure and its two classes are not linearly separable.
4) Uniformly distributed "noise" in a 10-dimensional unit hypercube. All the 1 000 points are randomly labeled into two classes with 500 points per class. This example is used to test how feature extraction and data projection algorithms behave on data sets with no meaningful structure.
5) IRIS data set. The well-known data set consisting of 150 four-dimensional patterns from three classes. It contains four measurements on 50 flowers from each of the three species of the Iris flower.
6) 8OX data set. It consists of 45 eight-dimensional patterns from three classes (the handprinted characters "8", "O" and "X") with 15 patterns per class. This is a very sparse data set.
7) Data set containing four features extracted from a range image [11]. Fig. 6(b) shows the range image of a polyhedral object. The gray level at each pixel in this range image encodes the distance (depth) from the sensor to the corresponding point on the object's surface. The four features are the three components of the surface normal vector and the depth value at each pixel. A total of 1 000 pixels are randomly chosen from the polyhedral object. These 1 000 patterns are labeled with five classes (four visible surfaces plus a class of boundary and edge points).
8) Data set containing 22 Gabor filter features [18] extracted from an image with 16 different textures shown in Fig. 6(c). A total of 1 000 pixels are randomly chosen from the image (512×512).

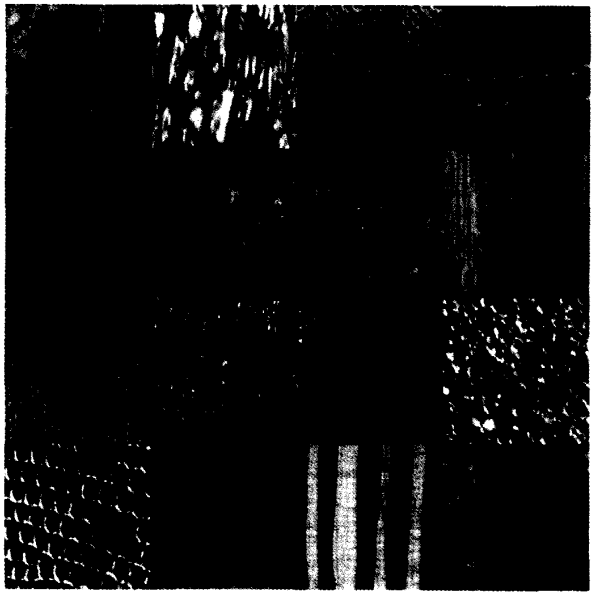
These eight data sets differ from each other in one or more characteristics, such as the data source (artificial/real data), dimensionality of the pattern vector (d), linear dimensionality (d_l), number of classes/clusters (c), number of patterns (n), linear separability (λ_s), inherent structure of the data, and sparseness. The linear dimensionality (d_l) of a data set is measured by the number of significant eigenvalues (more than 97% of the total variance is retained by the first d_l principal components) of the covariance matrix of the data. The linear separability (λ_s) is defined as the largest eigenvalue of $\Sigma_T^{-1}\Sigma_B$ [9]. Note that λ_s is restricted to the [0.0,1.0] range. As λ_s increases from 0.0 to 1.0, the data set becomes more and more linearly separable. The inherent structure of the data is ranked as "random," "weak," "mediun" or "strong" based on our *a priori* knowledge about the data set. The sparseness of a data set is measured by the ratio of the dimensionality to the number of patterns in the data set; the larger this ratio, the sparser the data. This index does not take the range of data into consideration because the range can be easily scaled. These characteristics of the eight data sets are summarized in Table II, from which we can see that they constitute a reasonable benchmark for the evaluation of feature extraction and data projection methods.



(a)



(b)



(c)

Fig. 6. Some data sets used in the comparative study. (a) A perspective view of data set 2. (b) Range image and (c) texture image on which data sets 7 and 8 are extracted, respectively.

The performance of the five neural networks on these eight data sets are evaluated based on visual judgment of the 2D projection maps and three numerical criteria. The three criteria proposed in [23] are modified and utilized in this paper for the numerical evaluation of neural networks for feature extraction and data projection. These are: i) Sammon's stress (13), ii) the nearest-neighbor classification error rate P_e^{NN} on projected data, and iii) the minimum-distance classification error rate P_e^{MD} of projected data. Since the classification error of the projected data P_e(projection) depends on the classification error of the original data P_e(original), we propose to use a normalized ratio

$$R_e = \frac{1 + P_e(\text{projection})}{1 + P_e(\text{original})}. \tag{33}$$

The value of R_e is within the range of (0.5, 2.0). When $R_e = 1.0$, it means that the extracted features have the same discriminatory power as the original features for a given classifier. If $R_e < 1.0$, the extracted features have a better performance in terms of classification accuracy. This could happen, for example, if the effect of the "curse of dimensionality" is eliminated by the feature extraction and data projection. The Sammon's stress measures how well the projection preserve the interpattern distances. The nearest-neighbor classification error rate indicates how well the extracted features or projection preserve the local category structure of the data, while the minimum-distance classification error rate provides some information on the linear separability in the new feature space. The error rates are evaluated using the "leave-one-out" (cross-validation) technique.

TABLE II
CHARACTERISTICS OF THE EIGHT DATA SETS

Data set	Name	source	d	d_i	c	n	λ_*	structure	sparseness
1	Gauss2	artificial	10	10	2	1000	0.91	strong	0.01
2	Curve2	artificial	3	2	2	1000	0.70	strong	0.003
3	Sphere2	artificial	3	3	2	1000	0.38	strong	0.003
4	Random	artificial	10	10	1(2)	1000	0.48	random	0.01
5	IRIS	real	4	2	3	150	0.97	medium	0.027
6	8OX	real	8	7	3	45	0.87	weak	0.178
7	Range	real	4	3	5	1000	0.81	strong	0.004
8	Texture	real	22	15	16	1000	0.95	medium	0.022

B. Visualization of Projection Maps

Information about the structure, intrinsic dimensionality, cluster tendency, and cluster shape of the data is very useful in choosing a proper classification or clustering method. However, this information is not directly accessible for high dimensional data. Data projection is a tool which is frequently used to explore various properties of the data. Here, we project all the eight data sets onto two dimensions using the five networks. From these maps, we can observe not only some of the properties of the eight data sets, but also different characteristics of the five networks. Figs. 7–14 show the 2D projection maps for the eight data sets (For these data sets containing 1 000 patterns, only 500 randomly-chosen patterns are displayed). In each of these figures, there are four 2D maps which are generated by the following four networks: PCA, LDA, SAMANN and NDA networks. The projection maps obtained by the NP-SOM network will be discussed later.

The PCA network performs a linear orthogonal projection. The 2D PCA map produced by the PCA network is spanned by two orthogonal vectors (in the original space) along which the data have the largest and the second largest variances. Therefore, PCA maps are the easiest one to interpret. For some of the data sets, for example, *Gauss2* (Fig. 7(a)), IRIS (Fig. 11(a)), and *Range* (Fig. 13(a)), their PCA maps also

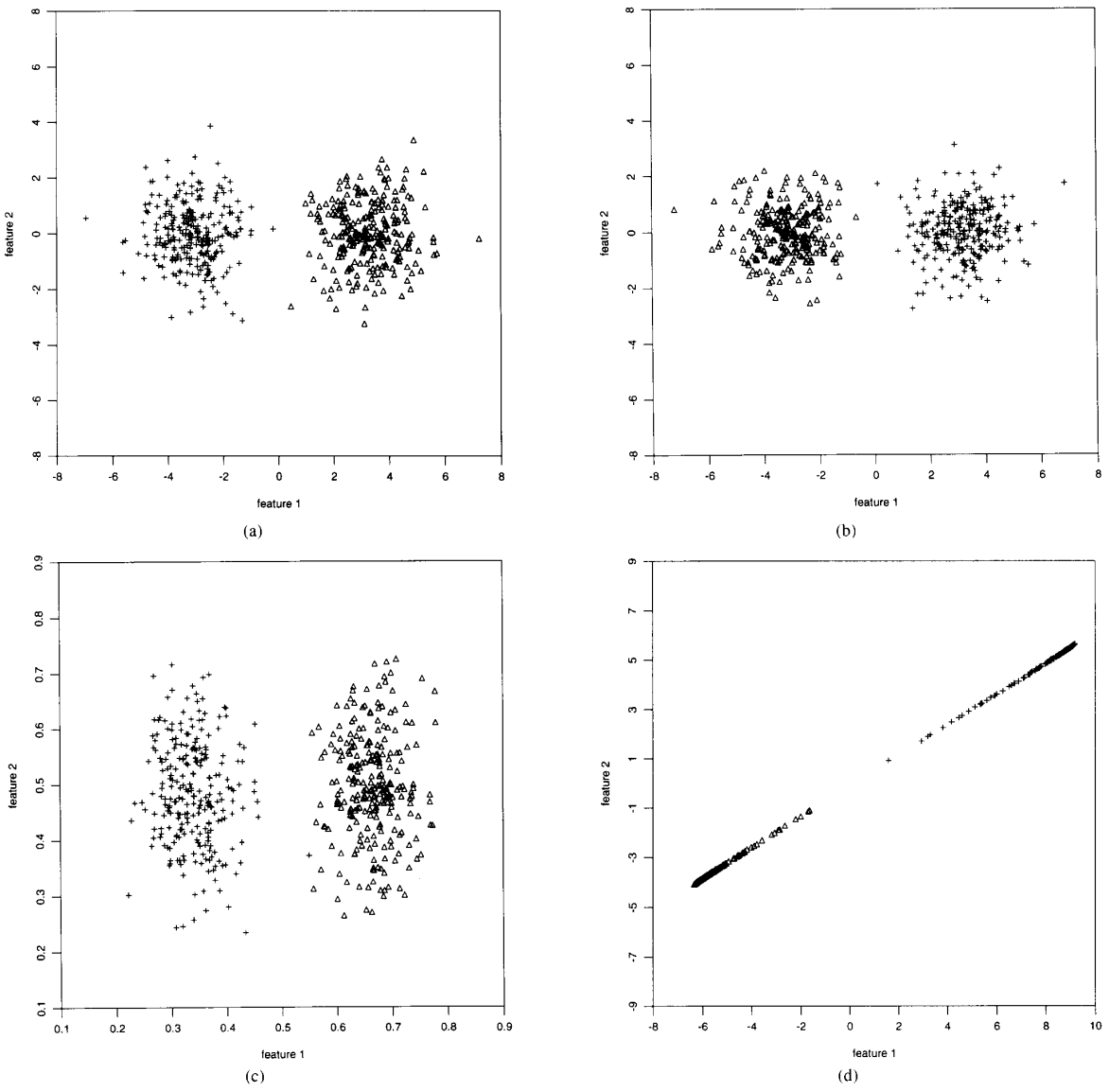


Fig. 7. 2D projection maps of data set *Gauss2* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network.

exhibit much of the class-discriminatory information. But, this is not always true. For some other data sets, e.g., *Sphere2* (Fig. 9(a)), *Curve2* (Fig. 8(a)), and *8OX* (Fig. 12(a)), patterns from different classes either overlap or are not linearly separable in the PCA projection.

The LDA network also implements a linear, but not necessarily orthogonal, projection. It attempts to find a plane in the high dimensional space and then skews the plane in order to increase the discriminatory ability of both the projection axes. Examples are the LDA maps of the data sets: *Curve2* (Fig. 8(b)), *8OX* (Fig. 12(b)) and *IRIS* (Fig. 11(b)). Note that the elongated clusters in the PCA map of the *Range* data are squashed in the LDA map. However, the LDA network can not improve the linear separability of the data sets *Sphere2* (Fig. 9(b)) and *Random* (Fig. 10(b)), because it is a linear method.

The SAMANN maps are very similar to the PCA maps for all the eight data sets. This is not only because the SAMANN network uses the PCA map as the initial pattern configuration which determines the initial weights of the SAMANN network, but also because the Sammon's projection attempts to maximally preserve the interpattern distances, which means that it also preserves the variance as much as possible. However, the SAMANN network can refine the PCA map using its inherent nonlinearity in order to obtain a map with less distortion in interpattern distances.

The NDA network tries to reorganize and group all the patterns by categories such that patterns in the projection

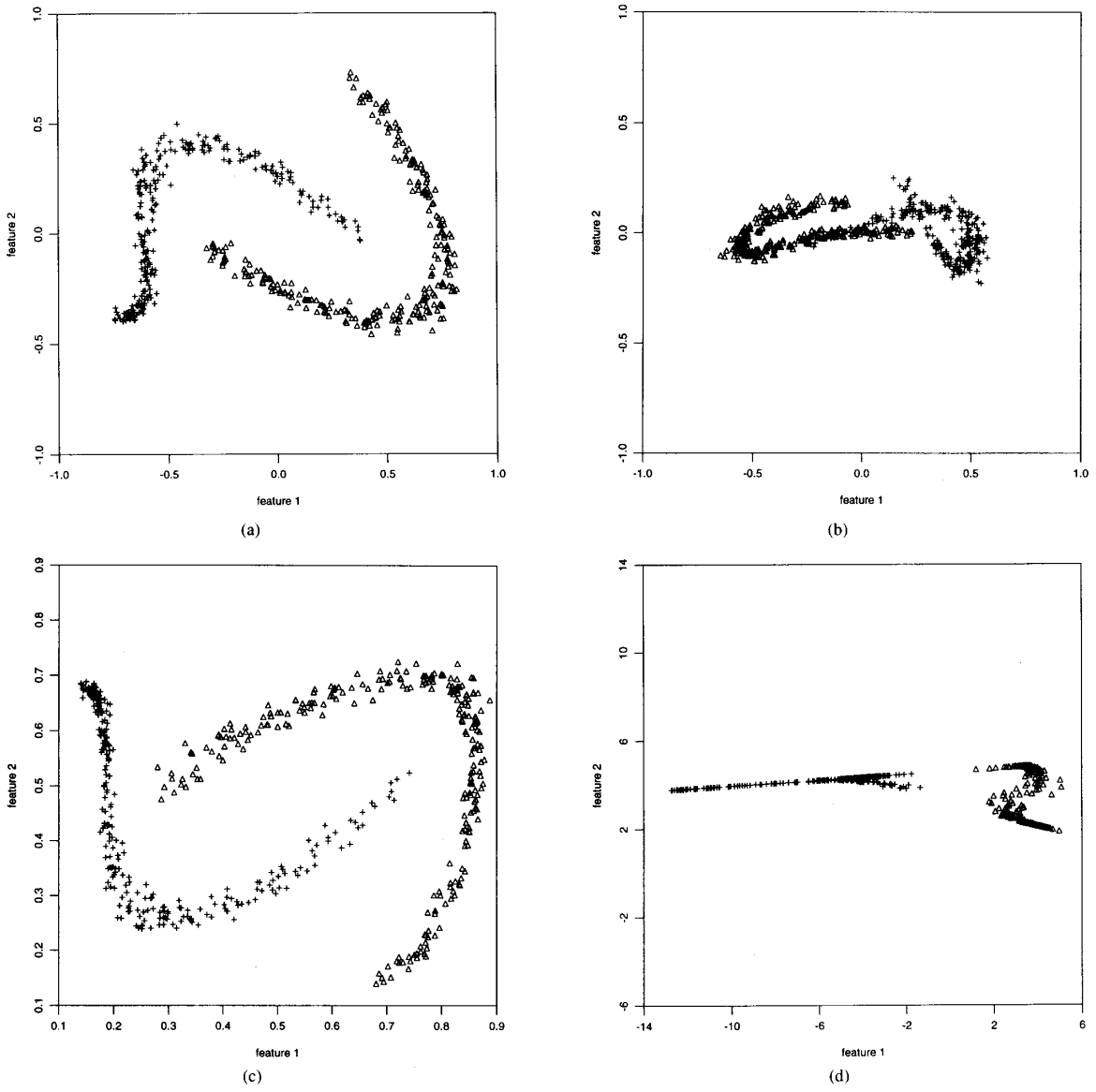


Fig. 8. 2D projection maps of data set *Curve2* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network (2 classes become linearly well-separable).

map can be separated much more easily. This property is fully demonstrated on all the eight data sets, especially on the linearly nonseparable data sets such as, *Curve2* (Fig. 8(d)) and *Sphere2* (Fig. 9(d)). It is remarkable to see that the data points on the entire outer sphere are stripped off from the inner sphere (see the small black "blob" which is isolated from a cloud of patterns (on the outer sphere) in Fig. 9(d))! This indicates that incorporating a nonlinear transform as well as category information lead to a very powerful method for projecting multivariate data.

Fig. 15 shows the distance images of Kohonen's SOM superimposed on the 2D projection maps (white dots) for the eight data sets using the NP-SOM network. We can see that these distance images are divided into several dark regions separated by bright wide curves. Pixels (units) in the same dark region usually have relatively small distances to each other in the original feature space. The cluster tendency of the eight data sets becomes apparent from these distance images. Data sets, *Gauss2* (Fig. 15(a)), *Curve2* (Fig. 15(b)), *Sphere2* (Fig. 15(c)) and *Range* (Fig. 15(g)), have well-separated (linearly or nonlinearly) clusters. The *IRIS* data set (Fig. 15(e)) has two well-isolated clusters, which is consistent with our *a priori* knowledge about the *IRIS* data set (two of the three categories are slightly overlapping in the feature space). The *Random* data set has no cluster tendency. The number of clusters in a data set can be roughly estimated by counting the number

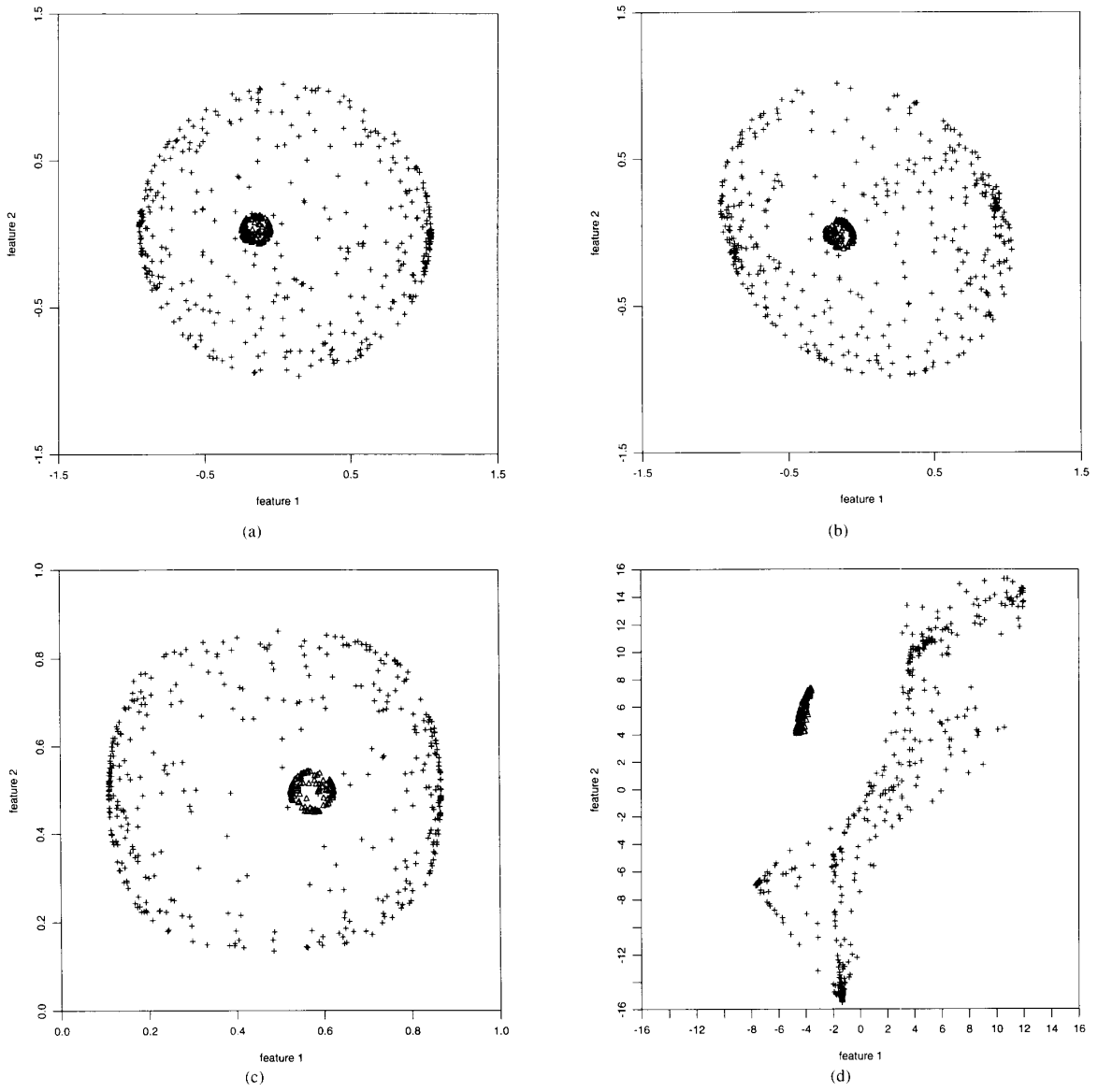


Fig. 9. 2D projection maps of data set *Sphere2* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network (the black blob exclusively contains all the patterns on the inner sphere).

of dark regions in the corresponding distance image. This is easy to do with data sets containing well-isolated clusters, but not trivial with others. The separation between two clusters is indicated by the brightness of the boundary between them. It is interesting to notice that each pattern in the *8OX* data set (Fig. 15(f)) occupies a separate dark region. This is because *8OX* data set (containing only 45 patterns in an 8D space) is too sparse and the size of the map (100×100) is too large. Since the category information for all the data sets except for the *Random* data set is known, we can determine the class labels for these white dots on the projection maps. If more than one pattern is projected into the same unit (pixel) in the map, the class label of the pixel is determined by the majority pattern class (ties are broken arbitrarily). We have manually drawn the rough boundaries (bright thin curves) between different classes on the projection maps, which are superimposed on the corresponding distance images. We can see from Fig. 15 that the 2D projection maps fit the corresponding distance images very well. The patterns from the same class are grouped into a single region except for the *Random* data set. For the *IRIS* data set (Fig. 15(e)), even though there is no clear bright curve between two of the three classes, the two classes occupy different regions in the image. For the *Range* data set, most patterns from the class containing edge (jump and crease) points in the range image (Fig. 6(b)) are scattered over a bright region in Fig. 15(g), which implies that these patterns do not

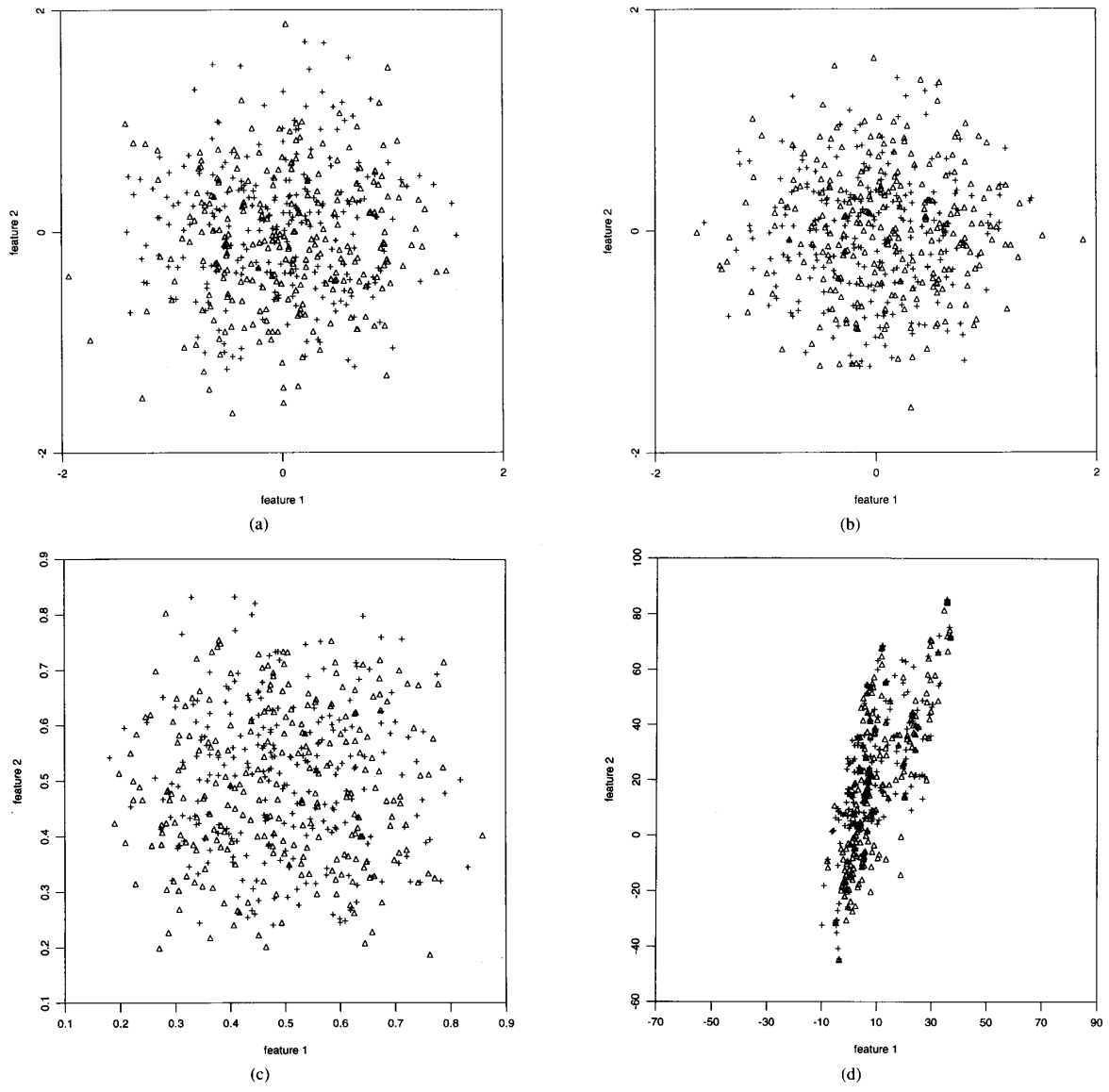


Fig. 10. 2D projection maps of data set *Random* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network.

form a compact cluster in the feature space, and should be considered as outliers. This observation is consistent with the fact that the surface normals at these points on the object are not reliably estimated. The distance images are very helpful in exploring the structure of the data, when the category information is not available. Since the patterns are relatively uniformly distributed in the 2D projection maps generated by the NP-SOM network compared to the PCA and SAMANN networks, it is very difficult to observe the cluster tendency based on the unlabeled 2D projection map for most data sets without the distance image.

Properties of the eight data sets revealed by visualizing them in 2D maps can suggest what type of classification or clustering algorithms to use. For the *Gauss2* data set, a linear classifier or a squared-error clustering algorithm can be applied. In fact, most classifiers and clustering algorithms will work well on this data set. We also notice that one single feature (the first principal component) contains almost all the discriminatory information. On the other hand, from the projection maps of the *Curve2* and *Sphere2* data sets, linear classifiers are not powerful enough to distinguish the two classes in these data sets. It becomes also clear that the square-error clustering algorithm is not suitable for these two data sets. These projection maps suggest the use of nonlinear classifiers such as, the k-nearest neighbor classifier and multilayer feedforward networks, or some hierarchical

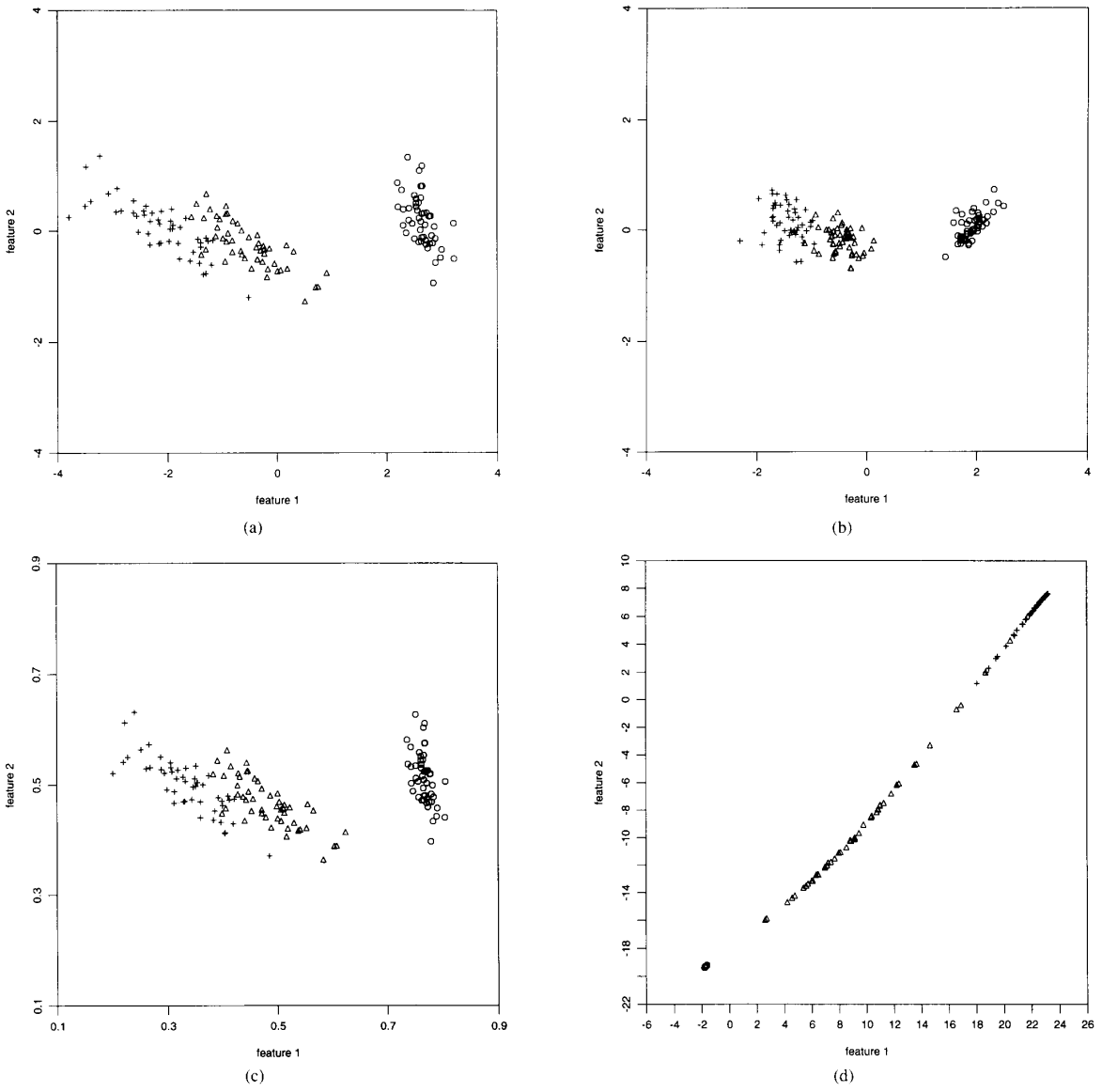


Fig. 11. 2D projection maps of the *IRIS* data set using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network (the black dot on the lower-left corner exclusively contains all the 50 patterns from class 1).

clustering algorithms [17] such as, single-link and complete-link clustering algorithms. For the data sets, *IRIS*, *8OX* and *Texture*, both linear and nonlinear classifiers can be applied, but a nonlinear classifier may give a better performance. The square-error clustering algorithm may also be applied to these data sets. Moreover, from the projection maps of the *Texture* data (Fig. 14), it is apparent that two extracted features are not sufficient for classification. This will be demonstrated further by the classification error rate using the 2D projection maps. An interesting property of the *Range* data set revealed by the projection maps in Fig. 13 is that this data contains mainly four elongated clusters. These elongated clusters are roughly linearly separable. This property suggests that a square-error clustering algorithm with the Euclidean distance metric should not be used for this data set.

C. Evaluation Based on Quantitative Criteria

We have seen various characteristics of the five networks for the purpose of visualizing high dimensional data. In this subsection, we provide a quantitative evaluation of these five networks based on three criteria using the eight data sets. The dimensionality of the output space is two for all the five networks. Every simulation of each of the five networks on each of the eight data sets has been repeated 10 times with different initial random weights in the networks. The average

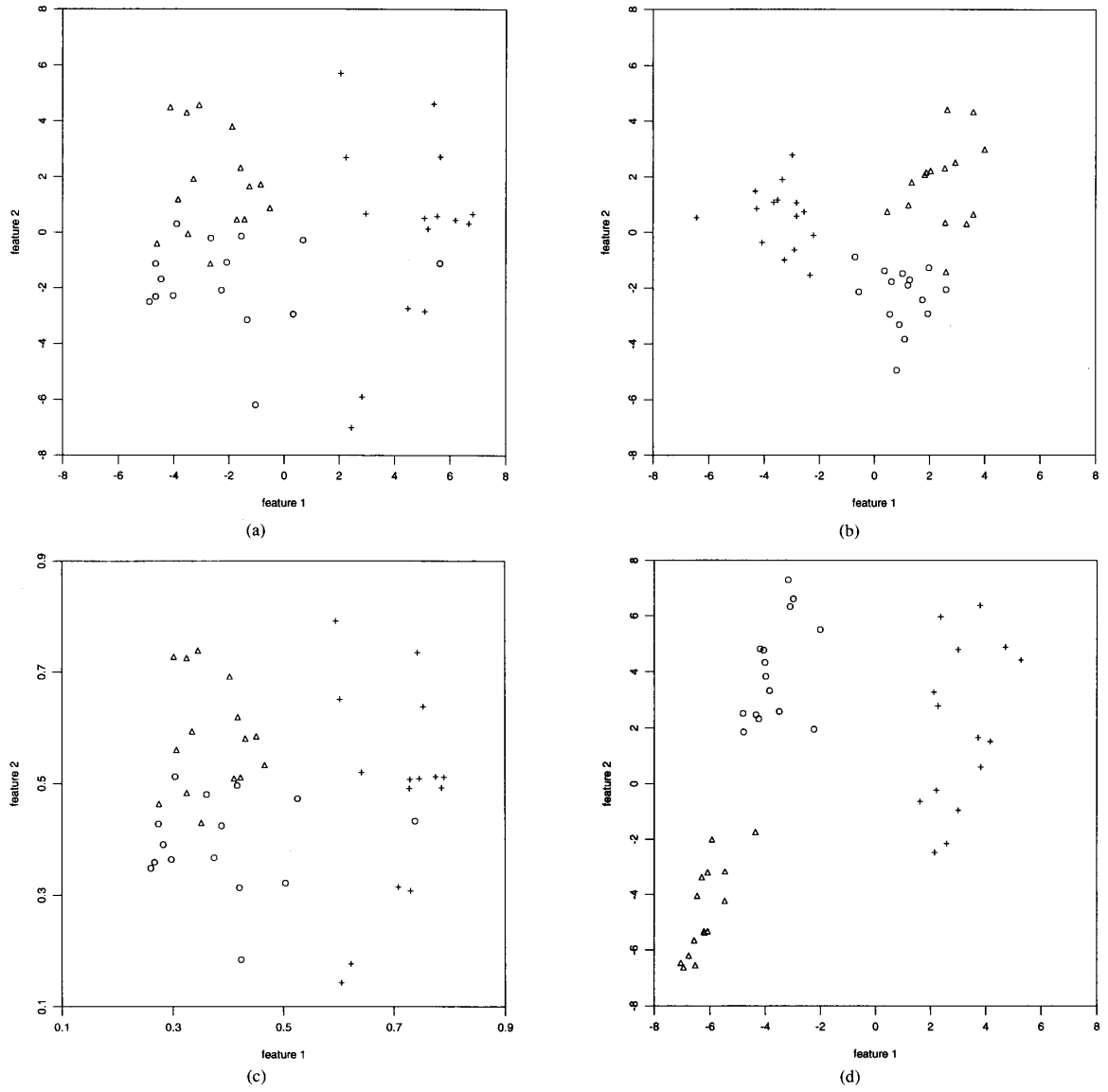


Fig. 12. 2D projection maps of the *8OX* data set using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network.

performance over these 10 trials is reported in the following tables.

Table III shows the average values of Sammon's stress of the projections performed by the five networks on the eight data sets. Among the five networks, the SAMANN network performs the best on seve out of the eight data sets in terms of Sammon's stress. The exception is the *Curve2* data set on which the PCA network has the lowest Sammon's stress. This happens because the initial configuration generated by the backpropagation algorithm in the SAMANN network is not a perfect PCA map due to the local minimum problem. These results are not surprising because the SAMANN network is designed to minimize the Sammon's stress, while others are not. The PCA network performs the second best on all the eight data sets (except the *Curve2* data on which it is the best). This is largely because the PCA network performs a linear orthogonal projection which maximally retains the variance of the data. The LDA network is comparable to the PCA network on the *Gauss2*, *Sphere2* and *Random* data sets because for these data sets, the discriminant vectors are similar to the principal vectors (see Figs. 7(a)–(b), 9(a)–(b), and 10(a)–(b)). For other data sets, the LDA network generates large values of Sammon's stress because the discriminant vectors are not orthogonal. The NDA network produces large values of Sammon's stress due to the fact that it reorganizes patterns and groups them by categories such that they can be easily separated, resulting in distortions in the interpattern distances. Some values are extremely large due to the different

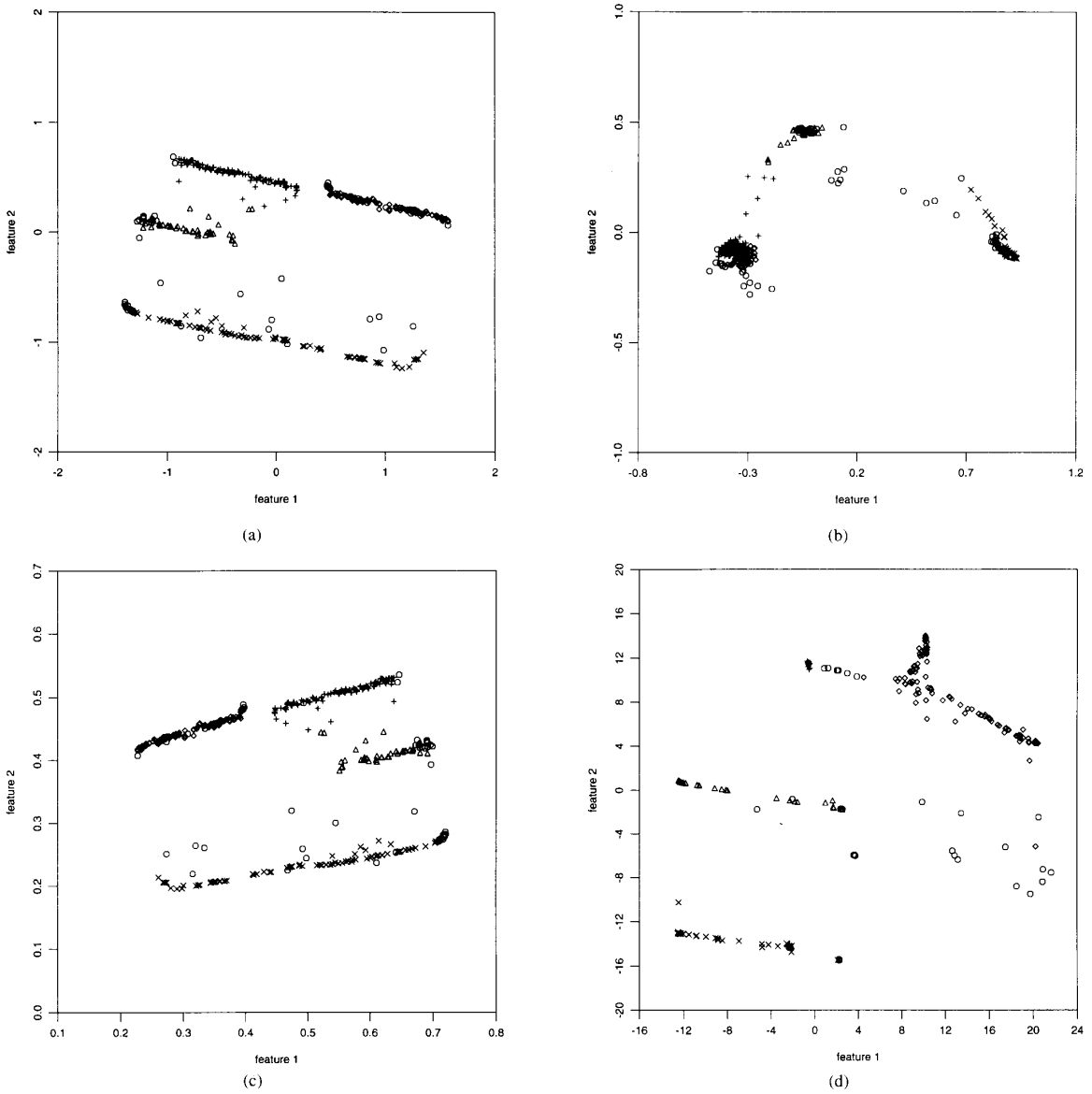


Fig. 13. 2D projection maps of data set *Range* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network.

TABLE III
THE AVERAGE VALUES OF SAMMON'S STRESS FOR THE PCA, LDA, SAMANN, NDA AND NP-SOM NETWORKS ON EIGHT DATA SETS

Data set		1	2	3	4	5	6	7	8
Sammon's stress, E	PCA	0.151	0.003	0.062	0.361	0.007	0.141	0.009	0.147
	LDA	0.161	0.169	0.077	0.386	0.134	0.289	0.408	0.519
	NDA	1.400	53.78	116.1	335.5	27.15	0.224	146.7	1.513
	SAMANN	0.115	0.005	0.052	0.231	0.007	0.084	0.008	0.084
	NP-SOM	0.346	0.005	0.299	0.954	0.034	0.493	0.069	0.420

ranges of the input features and output features. The average performance of the NP-SOM network is similar to that of the LDA network.

Table IV lists the average values of the nearest-neighbor classification error rate P_e^{NN} and relative ratio R_e^{NN} for the five networks on the eight data sets. The NDA network achieves the best performance in terms of the nearest-neighbor classification error rate among all the networks over all the eight data sets with the exception of *Range* and *Texture* on which the NP-SOM network performs slightly better. The values of P_e^{NN} on the projected data is comparable to those on the data in the original space (the values of R_e^{NN} are close to 1.0). These results are consistent with the known fact that feedforward network classifiers and nearest-neighbor classifiers have comparable performances [13]. The NP-SOM network also achieves comparable performance with

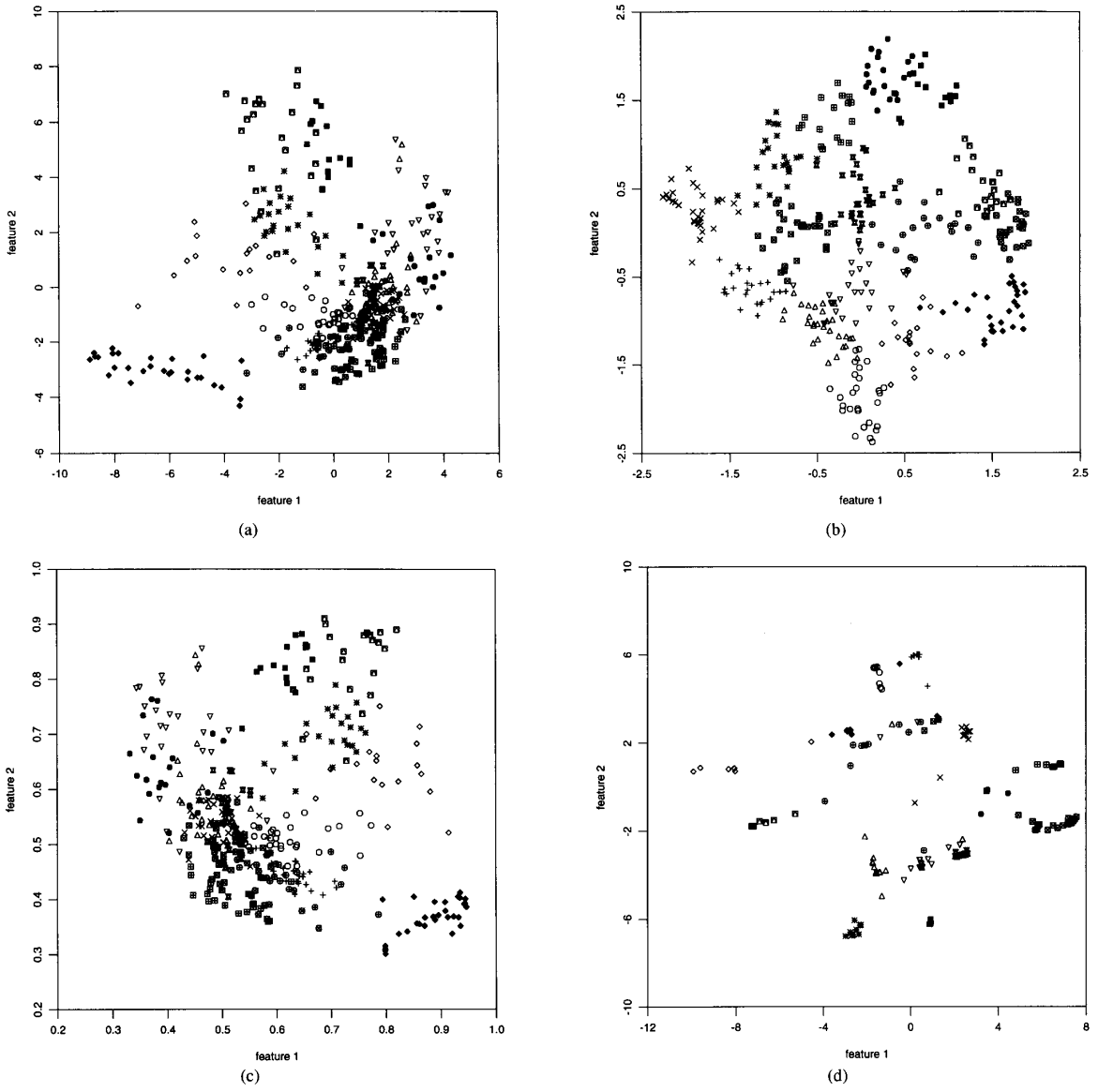


Fig. 14. 2D projection maps of data set *Texture* using the four networks. (a) PCA network, (b) LDA network, (c) SAMANN network, and (d) NDA network.

the nearest-neighbor classifier on the original data. This is because in the NP-SOM network, the nearest-neighbor classification is actually done in the quantized feature space with the same dimensionality as the input space rather than in the projected space. The PCA and LDA networks perform worse than the NDA and NP-SOM networks on most of the data sets because they are driven by the global properties of the data, not by the nearest neighbor's category information. Although the SAMANN network is designed to maximally preserve the interpattern distances, the nearest neighbor classification is not good for some projected data sets obtained by this method since it makes no use of the category information.

The average values of the minimum-distance classification error rate P_e^{MD} and relative ratio R_e^{MD} for the five networks on the eight data sets are shown in Table V. Comparing the PCA network and the LDA network (both are linear networks), the LDA network performs better than the PCA network over all the eight data sets. For the *IRIS*, *8OX* and *Texture* data sets, which are all real data sets, the LDA network achieves significant improvements over the PCA network. Note that the nearest mean classifier is a linear classifier based on global structure of the data set. The LDA network can find the best two features which have the most linear discriminant power. The NDA algorithm is comparable to the LDA for the linearly-separable data sets (*Gauss2*, *IRIS*,

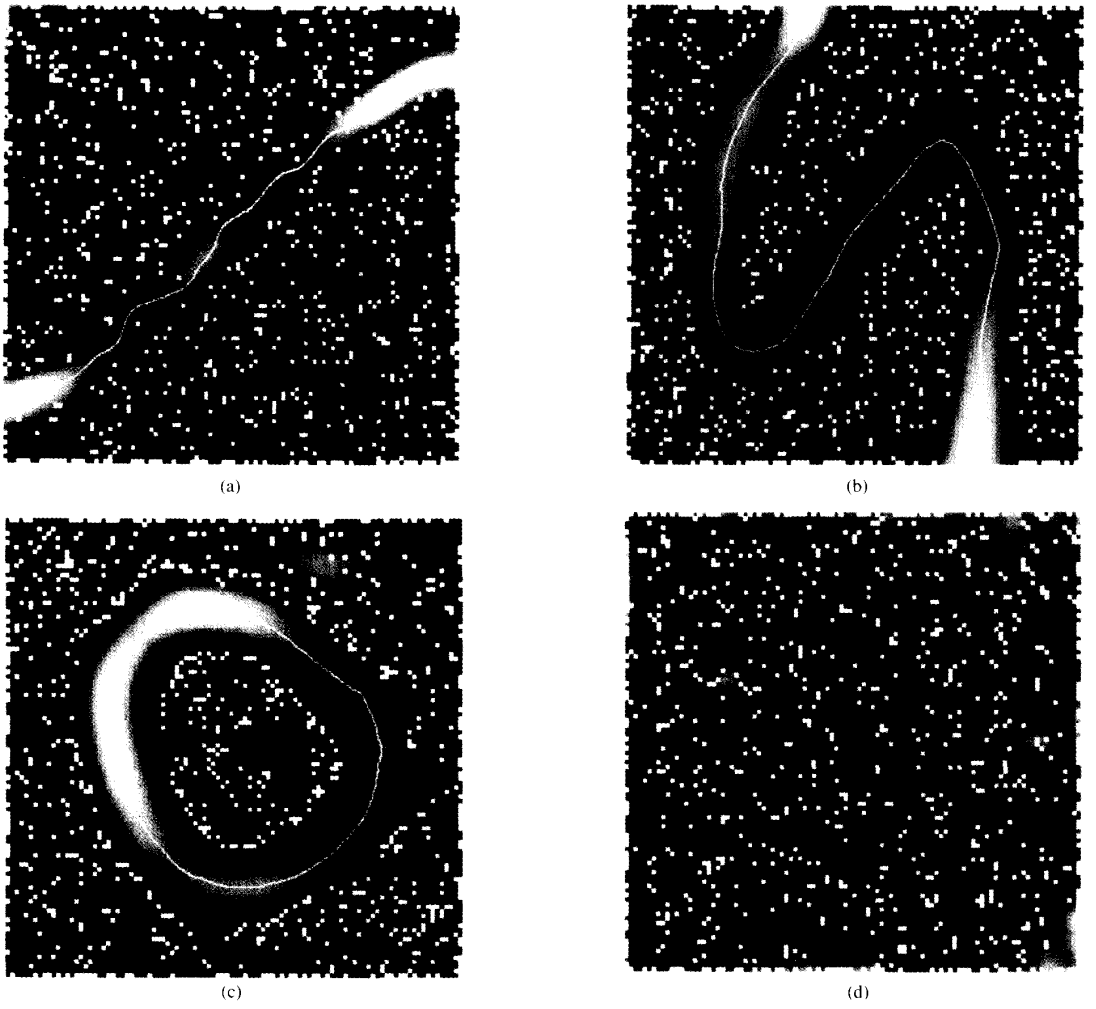


Fig. 15. The distance images of Kohonen's SOM superimposed by the 2D projection maps and boundaries between classes for the eight data sets using the NP-SOM network. (a) *Gauss2*. (b) *Curve2*. (c) *Sphere2*. and (d) *Random*.

Range, and *Texture*), but much better for linearly nonseparable data sets (*Curves*, *Sphere2*, and *8OX*). In general, the NDA network performs the best on most of the data sets. The performances of the PCA, SAMANN and NP-SOM networks are comparable.

Both the PCA and SAMANN networks perform very badly on the *Texture* data in terms of the nearest neighbor and minimum distance classification error rates. This is because the linear dimensionality of the *Texture* data is 15, which is much larger than the number of extracted features (dimensionality of the projected map). However, other networks still perform reasonably well on this data set due to their nonlinearity and the use of category information. We should mention an interesting phenomenon which is that sometimes the nearest neighbor classifier and minimum distance classifier perform better on projected data than on the original data ($R_e < 1.0$), particularly for the *IRIS* and *8OX* data sets in which the number of patterns is relatively small. This phenomenon may be explained by the fact that the effect of "curse of dimensionality" [16] can be reduced by feature extraction or data projection, which is exhibited using the entire data set (not just the training data) in our experiments (but classification errors are evaluated using the "leave-one-out" technique).

IV. Conclusions

We have designed the following neural networks for projecting multivariate pattern vectors: linear discriminant analysis (LDA) network, SAMANN network for Sammon's projection, nonlinear projection based on Kohonen's SOM (NP-SOM), and nonlinear discriminant analysis network (NDA) using a feedforward network classifier. A common attribute of these networks is that they all employ adaptive learning algorithms which makes them suitable in some environments where the distribution of patterns in feature space changes with respect to time. The availability of these networks also facilitates hardware implementation of well-known classical feature ex-

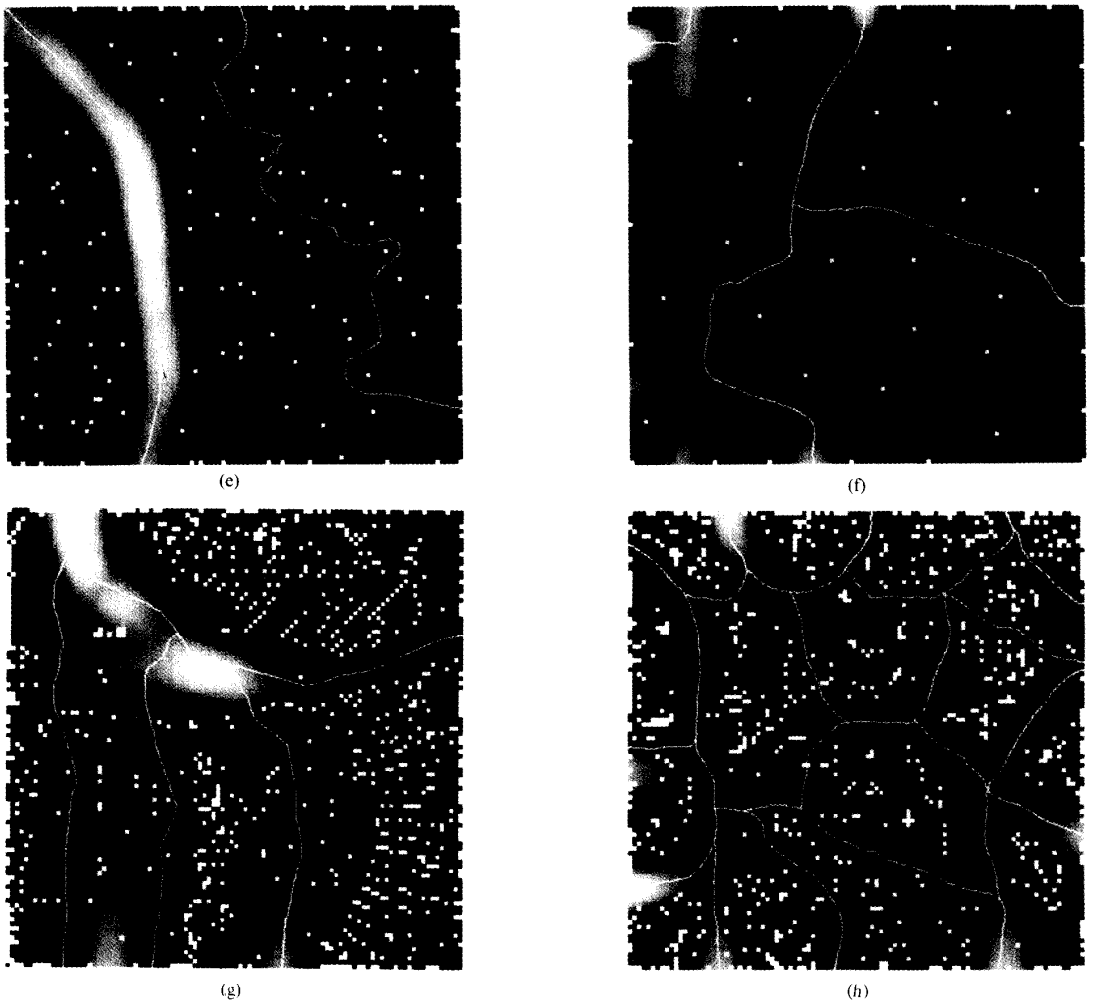


Fig. 15. continued. (e) *IRIS*, (f) *8OX*, (g) *Range*, and (h) *Texture*.

traction and projection approaches. Moreover, the SAMANN network offers the generalization ability of projecting new data, which is not present in the original Sammon's projection algorithm; the NDA method and NP-SOM network provide new powerful approaches for visualizing high dimensional data.

The performances of the five networks (PCA, LDA, SAMANN, NDA and NP-SOM) for feature extraction and data projection have been evaluated based on visual inspection of the projection maps and on three numerical criteria on eight data sets. Based on our experimental results and analysis, we draw the following general conclusions: i) the NP-SOM network has a good performance for data visualization. It reveals the cluster tendency in the multivariate data very well due to the role of the distance image; ii) the SAMANN and PCA networks preserve the data structure, cluster shape, and interpattern distances better than the LDA, NDA and NP-SOM networks; iii) the NDA network is superior to all the other networks for classification purposes, especially when the data are not linearly-separable, but it severely distorts the structure of the data and the interpattern distances; iv) there is no general conclusion on whether the PCA or the LDA is better for preserving the nearest-neighbor category information; v) the LDA network outperforms the PCA, SAMANN and NP-SOM networks in the sense of nearest mean classification error; vi) feature extraction can help to reduce or eliminate the "curse of dimensionality;" vii) it is often necessary to use more than one method (network) in order to reveal various properties of multivariate data; and viii) knowledge of the structure of the data set obtained from the projection maps can guide in choosing a proper classification and clustering tools.

Our simulation results are consistent with the high-level analysis based on our knowledge of the traditional feature extraction and data projection approaches. That is one of the main advantage of studying the links between neural networks and traditional pattern recognition and data analysis approaches.

TABLE IV
THE AVERAGE VALUES OF THE NEAREST NEIGHBOR CLASSIFICATION ERROR RATE P_e^{NN} AND RELATIVE RATIO R_e^{NN} FOR THE PCA, LDA, SAMANN, NDA AND NP-SOM NETWORKS ON EIGHT DATA SETS

Data set		1	2	3	4	5	6	7	8
P_e^{NN}	Original	0.002	0.000	0.000	0.521	0.040	0.044	0.041	0.036
	PCA	0.005	0.000	0.007	0.479	0.040	0.216	0.083	0.405
	LDA	0.003	0.037	0.009	0.501	0.037	0.067	0.126	0.101
	NDA	0.000	0.000	0.000	0.355	0.035	0.002	0.078	0.080
	SAMANN	0.004	0.000	0.005	0.488	0.051	0.200	0.084	0.405
	NP-SOM	0.003	0.000	0.000	0.507	0.041	0.049	0.073	0.046
R_e^{NN}	PCA	1.003	1.000	1.007	0.972	1.000	1.165	1.040	1.356
	LDA	1.001	1.037	1.009	0.987	0.997	1.022	1.081	1.063
	NDA	0.998	1.000	1.000	0.891	0.995	0.960	1.036	1.042
	SAMANN	1.002	1.000	1.005	0.978	1.011	1.149	1.041	1.356
	NP-SOM	1.001	1.000	1.000	0.991	1.001	1.005	1.031	1.010

TABLE V
THE AVERAGE VALUES OF THE MINIMUM DISTANCE CLASSIFICATION ERROR RATE P_e^{MD} AND RELATIVE RATIO R_e^{MD} FOR THE PCA, LDA, SAMANN, NDAAND NP-SOM NETWORKS ON EIGHT DATA SETS

Data set		1	2	3	4	5	6	7	8
P_e^{MD}	Original	0.001	0.133	0.390	0.483	0.080	0.044	0.152	0.065
	PCA	0.001	0.133	0.391	0.458	0.100	0.164	0.187	0.420
	LDA	0.001	0.106	0.387	0.455	0.020	0.022	0.166	0.112
	NDA	0.001	0.006	0.072	0.487	0.058	0.007	0.117	0.100
	SAMANN	0.001	0.136	0.409	0.458	0.093	0.178	0.186	0.413
	NP-SOM	0.031	0.135	0.474	0.506	0.066	0.129	0.158	0.088
R_e^{MD}	PCA	1.000	1.000	1.001	0.983	1.019	1.115	1.030	1.333
	LDA	1.000	0.976	0.998	0.981	0.944	0.979	1.012	1.044
	NDA	1.000	0.888	0.771	1.003	0.980	0.965	0.970	1.033
	SAMANN	1.000	1.003	1.014	0.983	1.012	1.128	1.030	1.328
	NP-SOM	1.030	1.002	1.060	1.016	0.987	1.081	1.005	1.022

REFERENCES

[1] H. M. Abbas and M. M. Fahmy, "A neural model for adaptive Karhunen Loeve Transform (KLT)," in *Proc. IEEE Int. Joint Conf. on Neural Networks*, Baltimore, MD, June 1992, vol. 2, pp. 975–980.

[2] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Trans. Neural Networks*, vol. 2, pp. 53–58, 1989.

[3] A. R. Barron and R. L. Barron, "Statistical learning networks: A unifying view," in *Proc. 10th Symposium Interface*, E.G. Wegman, Ed., American Statistical Association, Washington, D.C., 1988, pp. 192–203.

[4] G. Biswas, A. K. Jain, and R. C. Dubes, "Evaluation of projection algorithms," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-3, no. 6, pp. 701–708, Nov. 1981.

[5] P. Foldiak, "Adaptive network for optimal linear feature extraction," in *Proc. IEEE Int. Joint Conf. on Neural Networks*, Washington, D.C., 1989, vol. 1, pp. 401–405.

[6] D. H. Foley and J. W. Sammon, "An optimal set of discriminant vectors," *IEEE Trans. Comput.*, vol. C-24, no. 3, pp. 281–289, Mar. 1975.

[7] J. H. Friedman and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis," *IEEE Trans. Comput.*, vol. C-23, pp. 881–890, Sept. 1974.

[8] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Second Edition. New York: Academic Press, 1990.

[9] P. Gallinari, S. Thiria, F. Badran, and F. Fogelman-Soulie, "On the relations between discriminant analysis and multilayer perceptrons," *Neural Networks*, vol. 4, pp. 349–360, 1991.

[10] D. O. Hebb, *The Organization of Behavior*. New York: Wiley, 1949.

[11] R. L. Hoffman and A. K. Jain, "Segmentation and Classification of Range Images," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 608–620, 1987.

[12] K. Hornik and C.-M. Kuan, "Convergence analysis of local feature extraction algorithm," *Neural Networks*, vol. 5, pp. 229–240, 1992.

[13] W. Y. Huang and R. P. Lippmann, Comparisons between neural net and traditional classifiers," in *IEEE 1st Int. Conf. Neural Networks*, San Diego, CA, June 1987, pp. IV-485–IV-493.

[14] P. J. Huber, "Projection pursuit," *Ann. Statist.*, vol. 13, pp. 435–475, 1985.

[15] J.-N. Hwang, S.-R. Lay, and A. Lippman, "Unsupervised learning for multivariate probability density estimation: Radial basis function and exploratory projection pursuit," in *1993 IEEE Int. Conf. Neural Networks*, San Francisco, CA, Mar. 28–Apr. 1, 1993, vol. III, pp. 1486–1491.

[16] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Handbook of Statistics*, P. R. Krishnaiah and L. N. Kanal, Eds. New York: North Holland, vol. 2, 1982, pp. 835–855.

[17] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[18] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[19] A. K. Jain and J. Mao, "Artificial neural network for nonlinear projection of multivariate data," in *Proc. IEEE Int. Joint Conf. Neural Networks*, Baltimore, MD, June 1992, vol. 3, pp. 335–340.

[20] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, pp. 59–69, 1982.

[21] T. Kohonen, *Self Organization and Associative Memory, 3rd Ed.* New York: Springer-Verlag, 1989.

[22] T. Kohonen, "Self-organized network," *Proc. IEEE*, vol. 43, 1990, pp. 59–69.

[23] M. A. Kraaijveld, J. Mao, and A. K. Jain, "A nonlinear projection method based on Kohonen's topology preserving maps," in *Proc. Int. Conf. Pattern Recognition*, The Hague, The Netherlands, 1992, vol. 2, pp. 41–45.

[24] A. Krogh and J. A. Hertz, "Hebbian learning of principal components," in *Parallel Processing in Neural Systems and Computers*, R. Eckmiller, G. Hartmann, and G. Hauske, Eds. New York: North-Holland, 1990, pp. 183–186.

[25] H. Kuhnel and P. Tavan, "A network for discriminant analysis," in *Artificial Neural Networks*, T. Kohonen, K. Makisara, O. Simula, and J. Kangas, Eds. New York: Elsevier, 1991.

[26] S. Kung and K. Diamantaras, "A neural network learning algorithm for adaptive principal component extraction (APEX)," in *IEEE Int. Conf. Acoustics, Speech,Signal Processing*, 1990, vol. 1, pp. 861–864.

[27] R. Linsker, "From basic network principles to neural architecture," in *Proc. Nat. Academy Sciences*, USA, vol. 83, 1986, pp. 7508–7512, 8390–8394, 8779–8783.

[28] J. Mao and A. K. Jain, "Discriminant analysis neural networks," in *Proc. IEEE Int. Conf. Neural Networks*, San Francisco, CA, Mar. 1993, vol. 1, pp. 300–305.

[29] J. Mao and A. K. Jain, "Regularization techniques in artificial neural networks," in *Proc. World Congress ral Networks*, Portland, OR, Jul. 1993, pp. IV-75–IV-79.

[30] E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biology*, vol. 15, pp. 267–273, 1982.

[31] E. Oja, "Neural networks, principal components, and subspaces," *Int. J. Neural Syst.*, vol. 1, pp. 61–68, 1989.

[32] T. Okada and S. Tomita, "An optimal orthonormal system for discriminant analysis," *Patt. Recognition*, vol. 18, pp. 139–144, 1985.

[33] J. Rubner and K. Schulten, "Development of feature detectors by self-organization," *Biol. Cybern.*, vol. 62, pp. 193–199, 1990.

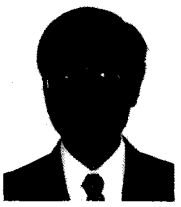
[34] J. Rubner and P. Tavan, "A self-organizing network for principal component analysis," *Europhysics Lett.*, vol. 10, pp. 693–698, 1989.

[35] J. W. Sammon, Jr., "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. C-18, pp. 401–409, 1969.

[36] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, vol. 2, pp. 459–473, 1989.

[37] A. R. Webb and D. Lowe, "The optimized internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks*, vol. 3, pp. 367–375, 1990.

[38] Y. Zhao and C. G. Atkeson, "Projection pursuit learning," in *Proc. 1991 IEEE Int. Joint Conf. Neural Networks*, Seattle, WA, Jul. 1991, vol. 1, pp. 869–874.



Jianchang Mao received the B.S. degree in physics in 1983 and the M.S. degree in electrical engineering in 1986, from East China Normal University, Shanghai, P.R. China. He completed the Ph.D. degree at the Department of Computer Science at Michigan State University in 1994.

During the summer of 1993, he worked at the Xerox Palo Alto Research Center on document image processing. Dr. Mao is now employed by IBM Almaden Research Center. His current research interests are pattern recognition, computer vision, neural networks, image processing, and parallel computing.

Anil K. Jain (S'70–M'72–SM'86–F'91) received the B.Tech. degree in 1969 from the Indian Institute of Technology, Kanpur, and the M.S. and Ph.D. degrees in electrical engineering from the Ohio State University, in 1970 and 1973, respectively.

He joined the faculty of Department of Computer Science at Michigan State University in 1974 and is a University Distinguished Professor. He served as Program Director of the Intelligent Systems Program at the National Science Foundation from 1980 to 1981, and has held visiting appointments at Delft University of Technology, The Netherlands, Norwegian Computing Center, Oslo and Tata Research Development and Design Center, Pune, India. He has also been a Consultant to several industrial, government, and international organizations. His current research interests are computer vision, image processing, artificial neural networks, and pattern recognition.

Dr. Jain has published papers on the following topics: statistical pattern recognition, artificial neural networks, exploratory pattern analysis, remote sensing, Markov random fields, texture analysis, interpretation of range images, and 3D object recognition. Several of his papers have been reprinted in edited volumes on image processing and pattern recognition. He received the best paper awards in 1987 and 1992 and certificates for outstanding contributions in 1976 and 1979 from the Pattern Recognition Society. He is the Editor-in-Chief of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and serves on the Editorial Boards of the *Pattern Recognition Journal, Pattern Recognition Letters, Journal of Intelligent Systems*, and *Journal of Mathematical Imaging and Vision*. He is the co-author of *Algorithms for Clustering Data*, (Prentice-Hall, 1988), has edited the book *Real-Time Object Measurement and Classification*, (Springer-Verlag, 1988), and co-edited the books, *Analysis and Interpretation of Range Images*, (Springer-Verlag, 1990), *Markov Random Fields: Theory and Applications*, (Academic Press, 1993), *Statistical Pattern Recognition and Artificial Neural Networks: Old and New Connections*, (North-Holland, 1991), and *Three-Dimensional Object Recognition Systems*, (Elsevier, 1993). He was the General Chairman of the IEEE Computer Society Workshop on Interpretation of 3D Scenes, Austin (1989), Co-Chairman of the Eleventh International Conference on Pattern Recognition, The Hague (1992), Program Director of the NATO Advanced Research Workshop on Realtime Object Measurement and Classification, Maratea (1987), and co-directed NSF supported Workshops on Challenges of Computer Vision: Future Research Directions, Maui (1991), Theory and Applications of Markov Random Fields, San Diego (1989) and Range Image Understanding, East Lansing (1988). Dr. Jain received the Distinguished Faculty Award from Michigan State University in 1989 and served as the Distinguished Visitor of the IEEE Computer Society during 1988-90.