Decision Boundary Feature Extraction for Neural Networks

Chulhee Lee, *Member, IEEE,* and David A. Landgrebe, *Fellow, IEEE*

***Abstract*— In this paper, we propose a new feature extraction method for feedforward neural networks. The method is based on the recently published decision boundary feature extraction algorithm which is based on the fact that all the necessary features for classification can be extracted from the decision boundary. The decision boundary feature extraction algorithm can take advantage of characteristics of neural networks which can solve complex problems with arbitrary decision boundaries without assuming underlying probability distribution functions of the data. To apply the decision boundary feature extraction method, we first give a specific definition for the decision boundary in a neural network. Then, we propose a procedure for extracting all the necessary features for classification from the decision boundary. Experiments show promising results.**

***Index Terms*— Neural networks, feature extraction, decision boundary, classification, decision boundary feature matrix, pattern recognition.**

I. Introduction

ALTHOUGH neural networks have been successfully applied in various fields [1]–[3], relatively few feature extraction algorithms are available for neural networks. In this paper, we propose a new feature extraction method for a feedforward neural network. A characteristic of neural networks is that they require extensive calculation in the training phase. Once a neural network is trained, the computational cost of the network is usually small compared with other nonparametric classifiers, such as the Parzen density estimator [4] and the kNN classifier [4]. However, the lack of efficient feature extraction methods inevitably would result in less efficient neural networks. The significance of this grows as more high-dimensional data and multisource data becomes available, resulting in neural networks which can become very complex.

Neural networks do not make any assumption about probability distribution functions of data and can solve complex problems with arbitrary decision boundaries. Thus, it is desirable that a feature extraction method for neural networks preserve this characteristic. In this paper, we apply the decision boundary feature extraction method to neural networks to accomplish this.

Manuscript received December 5, 1995; revised June 20, 1996. This work was supported in part by NASA under Grant NAGW-925.

C. Lee was with the School of Electrical and Computer Engineering, Purdue University, W. Lafayette, IN 47907-1285 USA. He is now with the Department of Electrical Engineering, Yonsei University, Seoul, Korea.

D. A. Landgrebe is with the School of Electrical and Computer Engineering, Purdue University, W. Lafayette, IN 47907-1285 USA.

Publisher Item Identifier S 1045-9227(97)00236-1.

II. Background and Previous Works

Feature extraction has been one of the most important topics in pattern recognition and has been studied by many authors [4]–[7]. In feature extraction, one seeks a new feature space which represents observations better for a given purpose. Numerous feature extraction algorithms have been proposed for parametric classification. One of the most widely used transforms for signal representation is the Karhunen–Loeve transformation (principal component analysis). The Karhunen–Loeve transformation is optimum for signal representation in the sense that it provides the smallest mean square error for a given number of features, and can be used as a general feature extraction. However, the features defined by the Karhunen–Loeve transformation are not optimum with regard to class separability.

In discriminant analysis [4], a within-class scatter matrix Σ_W and a between-class scatter matrix Σ_b are used to formulate a criterion function. A typical criterion is

$$J_1 = \mathrm{tr}(\Sigma_w^{-1}\Sigma_b)$$

where

$$\Sigma_w = \sum_i P(\omega_i)\Sigma_i \quad \text{(within-class scatter matrix)}$$

$$\Sigma_b = \sum_i P(\omega_i)(\boldsymbol{M}_i - \boldsymbol{M}_0)(\boldsymbol{M}_i - \boldsymbol{M}_0) \quad \text{(between-class scatter matrix)}$$

$$\boldsymbol{M}_0 = \sum_i P(\omega_i)\boldsymbol{M}_i.$$

Here $\boldsymbol{M}_i, \Sigma_i$, and $P(\omega_i)$ are the mean vector, the covariance matrix, and the prior probability of class ω_i, respectively. New feature vectors are selected to maximize the criterion. A problem with discriminant analysis is that it does not work for multimodal class problems, though neural networks are well suited for such problems.

A few authors have extended the discriminant analysis to nonparametric classifiers. Fukunaga proposed a nonparametric discriminant analysis method which is based on a nonparametric extension of commonly used scatter matrices [8]. Patrick proposed a nonparametric feature extraction process where a nonquadratic distance function, defined between classes, is used to define the best linear subspace [9]. However, these techniques are difficult to apply to neural networks.

Short and Fukunaga proposed a feature extraction algorithm using problem localization [10]. The problem is simplified by partitioning the distribution space into local subregions and

performing a linear estimation in each subregion. However, it would be difficult to apply the localization method to neural networks, since a neural network would need to be trained for each subregion.

On the other hand, using neural networks for feature extraction and data projection has been of great interest [11]–[13]. Mao and Jain presented extensive analysis in this area [14]. And various feature selection or extraction methods for neural networks have been proposed. The relationship between the discriminant analysis and neural networks has been studied [15], [16] and the nonlinear discriminant analysis has been adapted to neural networks [17]. Brill *et al*. proposed a genetic feature selection method for neural-network classifiers [18]. In their method, a genetic algorithm is used for feature selection in the context of neural networks. Some authors have applied the principal component analysis to neural networks. However, in general, principal component analysis is not an optimum feature extraction method for classification. On the other hand, pruning can also provide simpler, faster networks [19]–[22]. But, it is a completely different approach from feature extraction. In addition, the relationship between pruning and classification accuracy is not well established.

In this paper, we apply the recently published decision boundary feature extraction (DBFE) method to neural networks. The method is based directly upon the decision boundary, and can be used for parametric classification and nonparametric classification. Unlike other feature extraction algorithms, the DBFE method does not assume underlying probability distribution functions and can take advantage of the complex decision boundaries neural networks can define. By employing the DBFE method, it will be possible to reduce the complexity of the network substantially. As a result, if a network is to be used for a large size of test data, the DBFE method can reduce processing time significantly.

III. Decision Boundary Feature Extraction for Neural Networks

A. Decision Boundary Feature Extraction

In [23], Lee and Landgrebe showed that all necessary feature vectors for classification can be extracted from the decision boundary which divides the observation space into several regions. Referring to Fig. 1, a feature vector such as $\boldsymbol{V}_1$ is seen to be of no use for the purpose of classification since the classification result remains unchanged even though an observation moves along the direction of the vector (from $\boldsymbol{X}$ to $\boldsymbol{X}'$). In other words, projection onto feature vector $\boldsymbol{V}_1$ provides no information on class separation. Therefore, feature vector $\boldsymbol{V}_1$ can be eliminated without increasing classification error.

On the other hand, feature vector $\boldsymbol{V}_2$ is useful for the purpose of classification since the classification result can be changed as the observation moves along the direction of the vector (from $\boldsymbol{X}$ to $\boldsymbol{X}''$). That is, projection onto feature vector $\boldsymbol{V}_2$ provides information helpful for classification. Such feature vectors should be retained in order to maintain the same classification accuracy. It is found that such useful feature vectors have a component which is normal to the decision boundary at at least one point on the decision boundary and the useless feature vectors are orthogonal to the vector normal to decision boundary at every point on decision boundary.

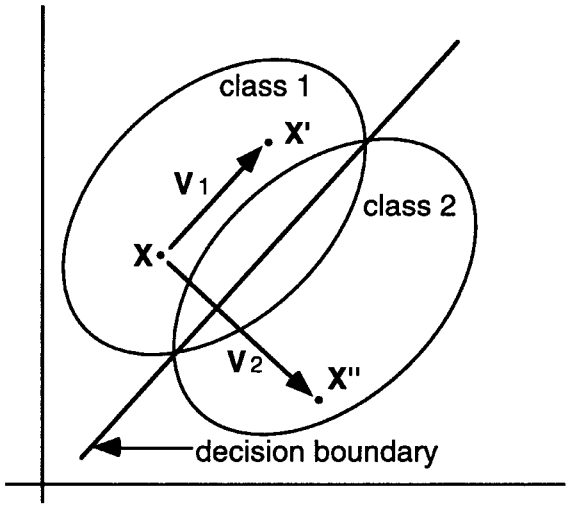

Fig. 1. Feature extraction from decision boundary.

In order to find useful feature vectors from the decision boundary, the decision boundary feature matrix was defined as follows:

Definition 1—The Decision Boundary Feature Matrix (DBFM): Let $\boldsymbol{N}(\boldsymbol{X})$ be the unit vector normal to the decision boundary at a point $\boldsymbol{X}$ on the decision boundary for a given pattern classification problem. Then the decision boundary feature matrix Σ_{DBFM} is defined as

$$\Sigma_{\mathrm{DBFM}} = \frac{1}{K}\int_S \boldsymbol{N}(\boldsymbol{X})\boldsymbol{N}^t(\boldsymbol{X})p(\boldsymbol{X})\,d\boldsymbol{X} \tag{1}$$

where $p(\boldsymbol{X})$ is a probability density function, $K = \int_S p(\boldsymbol{X})\,d\boldsymbol{X}, S$ is the decision boundary, and the integral is performed over the decision boundary.

From the decision boundary feature matrix, the following two theorems were derived [23].

Theorem 1: The rank of the decision boundary feature matrix Σ_{DBFM} of a pattern classification problem equals the smallest dimension whereby the same classification could be obtained as in the original space.

Theorem 2: The eigenvectors of the decision boundary feature matrix of a pattern recognition problem corresponding to nonzero eigenvalues are the necessary feature vectors to achieve the same classification accuracy as in the original space for the pattern recognition problem.

The theorems tell us that if only the eigenvectors corresponding to nonzero eigenvalues are retained, it is still possible to obtain the same classification accuracy as in the original space.

Further, although a classifier can define decision boundaries in the whole space, some portions of the decision boundaries are rarely utilized in actual classification. In Fig. 2, the decision boundaries displayed in bold are the most important portions in discriminating between classes, while the rest of the decision boundaries displayed in plain line play little

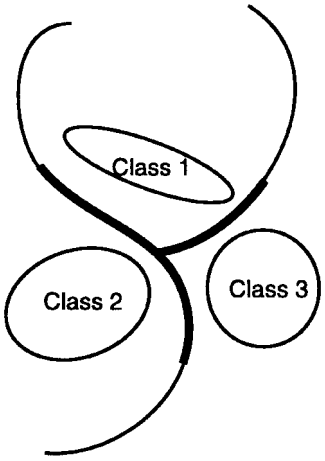

Fig. 2. Effective decision boundary.

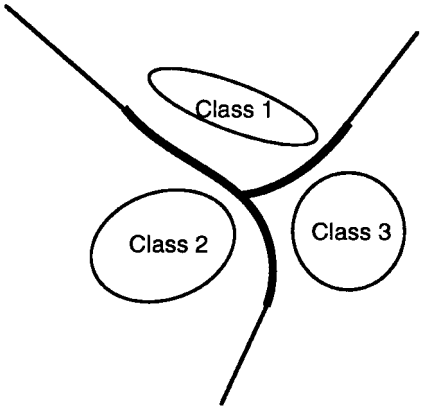

Fig. 3. New decision boundary extended by the effective decision boundaries.

role. We call the significant decision boundary "effective decision boundary" and by concentrating on the effective decision boundary, the feature extraction algorithm can be more efficient. Even though we disregard parts of the decision boundaries in our feature extraction algorithm, it does not mean that there are no decision boundaries in the regions where decision boundaries are disregarded. There will be new decision boundaries extended by the effective decision boundaries (Fig. 3).

The decision boundary feature extraction method has been successfully applied to parametric classifiers and nonparametric classifiers [24].[1] The method was shown to be robust and effective even when there are no class mean differences or no class covariance differences. In the case of nonparametric classifiers including the Parzen density estimator and neural networks, when outliers exist, the decision boundary feature matrix defined as in Definition 1 can be inappropriately influenced by insignificant portions of the decision boundary. Some outliers or noise may introduce a false decision boundary.

[1]This method for the parametric case has been implemented in a multispectral image data analysis system called MultiSpec which operates under the Macintosh operating system. Information about MultiSpec can be obtained by email to landgreb@ecn.purdue.edu. It can also be run under PC-Windows and two common versions of UNIX. See http://dynamo.ecn.purdue.edu/~biehl/MultiSpec/

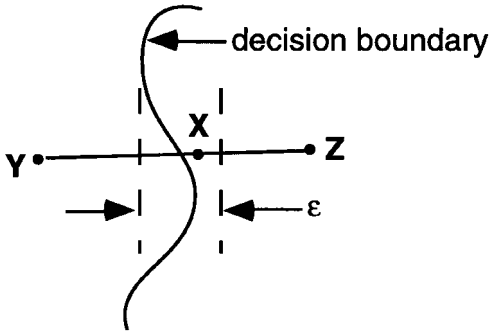

Fig. 4. Finding a decision boundary numerically.

Thus, if it is possible to remove such outliers, it will enhance the performance of the DBFE algorithm. This outlier problem is discussed in detail in [25].

B. Decision Boundaries in Neural Networks

In this paper, we consider feedforward neural networks, with backpropagation used to train the network. The decision rule is to select the class corresponding to the output neuron with the largest output [26].

We assume the number of output neurons is two (a two-class problem). The concept can be easily extended to multiclass problems which will be considered later. In order to utilize the decision boundary feature extraction algorithm for neural networks, the decision boundary must first be precisely defined. We define the decision boundary in a multilayer feedforward neural network as follows.

Definition 2: The decision boundary between two classes in a feedforward neural network is the locus of points where both output neurons have equal activations.

If we denote the activation of output neuron i as $\mathrm{OUT}_i(\boldsymbol{X})$ where $\boldsymbol{X}$ is an input vector and let $h(\boldsymbol{X}) = \mathrm{OUT}_1(\boldsymbol{X}) - \mathrm{OUT}_2(\boldsymbol{X})$, then the decision boundary can be defined as

$$\{\boldsymbol{X} | h(\boldsymbol{X}) = 0\}. \tag{2}$$

Since the decision boundary in a neural network may not be solved for analytically, the decision boundary must be found numerically. Assuming $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are classified differently, the line connecting $\boldsymbol{Y}$ and $\boldsymbol{Z}$ must pass through decision boundary. By moving along the line, we can find a point $\boldsymbol{X}$ in the vicinity of the decision boundary within any bound ε, as seen in Fig. 4.

Once the point $\boldsymbol{X}$ on the decision boundary is found, the normal vector to the decision boundary at $\boldsymbol{X}$ is given by

$$\nabla h(\boldsymbol{X}) = \frac{\partial h}{\partial x_1}\boldsymbol{x}_1 + \frac{\partial h}{\partial x_2}\boldsymbol{x}_2 + , \cdots , + \frac{\partial h}{\partial x_n}\boldsymbol{x}_n. \tag{3}$$

In the case of two-layer neural networks (input layer, hidden layer, and output layer), $\mathrm{OUT}_1(\boldsymbol{X})$ and $\mathrm{OUT}_2(\boldsymbol{X})$ are given by

$$\mathrm{OUT}_1(\boldsymbol{X}) = F(\boldsymbol{W}_h^1 F(\boldsymbol{W}_i \boldsymbol{X}))$$
$$\mathrm{OUT}_2(\boldsymbol{X}) = F(\boldsymbol{W}_h^2 F(\boldsymbol{W}_i \boldsymbol{X}))$$

where $\boldsymbol{W}_i$ is a input weight matrix, $\boldsymbol{W}_h^1$ and $\boldsymbol{W}_h^2$ are weight vectors, and F is an activation function. Although it is possible to calculate the gradient of $h(\boldsymbol{X}) = O\mathrm{UT}1_(\boldsymbol{X}) - \mathrm{OUT}_2(\boldsymbol{X})$

analytically, it can be very complex, as there can be more hidden layers. As an alternative, the term $\nabla h(\boldsymbol{X})$ can be calculated numerically as follows:

$$\nabla h(\boldsymbol{X}) \approx \frac{\Delta H}{\Delta x_1}\boldsymbol{x}_1 + \frac{\Delta h}{\Delta x_2}\boldsymbol{x}_2 +, \cdots, + \frac{\Delta h}{\Delta x_n}\boldsymbol{x}_n. \quad (4)$$

In some neural networks, thresholding is applied to the outputs and neural networks may reject some samples. In such cases, $h(\boldsymbol{X})$ in (2) may not be well defined. In that case, one might find a vector along which the classification result changes most rapidly. For example, let $\boldsymbol{X}$ be a point on the decision boundary. Then find the smallest Δx_i such that the classification result of $\boldsymbol{X} + \Delta x_i \boldsymbol{x}_i$ is different from that of $\boldsymbol{X}$. We may then estimate a unit vector $\boldsymbol{N}$ along which the classification result changes most rapidly as follows:

$$\boldsymbol{N} = V/|V| \quad \text{where}$$
$$V \approx \frac{1}{\Delta x_1}\boldsymbol{x}_1 + \frac{1}{\Delta x_2}\boldsymbol{x}_2 +, \cdots, + \frac{1}{\Delta x_n}\boldsymbol{x}_n. \quad (5)$$

C. Decision Boundary Feature Extraction Procedure for Neural Networks

We thus propose the following procedure for feedforward neural networks utilizing the decision boundary feature extraction algorithm.

Decision Boundary Feature Extraction Procedure for Neural Networks (Two-Pattern Class Case)

Step 1) Train the neural network using all features.

Step 2) For each training sample correctly classified as class ω_1, find the nearest sample correctly classified as class ω_2. Repeat the same procedure for the samples classified as class ω_2.

Step 3) The line connecting a pair of samples found in STEP 2 must pass through the decision boundary since the pair of samples are correctly classified differently. By moving along this line, find the point on the decision boundary or near the decision boundary within a threshold.

Step 4) At each point found in Step 3), estimate the normal vector $\boldsymbol{N}_i$ by

$$\boldsymbol{N}_i = \frac{\nabla h(\boldsymbol{X})}{|\nabla h(\boldsymbol{X})|}$$

where

$$\nabla h(\boldsymbol{X}) \approx \frac{\Delta h}{\Delta x_1}\boldsymbol{x}_1 + \frac{\Delta h}{\Delta x_2}\boldsymbol{x}_2 +, \cdots, + \frac{\Delta h}{\Delta x_n}\boldsymbol{x}_n$$
$$h(\boldsymbol{X}) = \mathrm{OUT}_1(\boldsymbol{X}) - \mathrm{OUT}_2(\boldsymbol{X}) \ \{\text{see (2)}\}.$$

Step 5) Estimate the decision boundary feature matrix using the normal vectors found in Step 4) by

$$\Sigma_{\mathrm{DBFM}} = \frac{1}{L}\sum_i \boldsymbol{N}_i \boldsymbol{N}_i^t$$

where L is the number of samples correctly classified.[2]

[2] In [23], a modified version of ΣDBFM, called ΣEDBFM, the effective decision boundary feature matrix, which is based upon only the effective portion of the decision boundary, is defined and may be used at this point to improve the performance of the algorithm somewhat.

Step 6) Select the eigenvectors of the decision boundary feature matrix as new feature vectors according to the magnitude of the corresponding eigenvalues.

Since it is necessary to calculate new feature vectors along these eigenvectors, additional calculation is introduced. However, if a portion of the eigenvectors are found insignificant in discriminating between classes, it is still possible to reduce the overall calculation for test samples substantially. For example, the number of multiplications needed to classify a test sample using a two-layer feedforward neural network which has 20 inputs, 60 hidden neurons (assuming that the number of hidden neurons is three times the number of inputs), and ten output neurons is given by

$$20 * 60 + 60 * 10 = 1800.$$

The total number of neurons is $70(= 60 + 10)$. Assuming it is possible to obtain about the same performance with ten features selected by the DBFE method, the number of multiplications needed to classify a test sample can be reduced to

$$20 * 10 + 10 * 30 + 30 * 10 = 800.$$

The first $200(= 20 * 10)$ multiplications are needed to calculated the ten features from the original 20-dimensional data, and the total number of neurons is $50(= 10 + 30 + 10)$. Although one needs to train the networks twice, if the application is intended for a large size of test samples or repeated use, the DBFE algorithm can reduce processing time substantially. In addition, the second phase training time is generally much smaller than the first one.

If there are more than two classes, the procedure can be repeated for each pair of classes after the network is trained for all classes. Then the total decision boundary feature matrix can be calculated by averaging the decision boundary feature matrices of each pair of classes. If prior probabilities are available, the summation can be weighted. That is, if there are M classes, the total decision boundary feature matrix can be calculated as

$$\Sigma_{\mathrm{DBFM}} = \sum_{i}^{M}\sum_{j,j\neq i}^{M} P(\omega_i)P(\omega_j)\Sigma_{\mathrm{DBFM}}^{ij}$$

where $\Sigma_{\mathrm{DBFM}}^{ij}$ is a decision boundary feature matrix between class ω_i and class ω_j and $P(\omega_i)$ is the prior probability of class ω_i if available. Otherwise let $P(\omega_i) = 1/M$.

IV. Experiments

A. Experiments with Generated Data

In the following example, there are three classes. Class ω_1 is normal with the following statistics:

$$\boldsymbol{M}_1 = \begin{bmatrix}0\\0\\0\end{bmatrix} \quad \Sigma_1 = \begin{bmatrix}4 & 0 & 0\\0 & 4 & 0\\0 & 0 & 9\end{bmatrix}.$$

Class ω_2 is equally divided between two normal distributions with the following statistics:

$$\boldsymbol{M}_2^1 = \begin{bmatrix} 5 \\ 0 \\ 0 \end{bmatrix} \quad \Sigma_2^1 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 9 \end{bmatrix} \quad \text{and}$$

$$\boldsymbol{M}_2^2 = \begin{bmatrix} -5 \\ 0 \\ 0 \end{bmatrix} \quad \Sigma_2^2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 9 \end{bmatrix}.$$

Class ω_3 is equally divided between two normal distributions with the following statistics:

$$\boldsymbol{M}_3^1 = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad \Sigma_3^1 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 9 \end{bmatrix} \quad \text{and}$$

$$\boldsymbol{M}_3^2 = \begin{bmatrix} 0 \\ -5 \\ 0 \end{bmatrix} \quad \Sigma_3^2 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 9 \end{bmatrix}.$$

As can be seen from the statistics, feature x_3 contributes nothing in discriminating between the classes. The distributions of these classes are shown in Fig. 5 in the $x_1 - x_2$ plane, along with the decision boundary found by the procedure. Eigenvalues λ_i and eigenvectors ϕ_i of Σ_{EDBFM} are calculated as follows:

$$\lambda_1 = 0.600, \quad \lambda_2 = 0.400, \quad \lambda_3 = 0.00015$$

$$\phi_1 = \begin{bmatrix} 0.06 \\ 1.00 \\ -0.02 \end{bmatrix}, \quad \phi_2 = \begin{bmatrix} -1.00 \\ 0.06 \\ 0.02 \end{bmatrix}, \quad \phi_3 = \begin{bmatrix} 0.02 \\ 0.02 \\ 1.00 \end{bmatrix}$$

$$\mathrm{Rank}(\Sigma_{\mathrm{EDBFM}}) \approx 2.$$

It can be said that the rank of Σ_{EDBFM} is two. Thus two features are needed to achieve the same classification accuracy as in the original space (Theorem 1).

In the actual test, 1000 samples were generated for each class. Five hundred samples were used for training and the rest for test. The experiment was repeated ten times with different data sets. Table I shows the classification accuracies along with the normalized classification accuracies obtained by dividing the classification accuracies by the classification accuracy obtained using the original three features. Standard deviations are also shown in Table I. As can be seen, the decision boundary feature extraction method finds the correct two features, achieving the same classification accuracy as can be obtained with the original three features.

B. Experiments with Real Data

Experiments were done using multispectral remote sensing data from the field spectrometer system (FSS), a helicopter-mounted field spectrometer [27]. Table II shows the major parameter values of FSS.

Along with the proposed algorithm, two other feature extraction algorithms, the Karhunen–Loeve transformation (principal component analysis), and discriminant analysis [4] were used to evaluate the performance of the proposed algorithm.

In the following test, there were three classes and each class had two subclasses. In other words, two subclasses were combined to form a new class. By purposely combining data from different classes, the data are made to be multimodal. The number of samples of the three classes are 1209, 1146, and 1103. Five hundred randomly selected samples from each class were used as training data and the rest were used for test. The test was repeated five times with different training and test data sets.

Figs. 6–7 show the performance comparison. First the original 60-dimensional data was reduced to 17-dimensional data using the band combination procedure. The classification accuracy with the 17-dimensional data was 94.6% with standard deviation 0.5%. At very low dimensionality $(N \leq 2)$, discriminant analysis showed the best performance. However, the classification accuracy was much smaller than the maximum possible classification accuracy, and the comparison is thus not relevant. When more than two features were used, the

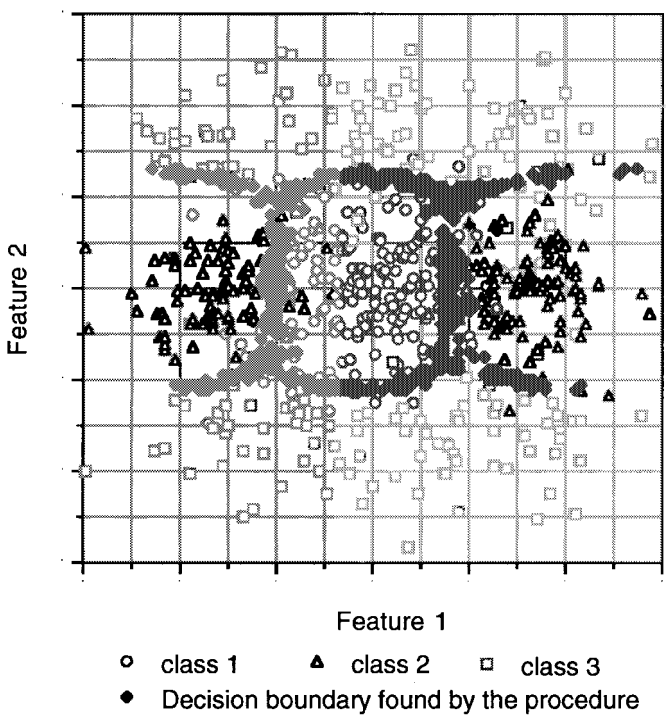

Fig. 5. Data distribution and the decision boundary found by the proposed procedure.

TABLE I
CLASSIFICATION ACCURACIES USING THE DECISION BOUNDARY FEATURE EXTRACTION

No. Features	Classification Accuracies	Normalized Classification Accuracies
1	63.7% (σ =2.47%)	74.4% (σ =2.80%)
2	85.7% (σ =0.64%)	100.0% (σ =0.22%)
3	85.7% (σ =0.58%)	100.0% (σ =0.26%)

TABLE II
PARAMETERS OF FIELD SPECTROMETER SYSTEM

Number of Bands	60
Spectral Coverage	0.4 - 2.4 μm
Altitude	60 m
IFOV(ground)	25 m

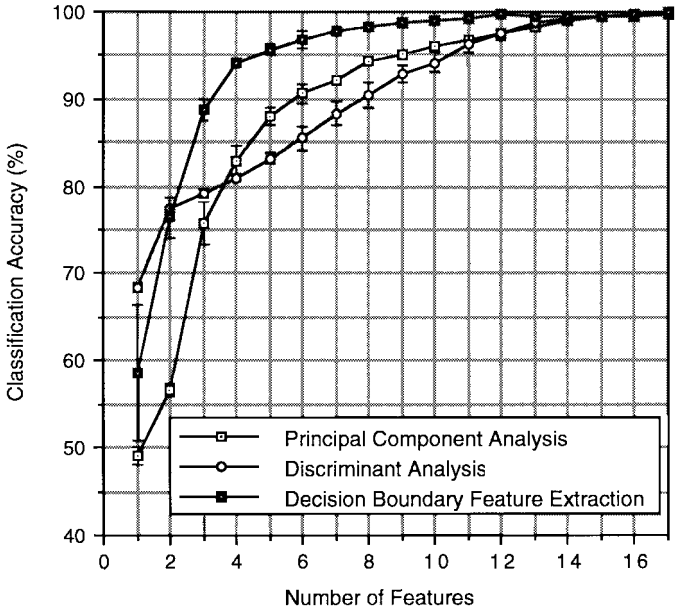

Fig. 6. Performance comparison (training data).

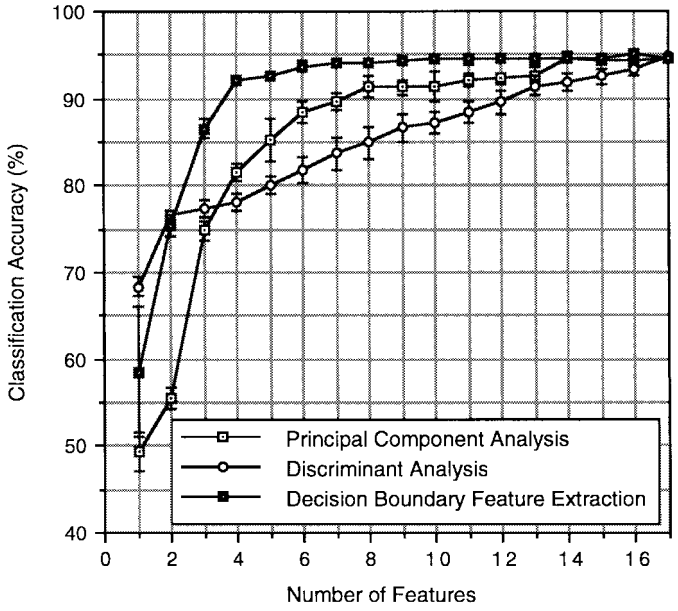

Fig. 7. Performance comparison (test data).

decision boundary feature extraction method outperformed all other methods. The decision boundary feature extraction method achieved about the same classification accuracy for test data with just four–six features as could be obtained in the original 17-dimensional space. In particular, with four features, the classification accuracy of the decision boundary feature extraction method for test data was 92.2% ($\sigma = 0.5\%$) while the classification accuracies of principal component analysis, and discriminant analysis were 81.6% ($\sigma = 1.0\%$), and 78.1% ($\sigma = 1.0\%$), respectively. With six features, the classification accuracies of test data were 93.7% ($\sigma = 0.6\%$), 88.4% ($\sigma = 1.2\%$), and 81.9% ($\sigma = 1.5\%$), respectively. Table III summarizes the results.

TABLE III
PERFORMANCE COMPARISON (PCA: PRINCIPAL COMPONENT ANALYSIS, DA: DISCRIMINANT ANALYSIS, DBFE: DECISION BOUNDARY FEATURE EXTRACTION)

Number of Features	PCA test Accuracy	DA test Accuracy	DBFE test Accuracy
4	81.6% (1.0)	78.1% (1.0)	92.2% (0.5)
6	88.4% (1.2)	81.9% (1.5)	93.7% (0.6)
17	94.6% (0.5)	94.8% (0.3)	94.6% (0.5)

Table IV shows the eigenvalues, proportions of the eigenvalues, accumulation of the eigenvalues, the classification accuracies, and the normalized classification accuracies obtained by dividing the classification accuracies by the classification accuracy obtained using the original 17 features. Fig. 8 shows the relationship between the accumulation of the eigenvalues and the normalized classification accuracies. Fig. 9 shows a magnified portion of Fig. 8. As can be seen, there is a strong linear relationship until the accumulation of eigenvalues reaches 90% (up to four features). With four features, 97.4% of the classification accuracy obtained with the original 17-dimensional data is achieved (normalized classification accuracy). After four features, adding more features results in marginal improvement in classification accuracies. With six features, the normalized classification accuracy is 99%. After six features, adding more features results in negligible improvement. From Table IV and Figs. 8 and 9, it can be said, roughly speaking, that eigenvectors corresponding to eigenvalues whose proportions are smaller than 1% make negligible contribution in improving classification accuracies.

It is noted that there is no direct linear relationship between the eigenvalues and the classification accuracy. And care needs to be taken in selecting eigenvectors since discarding eigenvectors corresponding to any nonzero eigenvalues may increase classification error. However, experiments suggest that eigenvectors corresponding to eigenvalues whose energy is less than 1% make negligible contribution in improving classification accuracies.

If a user needs to know the exact classification accuracy with a new feature set, new networks can be calculated for the new feature set and tested for the training data. However, this can be a time-consuming process. An alternative is to use other nonparametric classifiers such as the k nearest neighborhood (kNN) classifier [4] to estimate the classification accuracy of a new feature set. Unlike neural networks, the kNN classifier does not require a long training time and classification for a new feature set can be done very fast and efficiently, assuming that a new feature is added to the existing features.

The next experiment illustrates how the kNN classifier can be used to estimate the performance of a neural network. Four classes are generated using the standard data in [4] (dimension = 8, number of samples for each class = 1000, normal distribution). Then two classes are combined to form a new class and the final number of classes is two. A new feature set is obtained using the proposed DBFE algorithm. Table V shows proportion and accumulation of the eigenvalues. Fig. 10 shows the classification accuracies of neural networks and the kNN classifiers with the new features. Overall, the neural network shows better performance. However, both classifiers

TABLE IV
EIGENVALUES, PROPORTIONS OF THE EIGENVALUES, ACCUMULATION OF THE EIGENVALUES, CLASSIFICATION ACCURACIES, AND NORMALIZED CLASSIFICATION ACCURACIES. STANDARD DEVIATIONS ARE ALSO SHOWN IN PARENTHESIS

No. Features	Eigenvalue	Proportion of Eigenvalues(%)	Accumulation of Eigenvalues(%)	Classification Accuracies (%)	Normalized Classification Accuracies(%)
1	1080.1 (87.2)	36.8 (3.3)	36.8 (3.3)	58.5 (7.5)	61.8 (7.9)
2	781.8 (49.9)	26.6 (1.7)	63.4 (4.0)	75.4 (1.2)	79.6 (1.5)
3	448.0 (43.5)	15.2 (1.4)	78.6 (3.7)	86.5 (1.1)	91.4 (1.4)
4	319.5 (56.7)	10.9 (1.9)	89.5 (2.8)	92.2 (0.5)	97.4 (0.9)
5	106.4 (16.4)	3.6 (0.6)	93.1 (3.0)	92.7 (0.3)	97.9 (0.6)
6	64.4 (27.8)	2.2 (0.9)	95.3 (2.2)	93.7 (0.6)	99.0 (1.0)
7	41.8 (13.6)	1.4 (0.4)	96.7 (1.8)	94.0 (0.3)	99.4 (0.4)
8	25.8 (12.4)	0.9 (0.4)	97.6 (1.4)	94.1 (0.5)	99.5 (0.5)
9	20.9 (9.1)	0.7 (0.3)	98.3 (1.1)	94.4 (0.4)	99.7 (0.7)
10	15.7 (9.6)	0.5 (0.3)	98.8 (0.8)	94.7 (0.3)	100.1 (0.7)
11	9.8 (5.7)	0.3 (0.2)	99.2 (0.6)	94.5 (0.7)	99.9 (0.8)
12	7.9 (5.4)	0.3 (0.2)	99.4 (0.4)	94.7 (0.2)	100.1 (0.4)
13	5.6 (3.5)	0.2 (0.1)	99.6 (0.3)	94.5 (0.6)	99.9 (0.6)
14	4.4 (3.2)	0.1 (0.1)	99.7 (0.2)	94.7 (0.6)	100.1 (0.4)
15	3.4 (2.7)	0.1 (0.1)	99.9 (0.1)	94.7 (0.5)	100.1 (0.8)
16	2.4 (2.1)	0.1 (0.1)	100.0 (0.1)	95.0 (0.6)	100.4 (0.5)
17	1.7 (1.4)	0.0 (0.1)	100.0 (0.0)	94.6 (0.5)	100.0 (0.7)

TABLE V
PROPORTION AND ACCUMULATION OF EIGENVALUES (TEN REPETITIONS)

Proportion of eigenvalues (%)	Accumulation of eigenvalues (%)
85.11 (σ = 1.369)	85.1 (σ = 1.369)
11.45 (σ = 1.146)	96.6 (σ = 1.251)
3.10 (σ = 1.110)	99.7 (σ = 0.239)
0.27 (σ = 0.193)	99.9 (σ = 0.053)
0.05 (σ = 0.030)	100.0 (σ = 0.029)
0.02 (σ = 0.019	100.0 (σ = 0.012)
0.01 (σ = 0.012)	100.0 (σ = 0.003)
0.00 (σ = 0.003)	100.0 (σ = 0.000)

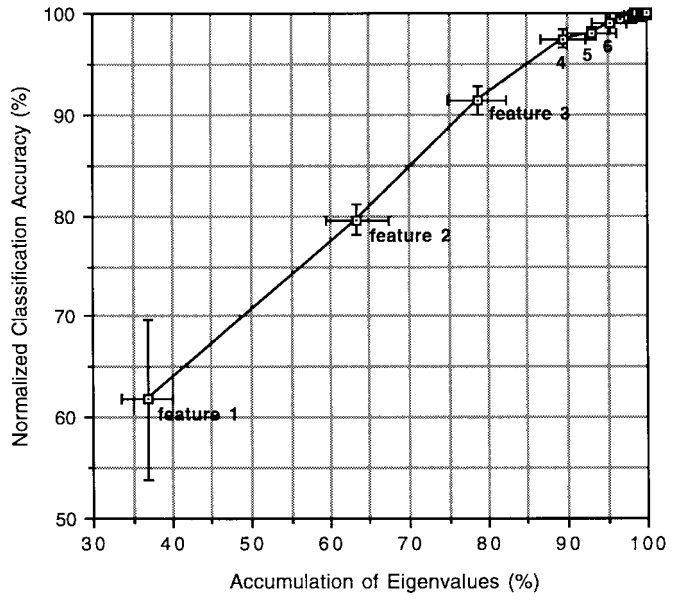

Fig. 8. Relationship between the accumulation of eigenvalues and normalized classification accuracies.

show similar performance characteristics almost reaching the maximum accuracy with three features. Fig. 11 shows the normalized classification accuracy ratio between neural network and the kNN classifier. As can be seen, the performance of

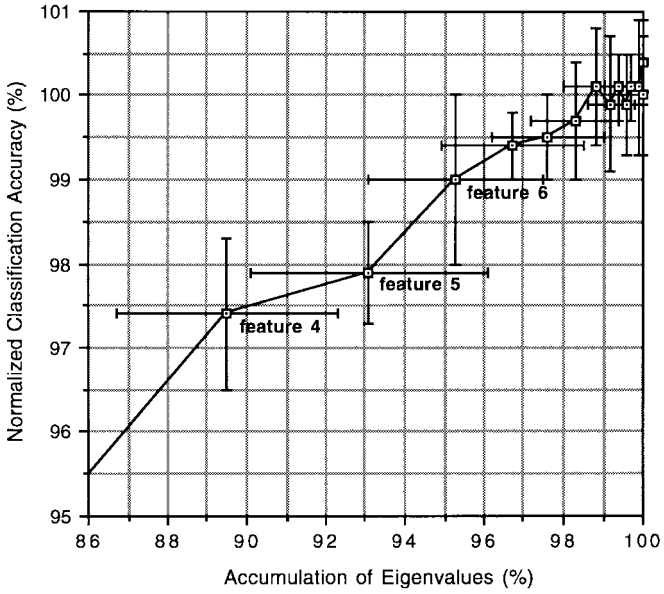

Fig. 9. Magnified portion of Fig. 8.

the kNN classifier can provide a good performance estimation of neural networks.

V. CONCLUSION

In this paper, we have extended the recently published decision boundary feature extraction algorithm to feedforward neural networks. We began by defining the decision boundary in a neural network. From the decision boundary, we estimated the normal vectors to the decision boundary, and calculated the decision boundary feature matrix. A new feature set was calculated from the decision boundary feature matrix. Then, a procedure for applying the decision boundary feature extraction algorithm to neural networks was presented. Experimental tests show promising performance.

Also discussed was the effectiveness of the eigenvectors of the decision boundary feature matrix. As expected from the theorem, it was observed that the effectiveness of the

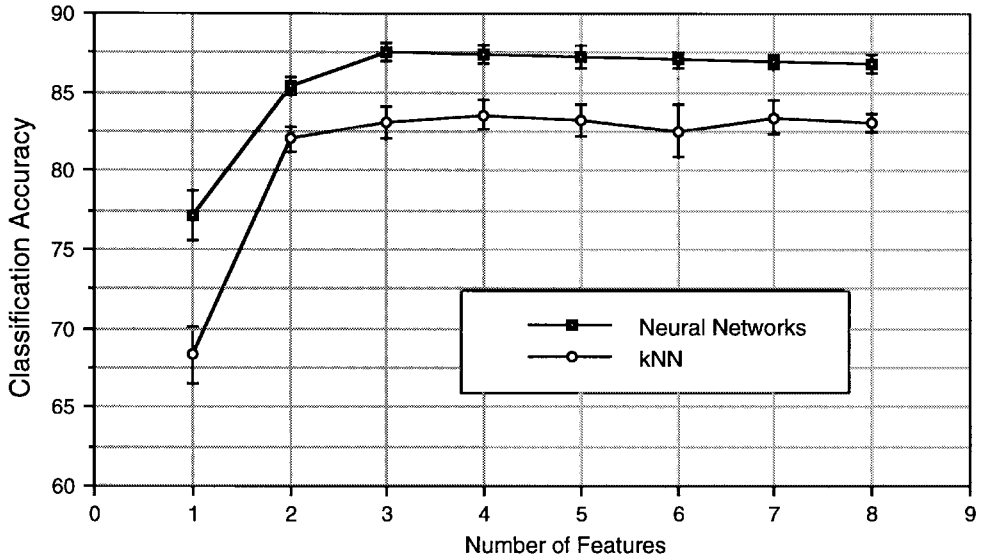

Fig. 10. Classification accuracies of neural networks and the kNN classifier.

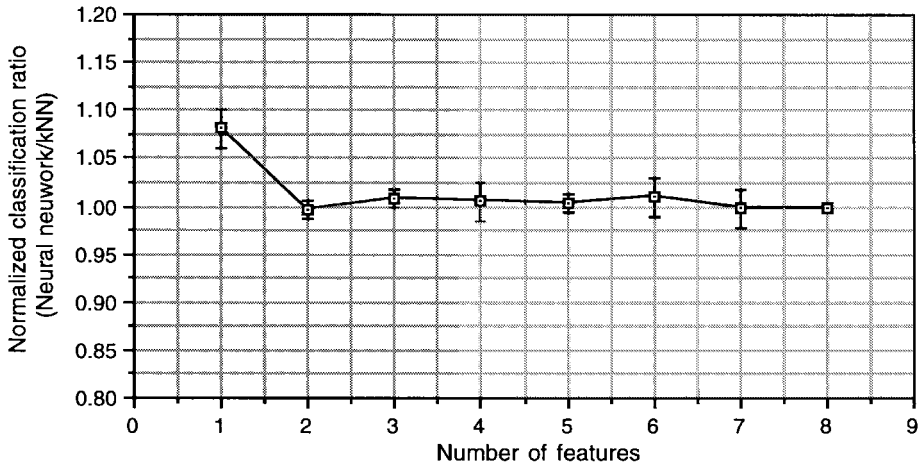

Fig. 11. Normalized classification accuracy ratio between neural networks and the kNN classifier.

eigenvectors is directly related to the proportion of the corresponding eigenvalues, and the eigenvectors corresponding to small eigenvalues make negligible contribution in improving the classification accuracy.

The proposed algorithm preserves the nature of neural networks, which can define quite complex decision boundaries. By employing the proposed algorithms, it is possible to reduce the number of features, and equivalently the number of neurons substantially. This reduction results in simpler networks and shorter classification time for test data. As a result, the proposed algorithm can be successfully applied to applications which utilize neural networks for large data sets or repeated use. The proposed algorithm has been tested on various data set using different initialization and control parameters (learning rate, the number of hidden neurons and etc.). The data includes remotely sensed data, multisource data, and simulated data specially designed to test the robustness of the algorithm. The proposed algorithm consistently gives satisfactory results and appears robust.

In addition, the DBFE algorithm will be helpful to understand a given problem. Since neural networks can solve complex problems without assuming any underlying probability functions and the DBFE algorithm can find a relevant subspace in a way no other feature extraction method can, they can together provide a helpful insight into the problem.

REFERENCES

[1] O. Ersoy and D. Hong, "Parallel self-organizing hierarchical neural networks," *IEEE Trans. Neural Networks*, vol. 1, June 1990.
[2] F. Fukushima and N. Wake, "Handwritten alphanumeric character recognition by the neocognitron," *IEEE Trans. Neural Networks*, vol. 2, pp. 355–365, May 1991.
[3] A. Moonpenn, J. Lambe, and A. P. Thakoor, "Electronic implementation of associative memory based on neural-network models," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, pp. 325–331, Mar./Apr. 1987.
[4] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* New York: Academic, 1990.
[5] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973.
[6] K. Fukunaga and W. L. G. Koontz, "Application of the Karhunen–Loeve expansion to feature selection and ordering," *IEEE Trans. Comput.* vol. C-19, pp. 311–318, Apr. 1970.

[7] D. H. Foley and J. W. Sammon, "An optimal set of discriminant vectors," *IEEE Trans. Comput.*, vol. C-24, pp. 281–289, Mar. 1975.
[8] K. Fukunaga and J. M. Mantock, "Nonparametric discriminant analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 671–678, Nov. 1983.
[9] E. A. Patrick and F. P. F. II, "Nonparametric feature selection," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 577–584, 1969.
[10] R. D. Short and K. Fukunaga, "Feature extraction using problem localization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-4, pp. 323–326, May 1982.
[11] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Networks*, vol. 2, pp. 53–58, 1989.
[12] P. Foldiak, "Adaptive network for optimal linear feature extraction," in Proc. *IEEE Int. Joint Conf. Neural Networks*, Washington, D.C., 1989, pp. 401–405.
[13] 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, 1992, pp. 335–340.
[14] J. Mao and A. K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," *IEEE Trans. Neural Networks*, vol. 6, pp. 296–317, 1995.
[15] 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.
[16] 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.
[17] J. Mao and A. K. Jain, "Discriminant analysis neural networks," in Proc. *IEEE Int. Joint Conf. Neural Networks*, San Francisco, CA, 1993, pp. 300–305.
[18] F. Z. Brill, D. E. Brown and W. N. Martin, "Fast genetic selection of features for neural network classifiers," *IEEE Trans. Neural Networks*, vol. 3, pp. 324–328, Mar. 1992.
[19] J. Mao, K. Mohiuddin, and A. K. Jain, "Parsimonious network design and feature selection through node-pruning," in Proc. *Proc. 12th Int. Conf. Pattern Recognition*, Jerusalem, Israel, Oct. 1994, pp. 622–624.
[20] M. Heywood and P. Noakes, "Simple addition to backpropagation learning for dynamic weight pruning, sparse network extraction and fast learning," in Proc. *IEEE 1993 ICNN*, 1993, pp. 620–625.
[21] S. Ridella, G. Speroni, P. Trebion and R. Zunino, "Pruning and rule extraction using class entropy," in Proc. *IEEE1993 ICNN*, 1993, pp. 250–256.
[22] E. D. Karnin, "A simple procedure for pruning backpropagation trained neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 239–242, 1990.
[23] C. Lee and D. A. Landgrebe, "Feature extraction based on decision boundaries," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 388–400, 1993.
[24] ——, "Decision boundary feature extraction for nonparametric classification," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 2, pp. 433–444, 1993.
[25] C. Lee, "Feature extraction and classification algorithms for high-dimensional data," Ph.D. dissertation, Purdue Univ., W. Lafayette, IN, 1992.
[26] R. Lippmann, "An introduction to computing with neural nets," *IEEE Acoust. Speech, Signal Processing Mag.*, Apr. 1987.
[27] L. L. Biehl et. al., "A crops and soils data base for scene radiation research," in *Proc. Machine Processing Remotely Sensed Data Symp.*, W. Lafayette, IN, 1982.

Chulhee Lee (S'89–M'91) received the B.S. and the M.S. degrees in electronics engineering from Seoul National University, Korea, in 1984 and 1986, respectively, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1992.

From 1986 to 1987, he was Researcher in the Acoustic Laboratory at Technical University of Denmark (DTH). From 1993 to 1996, he worked with National Institutes of Health, Bethesda, MD. Currently, he is Professor of the Department of Electronic Engineering at Yonsei University, Seoul, Korea. His research interests include pattern recognition, image/signal processing, neural networks, and artificial intelligence.

Dr. Lee is a member of Tau Beta Pi and Eta Kappa Nu.

David A. Landgrebe (S'54–M'57–SM'74–F'77) received the B.S.E.E., M.S.E.E., and Ph.D. degrees from Purdue University, West Lafayette, IN.

He is presently Professor of Electrical and Computer Engineering at Purdue University. He is a coauthor of the text, *Remote Sensing: The Quantitative Approach*, (New York: McGraw Hill, 1978) and a contributor to the book, *Remote Sensing of Environment*, and the *ASP Manual of Remote Sensing (1st ed.)* (New York: Amer. Soc. Photogrammerty, 1974). His area of specialty in research is communication science and signal processing, especially as applied to earth observational remote sensing. His contributions over the last 25 years in that field have related to the proper design from a signal processing point of view of multispectral imaging sensors, suitable spectral and spectral/spatial analysis algorithms, methods for designing and training classifier algorithms, and overall systems analysis.

He has been a member of the editorial board of the journal *Remote Sensing of Environment*, since its inception. He was President of the IEEE Geoscience and Remote Sensing Society for 1986 and 1987 and a Member of its Administrative Committee from 1979 to 1990. He received that society's Outstanding Service Award in 1988. Dr. Landgrebe is a Fellow of the American Society of Photogrammetry and Remote Sensing, and a Member of the American Association for the Advancement of Science and the American Society for Engineering Education, as well as Eta Kappa Nu, Tau Beta Pi, and Sigma Xi honor societies. He received the NASA Exceptional Scientific Achievement Medal in 1973 for his work in the field of machine analysis methods for remotely sensed earth observational data. He was the 1990 recipient of the William T. Pecora Award, presented by NASA and the U.S. Department of Interior, for his career-long contributions to the field of remote sensing. He was the 1992 recipient of the IEEE Geoscience and Remote Sensing Society's Distinguished Achievement Award.