Neural Networks, Vol. 3, pp. 367–375, 1990
Printed in the USA.

0893-6080/90 $3.00 + .00

ORIGINAL CONTRIBUTION

The Optimised Internal Representation of Multilayer Classifier Networks Performs Nonlinear Discriminant Analysis

ANDREW R. WEBB AND DAVID LOWE

Royal Signals and Radar Establishment, Great Malvern, U.K.

(*Received* 31 *October* 1988; *revised and accepted* 24 *August* 1989)

Abstract—*This paper illustrates why a nonlinear adaptive feed-forward layered network with linear output units can perform well as a pattern classification device. The central result is that minimising the error at the output of the network is equivalent to maximising a* particular *norm, the network discriminant function, at the output of the* hidden *units. The first part of the network is explicitly performing a nonlinear transformation of the data into a space in which the classes may be more easily separated. The specific nature of this transformation is constrained to maximise the network discriminant function. If the targets are appropriately chosen, this discriminant function relates the pseudo-inverse of the total covariance matrix and the* weighted *between-class covariance matrix of the hidden unit patterns. Numerical simulations are presented to illustrate the results.*

Keywords—Adaptive layered networks, Nonlinear discriminant analysis, Learning, Pattern classification, Nonlinear optimisation.

1. INTRODUCTION

Adaptive feed-forward layered networks as exemplified by the *multi-layer perceptron* are known to be particularly useful as pattern classification techniques (see for instance Gorman & Sejnowski, 1988; Webb, Lowe, & Bedworth, 1988). What is not understood is why they perform good classification, and what underlying mechanism is responsible.

In certain instances, it is possible to identify the action of a layered network structure with the operation of a conventional classification scheme. For instance, Bourlard and Kamp (1987) have shown that for a multi-layer perceptron performing an *auto*-associative task with linear output units and a single layer of n_0 hidden units, the network cannot produce an output mapping with a lower mean square error than a classical singular value decomposition of the target data. Therefore, in such a situation the best solution of the problem can be achieved by a network with linear transfer functions throughout. In this case the network parameters are chosen to produce the best rank n_0 approximation to the target matrix so that the multi-layer perceptron performs a principal component analysis of the input data.

Similarly, Gallinari, Thiria, and Fogelman Soulie (1988) studied a *linear* multi-layer perceptron performing an *hetero*-associative mapping. This work made explicit the fact that, for a linear network performing a one-from-N classification, the solution of the weights which minimised the total mean-square output error also maximised the ratio of the determinants of the between-class and total covariance matrices of the patterns at the outputs of the hidden units. If a (linear) transformation of the input data can be made which produces an adequate separation of the classes as determined by the between-class and total covariance matrices, then a subsequent linear regression procedure should produce better classification results than working on the original data. Thus it is clear why the *linear* multi-layer perceptron is capable of performing well in such circumstances.

However, for a more interesting *nonlinear* transformation from the input data to the (usually, though not necessarily, dimension-reducing) space spanned by the hidden units, much less is known outside empirical observation. For instance, this nonlinear transformation may be the usual logistic transformation of the scalar products between input vectors and weight vectors as in the traditional multi-layer

Our interest was drawn to this problem by John Bridle, to whom we are also grateful for various discussions on the topic. We would like to thank Richard Lippman for his comments on an early version of the manuscript, and also thank Françoise Fogelman Soulie, Patrick Gallinari and Herve Bourlard for additional comments which helped to simplify and revise parts of the manuscript.

perceptron. However, more general nonlinear transformations of the input patterns may be considered (Broomhead & Lowe, 1988; Duda & Hart, 1973; Hampson, 1987; Hanson & Burr, 1987; Huang & Lippmann, 1987; Lippmann, 1987; Reilly, Cooper, & Elbaum, 1982; Reilly, Scofield, Elbaum, & Cooper, 1987).

This study reports theoretical and numerical results on a subclass of general layered nonlinear feed-forward adaptive networks which will demonstrate why such networks have the ability to perform classification tasks successfully. It will be proven that the discriminatory ability stems from the first half of the feed-forward network performing a specific nonlinear transformation of the input data into a space in which the discrimination should be easier. The precise form of this nonlinear transformation ensures that a particular network discriminant function, peculiar to feed-forward layered networks, is maximised.

Of interest to this paper is that class of layered feed-forward network which takes input data, performs an *arbitrary nonlinear* transformation to a space controlled by 'hidden' units, and finally executes a *linear* transformation which attempts to minimise the mean-square error to a set of known output targets. By a theoretical study of this structure, it will be apparent that a good discrimination between classes *in the space of the hidden units* is obtained by requiring a minimisation of the output error.

2. DISCUSSION OF THE NETWORK

This section introduces notation and the subclass of networks relevant to the theorem proposed in section 3.

As conventionally posed, we assume that the parameters (the weights and biases) of the network are specified in order to minimise the sum square error over a finite set of P training patterns. For the specific case of a multi-layer perceptron with a single hidden layer, the error may be expressed as

$$E = \sum_{p=1}^{P} \|\mathbf{t}^p - \mathbf{o}^p\|^2 \equiv \sum_{p=1}^{P} \sum_{k=1}^{n'} \left\{ t_k^p - \left(\lambda_{0k} + \sum_{j=1}^{n_0} h_j \left[\mu_{0j} + \sum_{i=1}^{n} x_i^p \mu_{ij} \right] \lambda_{jk} \right) \right\}^2 \quad (1)$$

where $\mathbf{t}^p \in \mathbf{R}^{n'}$ is the pth desired target pattern vector and $\mathbf{o}^p$ is the actual output of the network.[1]

The network output for input pattern $\mathbf{x}^p \in \mathbf{R}^n$ is determined by the weights $\{\mu_{ij}\}$ connecting the ith input node to the jth hidden node and $\{\lambda_{jk}\}$ connecting the jth hidden unit to the kth output node (μ_{0j} is the bias on the jth hidden node and λ_{0k} is the bias on the kth output node). The function h_j is the nonlinear transfer function of the jth hidden node which may be different for each node. Also, although we have assumed that the input to the jth hidden node is determined by a scalar product between the input pattern and the n_0 weight vectors of the first layer, other forms of transformation may be allowed (Broomhead & Lowe 1988; Hanson & Burr, 1987; Reilly et al., 1987) without affecting the following arguments. One slight restriction of the considered network is that we assume that the transfer functions of the output nodes are linear. The important consequence of this restriction, analytically and numerically, is that the weights connecting the hidden units to the output units may be analysed by linear optimisation methods. In particular, *given* the set of weights connecting the input to hidden units, the hidden-to-output weights may be adjusted by a linear least-mean-squares method to produce a *global* minimum in the error subspace spanned by the set of weights $\{\lambda_{jk}\}$. Consequently, given this latter set of weights, the initial input-to-hidden weights may be adjusted by a nonlinear optimisation strategy to find a better local minimum in the error subspace determined by the set of weights $\{\mu_{ij}\}$. This procedure may continue iteratively. For every "slow" adjustment of the input-to-hidden weights, the hidden-to-output weights respond rapidly always maintaining the global error minimum in that subspace—the output weights are "slaved" to the behaviour of the input weights (for a numerical comparison between this hybrid methodology and solving the entire set of weights by nonlinear optimisation see Webb & Lowe, 1988). This hybrid method is also closely related to the solution of the radial basis function network (Broomhead & Lowe, 1988) if the radial basis function centres (corresponding to the knots in curve fitting) are allowed to adjust themselves by nonlinear methods (Lowe, 1989).

Since this error (1) is an explicit, differentiable nonlinear function of the parameters of interest, one can use one of the many nonlinear optimisation techniques to find an acceptable local minimum (Webb et al., 1988; see also Watrous, 1986; Kollias & Anastassiou, 1988) which will give a suitable set of weight values. Although many other error functions may be chosen to be minimised at the output of the network, we will see that this particular choice has merits for discriminant analysis.

3. THEORETICAL ANALYSIS

The error at the output of the network introduced in the previous section may be expressed in matrix

[1] Vectors are denoted by small boldface symbols, matrices by capital boldface symbols. Components of vectors and matrices are not in a boldface. The adjustable parameters are denoted by Greek symbols and actual patterns by Roman symbols.

notation as

$$E = \|\mathbf{\Lambda H} - \mathbf{T}\|^2 \equiv \mathrm{Tr}[(\mathbf{\Lambda H} - \mathbf{T})(\mathbf{H}^*\mathbf{\Lambda}^* - \mathbf{T}^*)], \quad (2)$$

where $\mathbf{A}^*$ indicates the transpose of matrix $\mathbf{A}$ and Tr denotes the trace operation. The matrix $\mathbf{\Lambda}$ is an $n' \times (n_0 + 1)$ array of weight values, including the biases,

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_{01} & \lambda_{11} & \cdots & \lambda_{n_0 1} \\ \vdots & \vdots & \cdots & \vdots \\ \lambda_{0n'} & \lambda_{1n'} & \cdots & \lambda_{n_0 n'} \end{bmatrix}. \quad (3)$$

Matrix $\mathbf{T}$ is an $n' \times P$ array of desired 'target' values, i.e., P vectors each of length n'. Matrix $\mathbf{H}$ is the $(n_0 + 1) \times P$ array of the P output vectors of the n_0 hidden units plus a unit with unity output to feed the bias weights. Matrix $\mathbf{H}$ may be expressed as

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ h_1^1 & h_1^2 & \cdots & h_1^P \\ \vdots & \vdots & \cdots & \\ h_{n_0}^1 & h_{n_0}^2 & \cdots & h_{n_0}^P \end{bmatrix}, \quad (4)$$

where h_j^p, $j = 1, \ldots, n_0$, $p = 1, \ldots, P$ is the output value at the jth hidden unit corresponding to the pth pattern.

Since the output transfer functions are linear, an optimum set of final layer weights and biases may be obtained analytically which minimises the error expression. It is found that the minimum error expression is given by

$$E = \|\hat{\mathbf{T}} - \mathbf{\Lambda}'\hat{\mathbf{H}}\|^2, \quad (5)$$

where $\mathbf{\Lambda}'$ is the final layer weight matrix (without the bias vector) with an optimum solution of

$$\mathbf{\Lambda}' = \hat{\mathbf{T}}\hat{\mathbf{H}}^+, \quad (6)$$

where $\hat{\mathbf{H}}^+$ is the Moore–Penrose pseudo-inverse (Golub & Kahan, 1965) of $\hat{\mathbf{H}}$. Note that, generally, there will be many distinct sets of weight values which can minimise the overall error. The pseudo-inverse gives one particular set with the smallest overall norm.

The matrices $\hat{\mathbf{T}}$, $\hat{\mathbf{H}}$ are the target and hidden layer output matrices (without the bias vector) with the respective mean vectors subtracted off, i.e.,

$$\hat{\mathbf{T}} \triangleq \mathbf{T} - \bar{\mathbf{t}}\mathbf{1}^*, \quad \hat{\mathbf{H}} \triangleq \mathbf{H}' - \mathbf{m}^{\mathbf{H}}\mathbf{1}^*, \quad (7)$$

where

$$\bar{\mathbf{t}} \triangleq \frac{1}{P}\mathbf{T}\mathbf{1}, \quad (8)$$

is the mean target vector and

$$\mathbf{m}^{\mathbf{H}} \triangleq \frac{1}{P}\mathbf{H}'\mathbf{1} \quad (9)$$

is the mean output vector *at the hidden units*. Matrix $\mathbf{H}'$ is the matrix $\mathbf{H}$ (4) without the first row and $\mathbf{1}$ is a vector with all components equal to unity.

These expressions have used the fact that the optimum value for the output bias vector is

$$\boldsymbol{\lambda}_0 = \bar{\mathbf{t}} - \mathbf{\Lambda}'\mathbf{m}^{\mathbf{H}}. \quad (10)$$

An equivalent result to (10) was derived by Bourlard and Kamp (1987) for the auto-associative network, and it is also known in linear regression theory (Young & Calvert, 1974, p. 141). It simply states that the optimum bias vector compensates for the difference between the mean output vector of the network and the mean target vector over all the training patterns.

Substituting (6) into the error expression (5) and exploiting the properties of the pseudo-inverse gives successively,

$$\begin{aligned} E &= \mathrm{Tr}\{C\hat{\mathbf{T}} - \mathbf{\Lambda}'\hat{\mathbf{H}})(\hat{\mathbf{T}} - \mathbf{\Lambda}'\hat{\mathbf{H}})^*\} \\ &= \mathrm{Tr}\{\hat{\mathbf{T}}\hat{\mathbf{T}}^* \\ &\quad - \hat{\mathbf{T}}\hat{\mathbf{H}}^+\hat{\mathbf{H}}\hat{\mathbf{T}}^* - \hat{\mathbf{T}}\hat{\mathbf{H}}^*(\hat{\mathbf{H}}^+)^*\hat{\mathbf{T}}^* \\ &\quad + \hat{\mathbf{T}}\hat{\mathbf{H}}^+\hat{\mathbf{H}}\hat{\mathbf{H}}^*(\hat{\mathbf{H}}^+)^*\hat{\mathbf{T}}^*\} \\ &= \mathrm{Tr}\{\hat{\mathbf{T}}\hat{\mathbf{T}}^* - \hat{\mathbf{T}}\hat{\mathbf{H}}^*(\hat{\mathbf{H}}\hat{\mathbf{H}}^*)^+\hat{\mathbf{H}}\hat{\mathbf{T}}^*\}. \end{aligned} \quad (11)$$

Note that the matrix $(\hat{\mathbf{H}}\hat{\mathbf{H}}^*)$ is the *total covariance matrix*, $\mathbf{S}_T$ at the output of the hidden units,

$$\mathbf{S}_T = \hat{\mathbf{H}}\hat{\mathbf{H}}^* = \sum_{p=1}^{P} (\mathbf{h}^p - \mathbf{m}^{\mathbf{H}})(\mathbf{h}^p - \mathbf{m}^{\mathbf{H}})^*. \quad (12)$$

Thus, since the targets are fixed, minimising E is equivalent to maximising the discriminant function

$$C = \mathrm{Tr}\{\mathbf{S}_B\mathbf{S}_T^+\}. \quad (13)$$

where

$$\mathbf{S}_B \triangleq \hat{\mathbf{H}}\hat{\mathbf{T}}^*\hat{\mathbf{T}}\hat{\mathbf{H}}^*. \quad (14)$$

Equation (13) is the *network discriminant function*. It is only dependent on the weights in the first part of the network.

This result may be summarised as follows:

Choosing optimum weights and biases to minimise the square error at the output of the adaptive layered network, forces the first set of weights to be chosen such that the transformation from the input data to the output of the final hidden layer maximises the network discriminant function (13) in the space spanned by the outputs of the final hidden layer.

The significance of this result is clear. The network was constructed in such a way that the final layer performed an optimum separation of patterns into their classes by a linear transformation. In order to minimise the error of this procedure, it is necessary for the first part of the network to perform an appropriate *nonlinear* transformation of the original patterns into a space in which discrimination is made easier. This nonlinear transformation has to be such that the network discriminant function is maximised. The space generated by the outputs of the hidden units is the crucial one in which pattern analysis and

recognition should be performed. Thus the final linear transformation used in the network could be replaced by a more sophisticated statistical pattern analysis method which would benefit from the discriminatory power of the feed-forward network.

An additional insight generated by this result, which will become clearer after the discussion on particular target coding schemes, is that the transformation into a space in which discrimination is made easier may be precisely adapted to compensate for prior probabilities due to uneven class membership between train and test sets or relative class significance by appropriately modifying the error metric and/or scaling the hidden unit outputs during training.

Although the link with linear discriminant analysis has previously been made (Gallinari et al., 1988) for fully *linear* networks, the result presented here allowing for *nonlinear* transformations leads to a different discriminant function being maximised. However, it will be shown later that, restricting the network to only performing linear transformations, implies that the optimum weights which maximise the ratio-of-determinants discriminant function (as proposed in Gallinari et al., 1988) also maximise the network discriminant function. Thus, our result is a natural generalisation of the earlier work. In addition, the current analysis allows one to consider transformations into a space of higher dimensionality than the original data or target space. Normally, one conceives of dimension-reducing transformations (linear transformations limit the rank of the image space to be one less than the rank of the target or input space). Nonlinear transformations allow the flexibility to transform into higher-dimensional spaces in which discrimination is easier to perform. The discriminant function proposed by Gallinari, Thiria and Fogelman Soulie (1988) would become singular in this situation whereas the network discriminant function remains well-behaved. This is illustrated in the numerical example later on.

3.1. Particular Target Coding Schemes

We now consider the *interpretation* of the network discriminant function for various choices of target coding. Consider the specific choice of a one-from-n' coding scheme. Along with the other assumptions made on the form of the adaptive layered network structure, the desired target value of a particular pattern is unity if the chosen input pattern is in that class, and is zero otherwise. If there are n' classes, $\mathcal{C}_k$, $k = 1, \ldots, n'$ with n_k patterns in class $\mathcal{C}_k$, then for this particular coding scheme, the matrix $\mathbf{S}_B$ introduced in the previous section may be expanded as

$$\begin{aligned}\mathbf{S}_B &= \hat{\mathbf{H}}\hat{\mathbf{T}}^*\hat{\mathbf{T}}\hat{\mathbf{H}}^* \\ &= \sum_{k=1}^{n'} n_k^2(\mathbf{m}^{\mathbf{H}_k} - \mathbf{m}^{\mathbf{H}})(\mathbf{m}^{\mathbf{H}_k} - \mathbf{m}^{\mathbf{H}})^*, \end{aligned} \tag{15}$$

where $\mathbf{m}^{\mathbf{H}_k}$ is the mean output vector at the hidden units over all patterns in class $\mathcal{C}_k$,

$$\mathbf{m}^{\mathbf{H}_k} = \frac{1}{n_k} \sum_{\mathbf{h}^p \in \mathcal{C}_k} \mathbf{h}^p. \tag{16}$$

This equation (15) is recognised as the expression for the *weighted* between-class covariance matrix. Thus, for a one-from-n' output coding, the layered network maximises a discriminant function which is the trace of the product of the weighted between-class covariance matrix, and the inverse of the total covariance matrix. This is an interesting result, since it illustrates how adaptive layered networks implicitly incorporate the proportions of samples within each class as priors. It also demonstrates explicitly that networks trained with a one-from-n' target coding bias *strongly* in favour of those classes with largest membership. This may not be a desirable feature for certain pattern classification tasks where the importance of individual classes is equal, but there are many more patterns in one of the classes.

Consider an alternative coding scheme: the target of an input pattern is zero if the pattern is not in the class under consideration and is the reciprocal of the square root of the number of patterns in that class otherwise.

$$t_{kp} = \begin{cases} 1/\sqrt{n_k} & \text{if } \mathbf{h}^p \in \mathcal{C}_k \\ 0 & \text{otherwise} \end{cases}$$

In this case, the matrix $\mathbf{S}_B$ expands to

$$\tilde{\mathbf{S}}_B = \sum_{k=1}^{n'} n_k(\mathbf{m}^{\mathbf{H}_k} - \mathbf{m}^{\mathbf{H}})(\mathbf{m}^{\mathbf{H}_k} - \mathbf{m}^{\mathbf{H}})^*, \tag{17}$$

which is the *conventional* between-class covariance matrix. Thus, in a pattern classification problem where the number of training examples is unevenly distributed across the classes and which would be solved best by producing a discrimination which weighted the classes according to their class membership *in the training set* an adaptive layered network trained on a one-from-n' coding scheme would not produce the best results. Furthermore, this analysis is extendable to incorporate the expected class importance *on the test set* and compensate for unequal class membership on the training data.

There are two important instances where the distinction between the weighted and not-weighted between-class covariance matrices will *not* be made: a multi-class problem when the number of patterns in each class is the same, and in a *two-class* problem with unequal class membership. In these instances, the weighted between-class covariance matrix $\mathbf{S}_B$, and the conventional between-class covariance matrix $\tilde{\mathbf{S}}_B$, are connected by a multiplicative constant. For the latter case this relationship is

$$\mathbf{S}_B = \frac{2n_1n_2}{P}\tilde{\mathbf{S}}_B.$$

Thus, maximising the discriminant function with the weighted covariance will give the same result as maximising with the conventional covariance matrix for a two-class problem.

One final interesting point to note, is that for problems where there is only *one pattern per class* then the total covariance matrix and the between-class covariance matrix (weighted, or otherwise) are identical. The network discriminant function is a constant, *irrespective of the pattern distribution of the hidden unit patterns*. This implies that the network discriminant function will be maximised whatever initial value of random weight starts is chosen for the input-hidden layer of weights, and hence the error is automatically minimised—no iterative improvement of the error is possible.

3.2. The Linear Adaptive Layered Network

This section illustrates that our result incorporates the earlier result (Gallinari et al., 1988) on linear discriminant analysis as a special case. In this instance the adaptive layered network as a whole performs a linear transformation between the input and output spaces; the *rank* of the mapping is determined by the number of hidden units of the network (and the number of hidden units has to be less than or equal to the rank of the space spanned by the target vectors). The result which was previously illustrated (Gallinari et al., 1988) (under certain reasonable assumptions) was that the weight matrix between the input and hidden units which minimised the square error of the network, also maximised the discriminant function

$$\tilde{C}(\mathbf{W}') = \frac{|\mathbf{W}'^{*}\tilde{\mathbf{S}}_B^l\mathbf{W}'|}{|\mathbf{W}'^{*}\mathbf{S}_T^l\mathbf{W}'|}, \tag{18}$$

which is the ratio of the determinants of the between-class, and total covariance matrices in the transformed space of the input patterns. In this equation, $\tilde{\mathbf{S}}_B^l$ and $\mathbf{S}_T^l$ are the between-class and total covariance matrices of the original input data. Other discriminant functions are maximised by the same transformation (Fukunaga, 1972). The choice (18) is not the natural discriminant function which the nonlinear adaptive layered network maximises, as we have proven.

However, if the first part of the network performs a linear transformation, then the network discriminant function which is maximised may be expressed as

$$C = \mathrm{Tr}\{\mathbf{W}'\mathbf{S}_B^l\mathbf{W}'^{*}(\mathbf{W}'\mathbf{S}_T^l\mathbf{W}'^{*})^{-1}\}, \tag{19}$$

where $\mathbf{W}'$ is the matrix of weights between the input and hidden layers and matrices $\mathbf{S}_B^l$, $\mathbf{S}_T^l$ are the weighted between-class and total covariance matrices of the original data

$$\mathbf{S}_B^l \triangleq \hat{\mathbf{I}}\hat{\mathbf{T}}^{*}\hat{\mathbf{T}}\hat{\mathbf{I}}^{*}, \tag{20}$$

$$\mathbf{S}_T^l \triangleq \hat{\mathbf{I}}\hat{\mathbf{I}}^{*}, \tag{21}$$

and $\hat{\mathbf{I}}$ is the matrix of mean-shifted input patterns. Again, the optimum bias vector at the hidden units is chosen to compensate for the difference between the transformation of the mean of the input data, and the mean of the transformed patterns.

One can observe that in the linear case, the solution for $\mathbf{W}'$ which maximises (18) may be composed out of the eigenvectors of $\tilde{\mathbf{S}}_B^l(\mathbf{S}_T^l)^{-1}$ corresponding to the nonzero eigenvalues (giving a specific solution $\mathbf{W}_0$). The solution for $\mathbf{W}'$ which maximises (19) may be composed out of the eigenvectors of $\mathbf{S}_B^l(\mathbf{S}_T^l)^{-1}$ corresponding to nonzero eigenvalues. The matrices $\tilde{\mathbf{S}}_B^l(\mathbf{S}_T^l)^{-1}$, and $\mathbf{S}_B^l(\mathbf{S}_T^l)^{-1}$ both have the same rank and the corresponding eigenvectors span the same image space. Thus, there exists an invertible linear transformation between the two sets of eigenvectors. Therefore, since *any* linear invertible transformation of $\mathbf{W}_0$ also maximises (18), any solution which maximises (18) must also maximise (19). This is why it was permissible for Gallinari, Thiria, and Fogelman Soulie (1988) to use the conventional between-class covariance matrix as opposed to the *weighted* between class covariance matrix. In addition, it is also clear that in this linear case, the network discriminant function is maximised by the *sum* of the eigenvalues of the matrix $\mathbf{S}_B^l(\mathbf{S}_T^l)^{-1}$, whereas the ratio-of-determinants discriminant function is maximised to the *product* of the eigenvalues.

4. NUMERICAL ILLUSTRATION

In this section, the implications of the theorem in section 3 are illustrated. The chosen synthetic problem is to determine the number of groups of 1's in an 8-bit binary string. This is a generalisation of the Penzias problem (or the two-or-more clumps predicate (Denker et al., 1987) where the number of contiguous clumps of "1" determines which *class* the input pattern belongs to. Thus, for the 8-bit binary string there are 256 distinct patterns, each of which belongs to one of five classes. Table 1 gives the number of members of each class. Despite the lack of an intuitive interpretation of covariance matrices for such a problem, it is still true that the network discriminant function ought to be maximised.

The specific nonlinear transformation from the input layer to the hidden layer employed in this test is that transformation as determined by a multi-layer perceptron with 8 input units, 5 output units and a varying number of hidden units as the classification network. The output units have a linear transfer function. The output coding scheme adopted is a one-from-five coding, so that the matrix $\mathbf{S}_B$ is given

TABLE 1
Numbers of members in each class for groups of digit 1's

Class	Number of groups	Number of members in class
1	0	1
2	1	36
3	2	126
4	3	84
5	4	9

by eqn (15). Since there is an unequal number of members in each class, this matrix is not proportional to the conventional between-class covariance matrix $\tilde{\mathbf{S}}_B$ (17) and therefore differences should be observable between the results of the network discriminant function and the linear discriminant functions.

Four discriminant functions were evaluated at the output of the hidden units, namely the network discriminant function, $C = \mathrm{Tr}(\mathbf{S}_B\mathbf{S}_T^{-1})$; the function, $\mathrm{Tr}(\tilde{\mathbf{S}}_B\mathbf{S}_T^{-1})$; and the ratios of determinants $|\mathbf{S}_B|/|\mathbf{S}_T|$ and $|\tilde{\mathbf{S}}_B|/|\mathbf{S}_T|$. The method of solution of the least squares problem uses an iterative scheme to minimise the error (see Appendix A) and the discriminant functions are evaluated at each stage of the iteration.

This problem is one which cannot be solved by a linear transformation from input space to output space, or equivalently, a network with four linear transfer functions at the hidden layer (Gallinari et al., 1988). Performing a least-means-squares mapping from the 8 input units directly to the 5 output units, and classifying the patterns according to the minimum Euclidean distance in the output space gives 132 (= 51.56%) correct solutions (and one indeterminate solution since the null vector maps on to the null vector, which is equidistant from all classes). A network with four (linear) hidden units achieves the same performance, though the addition of the biases does enable the null vector to be classified correctly. Figure 1 plots the discriminants as a function of iteration number in the error minimisation routine. It is evident that all the discriminant functions considered were maximised as the error was iteratively minimised. In addition, we can be sure that a solution corresponding to the minimum error was obtained since, from the analysis of the previous section, we know that the network discriminant function has a maximum value equivalent to the sum of the eigenvalues of the generalised eigenvalue equation, whereas the ratio of determinants is maximised to the product of these eigenvalues. For this particular problem the nonzero eigenvalues are

$$\begin{aligned} \alpha_1 &= 11.77103, \\ \alpha_2 &= 2.618924, \\ \alpha_3 &= 0.3970126, \\ \alpha_4 &= 0.02553337. \end{aligned} \tag{22}$$

Thus the sum of the eigenvalues is 14.815 (compared with the maximum obtained by the network discriminant function of 14.8124 in Figure 1) and the product is 0.3125 (compared with the ratio-of-determinants final value of 0.3088 in the figure). However, we found that the ratio-of-determinants discriminant was not as robust to numerical error as the network discriminant function. In particular, note that the curves in Figure 1 corresponding to the ratio-of-determinants varies dramatically over an iteration span in which the network discriminant function remains almost constant. In this region a partial solution had been obtained which was not quite the absolute minimum error obtainable and although the network discriminant function had almost reached its plateau, reflecting this error trend, the ratio-of-determinants discriminant can still fluctuate wildly. Thus, the ratio-of-determinants discriminant does not maximise gracefully, as the network error is minimised.

Introducing a nonlinear transfer function at each of the hidden units gives improved classification performance. Figure 2 plots the discriminants as a function of iteration number for a network with four hidden units. For the particular random start configuration of the weights and biases chosen, the network achieved 189 (73.83%) correct solutions. The network discriminant function increases monotonically with the number of iterations of the algorithm. The sum-squared error at the output decreases monoton-

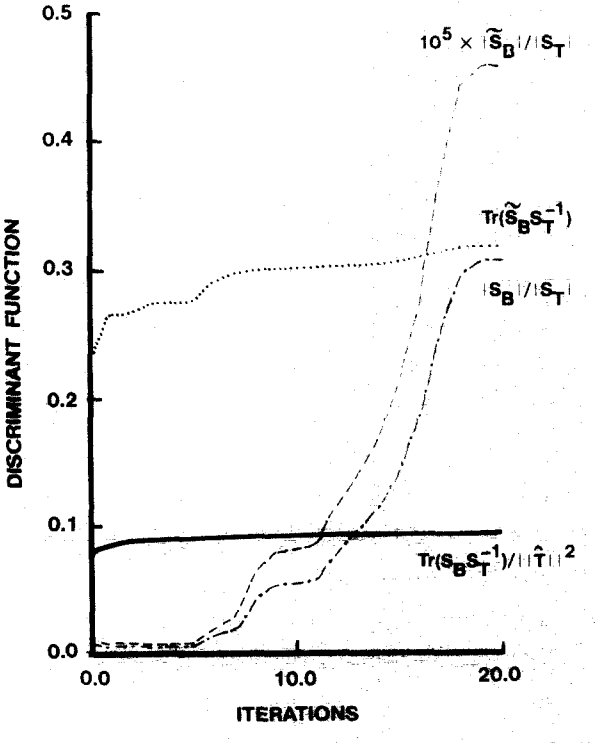

FIGURE 1. Plots of various discriminant measures as a function of iteration number as the square error is minimised. The network used four hidden units with linear transfer functions.

ically correspondingly. However, the function $C = \mathrm{Tr}(\tilde{\mathbf{S}}_B \mathbf{S}_T^{-1})$ settles at a value which is not its peak value during the iteration. Both discriminant functions which depict the ratio of determinants of the between-class and total covariance matrices are *not* maximised. In fact, in the situation where the number of hidden units is equal to one less than the number of classes ($n_0 = n' - 1$), as illustrated in this example, the determinants $|\mathbf{S}_B|$ and $|\tilde{\mathbf{S}}_B|$ are related by

$$|\mathbf{S}_B| = n' \frac{\prod_{i=1}^{n'} n_i}{\sum_{i=1}^{n'} n_i} |\tilde{\mathbf{S}}_B|. \tag{23}$$

Thus both discriminants exhibit the same behaviour, as observed in the figures.

With a nonlinear network, we are not restricted to having fewer hidden units than the number of classes as in linear discriminant analysis and Figure 3 plots the discriminant functions as a function of iteration number for a network with six hidden units. With this number of hidden units, the determinant of the between-class covariance matrix is identically equal to zero since the dimension of the matrix $\mathbf{S}_B$ is greater than the number of classes. Therefore, it is not meaningful to use the ratio-of-determinants discriminants as a measure of classification performance. This emphasises the limitation of the results derived by the linear network. An additional discriminant function, $\mathrm{Tr}\{\tilde{\mathbf{S}}_B\}/\mathrm{Tr}\{\mathbf{S}_T\}$, the ratio of traces of the between-class and total covariance matrices has also been plotted for comparison. Note that for the particular random start configuration used for this figure, the network achieved 192 (= 75%) correctly classified solutions.

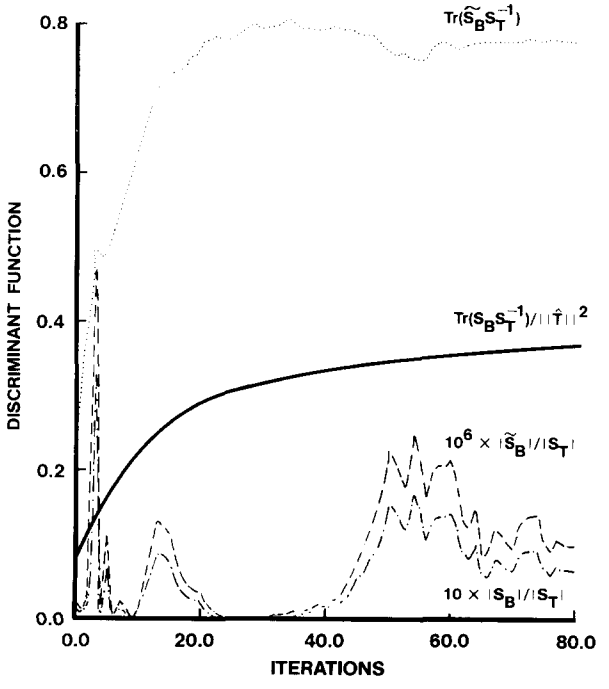

FIGURE 2. Plots of various discriminant measures as a function of iteration number as the square error is minimised. The network used four hidden units with nonlinear transfer functions.

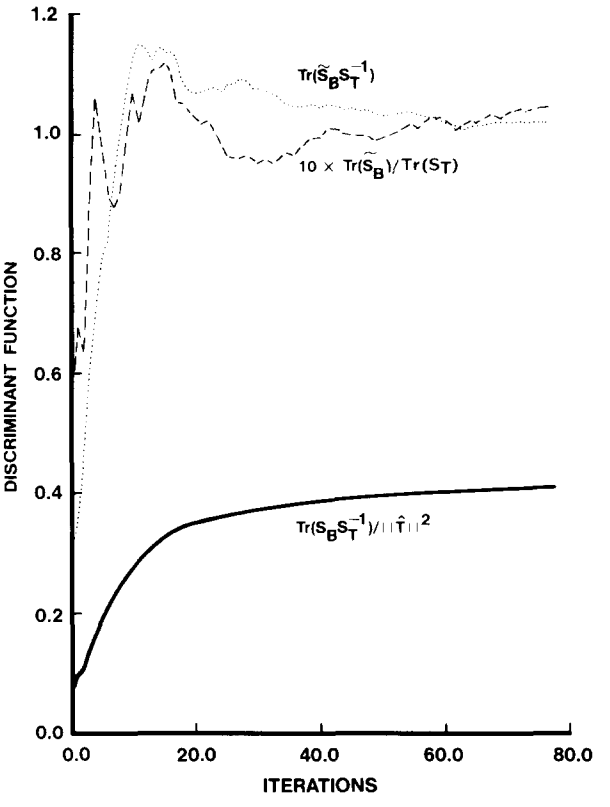

FIGURE 3. Plots of various discriminant measures as a function of iteration number as the square error is minimised. The network used *six* hidden units with nonlinear transfer functions.

This figure shows that the network discriminant function is maximised as the error is minimised but the trace of the product of the conventional between-class covariance matrix and the inverse of the total covariance matrix is not maximised.

These figures illustrate a natural evolution of discriminant analysis strategies as produced by a feed-forward adaptive layered network. The linear network produces an optimum linear transformation to a dimension reducing subspace where the patterns corresponding to different classes are in some sense maximally separated, and the patterns within each class are grouped according to linear discriminant analysis (this is the example illustrated by Figure 1). The next step is to allow for a *nonlinear* transformation on to a dimension-reducing subspace which should have the advantage of providing a better class discrimination transformation (the example illustrated by Figure 2). The final stage (Figure 3) is to allow for an embedding of the input patterns by a nonlinear transformation to a higher-dimensional space where an even better class separation may be achieved. This is something beyond the scope of linear discriminant analysis. Once a transformation has been performed into a space where the transformed

patterns are more easily distinguished, it is much easier for linear regression (the hidden-output layer of the multi-layer perceptron considered in the paper) to perform good classification. These general comments are reflected in the classification performance of the figures which rises from 132 to 192 patterns classified correctly. However, note that in all instances, the criterion for maximal class separation is determined by the network discriminant function.

5. CONCLUSION

Adaptive feed-forward layered networks are capable of performing classification tasks better than traditional methods. This paper has demonstrated that this ability arises out of the implicit way in which input data is mapped under a nonlinear transformation which maximises the network discriminant function in the space spanned by the hidden units.

This paper considered a general nonlinear transformation from the input patterns to a set of patterns in the space defined by the final layer of hidden units (there is no restriction on the number of layers constituting the nonlinear transformation) followed by a linear transformation to a set of output target patterns. If the network weights are adjusted to minimise the mean-square error between the desired target patterns and the actual output patterns of the network, then the transformation which achieves this in the first part of the network maximises the network discriminant function

$$C = \mathrm{Tr}\{\mathbf{S}_B\mathbf{S}_T^{\dagger}\} \tag{24}$$

at the output of the hidden layer, where $\mathbf{S}_B$ is defined in eqn (14) and $\mathbf{S}_T$ is the total covariance matrix of the patterns at the outputs of the final hidden layer. The matrix $\mathbf{S}_B$ may be interpreted to be the weighted between-class covariance matrix at the output of the final hidden layer if the target patterns are chosen as a one-from-n' coding. Equivalently, encoding the distribution of patterns between the classes into the target patterns (which is equivalent to weighting the error minimisation) allows the matrix $\mathbf{S}_B$ to be interpreted as the conventional between-class covariance matrix.

In traditional multi-class linear discriminant analysis, many different discriminant functions have been employed which are all generalisations of Fisher's linear discriminant, a linear transformation from a multi-dimensional problem to a one-dimensional problem (in the sense that for a two-class problem, maximising one of these generalised discriminant functions is equivalent to maximising Fisher's linear discriminant). There is often no criterion to choose one discriminant function in preference to the alternatives for multi-class problems. The action of a feed-forward network does *not* maximise more traditional functions employed in discrimination analysis (generally), as our numerical example illustrated. Indeed, such networks perform a *specific* type of nonlinear discriminant analysis which is constrained to maximise the network discriminant function. In the special case of a totally linear network with one hidden layer, the solution which maximises the network discriminant function also maximises the ratio of determinants of the between-class and total covariance matrices (the result obtained in Gallinari et al., 1988).

Thus an adaptive feed-forward layered network performs a natural generalisation of linear discriminant analysis by finding an embedding transformation which maximises a function relating the between-class and total covariance matrices of the transformed patterns. This is precisely why such networks have been demonstrated to perform classification tasks well.

REFERENCES

Bourlard, H., & Kamp, Y. (1987). *Auto-association by multilayer perceptrons and singular value decomposition* (Manuscript M217, Av. Van Becelaere 2-Box 8, B-1170). Brussels, Belgium: Philips Research Laboratory.

Broomhead, D. S., & Lowe, D. (1988). Multi-variable functional interpolation and adaptive networks. *Complex Systems*, **2**(3), 269–303.

Broyden, C. G. (1967). Quasi-Newton methods and their application to function minimisation. *Mathematics of Computation*, **21**, 368–381.

Byrne, G. D., & Hall, C. A. (Eds.). (1973). Numerical solutions of systems of nonlinear algebraic equations. New York: Academic Press.

Denker, J., Schwartz, D., Wittner, B., Solla, S., Howard R., Jackel, L., & Hopfield, J. (1987). Large automatic learning, rule extraction, and generalisation. *Complex Systems*, **1**, 877–922.

Duda, R. O., & Hart, P. E. (1973). Pattern classification and scene analysis. New York: John Wiley & Sons.

Fukunaga, K. (1972). Introduction to statistical pattern recognition. New York: Academic Press.

Gallinari, P., Thiria, S., & Fogelman Soulie, F. (1988). Multilayer perceptrons and data analysis. *IEEE Annual International Conference on Neural Networks* (pp. I-391–I-399). San Diego: SOS Printing.

Golub, G., & Kahan, W. (1965). Calculating the singular values and pseudo-inverse of a matrix. *Journal SIAM Numerical Analysis, Series B*, **2**(2), 205–224.

Gorman, R. P., & Sejnowski, T. J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, **1**, 75–89.

Hampson, S. E. (1987). Disjunctive models of Boolean category learning. *Biological Cybernetics*, **56**, 121–137.

Hanson, S. J., & Burr, D. J. (1987, February). Knowledge representation in connectionist networks (Internal Report, Bell Communications Research). Morristown, NJ.

Huang, W. Y., & Lippmann, R. P. (1987). Neural net and traditional classifiers. *Conference on Neural Information Processing Systems* (pp. 387–396). Denver, CO: American Institute of Physics, New York, 1988.

Kollias, S., & Anastassiou, D. (1988). Adaptive training of mul-

tilayer neural networks using a least squares estimation technique. In *IEEE International Conference on Neural Networks* (pp. I-383–I-390). San Diego: SOS Printing.

Lippmann, R. P. (1987, April). An introduction to computing with neural nets. *IEEE Acoustics, Speech, and Signal Processing Magazine*, pp. 4–22.

Lowe, D. (1989, October). Adaptive radial basis function nonlinearities and the problem of generalisation. *1st IEE International Conference on Artificial Neural Networks* (pp. 171–175). Institute of Electrical Engineers. Conference Publication No. 313.

Reilly, D. L., Cooper, L. N., & Elbaum, C. (1982). A neural model for category learning. *Biological Cybernetics*, **45**, 35–41.

Reilly, D. L., Scofield, C., Elbaum, C., & Cooper, L. N. (1987). Learning system architectures composed of multiple learning modules. In *IEEE 1st International Conference on Neural Networks* (pp. II-495-II-503). San Diego: SOS Printing.

Watrous, R. (1986). Learning algorithms for connectionist networks: Applied gradient methods of nonlinear optimisation (Tech. Rep. MS-CIS-87-51, LINC LAB 72). University of Pennsylvania.

Webb, A. R., Lowe, D., & Bedworth, M. D. (1988). A comparison of nonlinear optimisation strategies for feed-forward adaptive layered networks (Memorandum 4157). Great Malvern, Worcestershire, U.K.: Royal Signals and Radar Establishment.

Webb, A. R., & Lowe, D. (1988). A hybrid optimisation strategy for adaptive feed-forward layered networks (Memorandum 4193). Great Malvern, Worcestershire, U.K.: Royal Signals and Radar Establishment.

Young, T. Y., & Calvert, T. W. (1974). Classification, estimation and pattern recognition. New York: American Elsevier Publishing Company Inc.

APPENDIX: NUMERICAL SOLUTION OF THE LEAST-SQUARES PROBLEM

In the numerical example used to illustrate the theorem, the network employed had a single hidden layer with the transfer function of the hidden units described by a logistic function,

$$\phi(x) = \frac{1}{1 + \exp(-x)} \tag{25}$$

and an output layer employing linear units.

The square error at the output of the network is regarded as a nonlinear function of the weights and biases between the input layer and the hidden layer. This error may be minimised using any suitable nonlinear function optimisation strategy (Webb et al., 1988), and we have chosen to use a quasi-Newton technique, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method (Broyden, 1967; Byrne & Hall, 1973).

The minimisation proceeds as follows. Given an initial estimate $\{\boldsymbol{\mu}(t = 0)\}$, of the weights and biases $\{\boldsymbol{\mu}\}$ (chosen from a uniform random distribution in the interval $(-1, 1)$) between the input and hidden layers, the final layer weights and biases $\{\boldsymbol{\lambda}\}$ are calculated using eqns (6) and (10) and the value of the output error obtained. The gradient of the error with respect to the parameters $\{\boldsymbol{\mu}\}$ may then be calculated from eqn (1).

Thus, given an initial position and an initial search direction (taken to be the direction of the downhill gradient), the algorithm performs a search along this direction to obtain an estimate of the minimum of the error in this direction. Once this has been achieved, a new search direction is generated (using the BFGS prescription) and a search performed to find the minimum of the error in this new direction. This procedure continues until convergence. Note that each time that the error is evaluated (for each new estimate of the parameters $\{\boldsymbol{\mu}\}$) the values of the parameters $\{\boldsymbol{\lambda}\}$ must be obtained using eqns (6) and (10) *prior* to evaluation of the error. In this way, the values of the parameters $\{\boldsymbol{\lambda}\}$ are tied to the values of $\{\boldsymbol{\mu}\}$. This ensures that the method produces a global minimum in the subspace spanned by the parameters $\{\boldsymbol{\lambda}\}$.

Note that this is not the traditional method of solving such networks. There are advantages and disadvantages to solving such systems layer by layer rather than optimising for all the parameters simultaneously. In particular, solving layer by layer forces the optimisation strategy to operate in a reduced dimension subspace (the space of the input-hidden layer weights) which could increase the chance of unsuitable local minima being encountered. On the other hand, the strategy of solving the final layer of weights by a linear optimisation method does at least ensure that a global minimum solution for the final layer of weights is found, *given* specific values for the previous layers weights. Although for a specific problem, one of the approaches tends to be more effective, there is no obvious advantage to either method: we have not discovered one approach to be obviously superior to the other across a range of problems (Webb & Lowe, 1988).

A final comment is that the search strategy to find a minimum of the nonlinear function could have been performed by a standard (accelerated) steepest descents procedure. In our experience (Webb et al., 1988), this would have taken at least an order of magnitude longer in terms of CPU time, or the number of iterations. It was decided that the BFGS procedure was one of the more efficient techniques to use for this size of problem.