LETTER Communicated by Peter Bartlett

Generalized Discriminant Analysis Using a Kernel Approach

G. Baudat
Mars Electronics International, CH-1211 Geneva, Switzerland

F. Anouar
INRA-SNES, Institut National de Recherche en Agronomie, 49071 Beaucouzé, France

We present a new method that we call generalized discriminant analysis (GDA) to deal with nonlinear discriminant analysis using kernel function operator. The underlying theory is close to the support vector machines (SVM) insofar as the GDA method provides a mapping of the input vectors into high-dimensional feature space. In the transformed space, linear properties make it easy to extend and generalize the classical linear discriminant analysis (LDA) to nonlinear discriminant analysis. The formulation is expressed as an eigenvalue problem resolution. Using a different kernel, one can cover a wide class of nonlinearities. For both simulated data and alternate kernels, we give classification results, as well as the shape of the decision function. The results are confirmed using real data to perform seed classification.

1 Introduction

Linear discriminant analysis (LDA) is a traditional statistical method that has proved successful on classification problems (Fukunaga, 1990). The procedure is based on an eigenvalue resolution and gives an exact solution of the maximum of the inertia. But this method fails for a nonlinear problem. In this article, we generalize LDA to nonlinear problems and develop a generalized discriminant analysis (GDA) by mapping the input space into a high-dimensional feature space with linear properties. In the new space, one can solve the problem in a classical way, such as with the LDA method. The main idea is to map the input space into a convenient feature space in which variables are nonlinearly related to the input space. This fact has been used in some algorithms, such as unsupervised learning algorithms (Kohonen, 1994; Anouar, Badran, & Thiria, 1998) and in support vector machine (SVM) (Vapnik, 1995; Burges, 1998; Schölkopf, 1997). In our approach, the mapping is close to the mapping used for support vector method, a universal tool to solve pattern recognition problems. In the feature space, the SVM method selects a subset of the training data and defines a decision function that is a linear expansion on a basis whose elements are nonlinear functions parameterized by the support vectors. SVM was extended to dif-

ferent domains such as regression estimation (Drucker, Burges, Kaufman, Smola, & Vapnik, 1997; Vapnik, Golowich, & Smola, 1997). The basic ideas behind SVM have been explored by Schölkopf, Smola, and Müller (1996, 1998) to extend principal component analysis (PCA) to nonlinear kernel PCA for extracting structure from high-dimensional data sets. The authors also propose nonlinear variant of other algorithms such as independent component analysis (ICA) or kernel-k-means. They mention that it would be desirable to develop a nonlinear form of discriminant analysis based on the kernel method. A related approach using an explicit map into a higher-dimensional space instead of the kernel method was proposed by Hastie, Tibshirani, and Buja (1993). The foundations for the kernel developments described here can be connected to kernel PCA. Drawn from these works, we show how to express the GDA method as a linear algebraic formula in the transformed space using kernel operators.

In the next section, we introduce the notations used for this purpose. Then we review the standard LDA method. The formulation of the GDA using dot product and matrix form is explained in the third section. Then, we develop the eigenvalue resolution. The last section is devoted to the experiments on simulated and real data.

2 Notations

Let x be a vector of the input set X with M elements. X_l designs subsets of X; thus: $X = \bigcup_{l=1}^{N} X_l$. N is the number of the classes. x^t represents the transpose of the vector x. The cardinality of the subsets X_l is denoted by n_l; thus $\sum_{l=1}^{N} n_l = M$. C is the covariance matrix:

$$C = \frac{1}{M} \sum_{j=1}^{M} x_j x_j^t \tag{2.1}$$

Suppose that the space X is mapped into a Hilbert space F through a nonlinear mapping function ϕ:

$$\begin{aligned} \phi : \; & X \to F \\ & x \to \phi(x). \end{aligned} \tag{2.2}$$

The covariance matrix in the feature space F is:

$$V = \frac{1}{M} \sum_{j=1}^{M} \phi(x_j) \phi^t(x_j). \tag{2.3}$$

We assume that the observations are centered in F. Nevertheless, the way to center the data in the feature space can be derived from Schölkopf et al.

(1998). By B we denote the covariance matrix of the class centers. B represents the interclasses inertia in the space F:

$$B = \frac{1}{M}\sum_{l=1}^{N} n_l \bar{\phi}_l \bar{\phi}_l^t, \tag{2.4}$$

where $\bar{\phi}_l$ is the mean value of the class l,

$$\bar{\phi}_l = \frac{1}{n_l}\sum_{k=1}^{n_l} \phi(x_{lk}), \tag{2.5}$$

and x_{lk} is the element k of the class l.

In the same manner, the covariance matrix (see equation 2.3) of F elements can be rewritten using the class indexes:

$$V = \frac{1}{M}\sum_{l=1}^{N}\sum_{k=1}^{n_l} \phi(x_{lk})\phi^t(x_{lk}). \tag{2.6}$$

V represents the total inertia of the data into F.

In order to simplify, when there is no ambiguity in index of x_{lj} , the class index l is omitted.

To generalize LDA to the nonlinear case, we formulate it in a way that uses dot product exclusively. Therefore, we consider an expression of dot product on the Hilbert space F (Aizerman, Braverman, & Rozonoér, 1964; Boser, Guyon, & Vapnik, 1992) given by the following kernel function:

$$k(x_i, x_j) = k_{ij} = \phi^t(x_i)\phi(x_j).$$

For a given classes p and q, we express this kernel function by:

$$(k_{ij})_{pq} = \phi^t(x_{pi})\phi(x_{qj}). \tag{2.7}$$

Let K be a $(M \times M)$ matrix defined on the class elements by $(K_{pq})_{\substack{p=1,\ldots,N \\ q=1,\ldots,N}}$, where (K_{pq}) is a matrix composed of dot product in the feature space F:

$$K = (K_{pq})_{\substack{p=1,\ldots,N \\ q=1,\ldots,N}} \quad \text{where} \quad K_{pq} = (k_{ij})_{\substack{i=1,\ldots,n_p \\ j=1,\ldots,n_q}}. \tag{2.8}$$

K_{pq} is a $(n_p x n_q)$ matrix, and K is a $(M \times M)$ symmetric matrix such that $K_{pq}^t = K_{pq}$. We also introduce the matrix:

$$W = (W_l)_{l=1,\ldots,N}, \tag{2.9}$$

where W_l is a $(n_l x n_l)$ matrix with terms all equal to: $1/n_l$. W is a $(M \times M)$ block diagonal matrix.

In the next section, we formulate the GDA method in the feature space F using the definition of the covariance matrix V (see equation 2.6), the classes covariance matrix B (see equation 2.4), and the matrices K (see equation 2.8) and W (see equation 2.9).

3 GDA Formulation in Feature Space

LDA is a standard tool for classification based on a transformation of the input space into a new one. The data are described as a linear combination of the new coordinate values, called principal components, which represent the discriminant axis. For the common LDA (Fukunaga, 1990), the classical criteria for class separability are defined by the quotient between the interclasses inertia and the intraclasses inertia. This criteria should be larger when the interclasses inertia is larger and the intraclasses inertia is smaller. It was shown that this maximization is equivalent to eigenvalue resolution (Fukunaga, 1990) (see appendix A). Assuming that the classes have a multivariate gaussian distribution, each observation can be assigned to the class having the maximum posterior probability using the Mahalanobis distance.

Using kernel functions, we generalize LDA to the case where in the transformed space, the principal components are nonlinerally related to the input variables. The kernel operator K allows the construction of nonlinear decision function in the input space that is equivalent to linear decision function in the feature space F. As such for the LDA, the purpose of the GDA method is to maximize the interclasses inertia and minimize the intraclasses inertia. This maximization is equivalent to the following eigenvalue resolution: we have to find eigenvalues λ and eigenvectors ν, solutions of the equation

$$\lambda V\nu = B\nu. \tag{3.1}$$

The largest eigenvalue of equation 3.1 gives the maximum of the following quotient of the inertia (see appendix A):

$$\lambda = \frac{\nu^t B\nu}{\nu^t V\nu}. \tag{3.2}$$

Because the eigenvectors are linear combinations of F elements, there exist coefficients α_{pq} $(p = 1, \ldots, N; q = 1, \ldots, n_p)$ such that

$$\nu = \sum_{p=1}^{N} \sum_{q=1}^{n_p} \alpha_{pq} \phi(x_{pq}). \tag{3.3}$$

All solutions ν with nonzero eigenvalue lie in the span of $\phi(x_{ij})$.

The coefficient vector $\alpha = (\alpha_{pq})_{p=1,\ldots,N;q=1,\ldots,n_p}$ can be written in a condensed way as $\alpha = (\alpha_p)_{p=1,\ldots,N}$, where $\alpha_p = (\alpha_{pq})_{q=1,\ldots,n_p}$, α_p is the coefficient of the vector ν in the class p.

We show in appendix B that equation 3.2 is equivalent to the following quotient:

$$\lambda = \frac{\alpha^t KWK\alpha}{\alpha^t KK\alpha}. \tag{3.4}$$

This equation, developed in appendix B, is obtained by multiplying equation 3.1 by $\phi^t(x_{ij})$, which makes it easy to rewrite in a matrix form. Equation 3.1 has the same eigenvector as Schölkopf et al. (1998):

$$\lambda\phi^t(x_{ij})V\nu = \phi^t(x_{ij})B\nu. \tag{3.5}$$

We then express equation 3.5 by using the powerful idea of dot product (Aizerman et al., 1964; Boser et al., 1992) between the mapped pattern defined by the matrices K in F without having to carry out the map ϕ. We rewrite the two terms of the equality 3.5 in a matrix form using the matrices K and W, which gives equation 3.4 (see appendix B).

The next section resolves the eigenvector system (see equation 3.4), which requires an algebraic decomposition of the matrix K.

4 Eigenvalue Resolution

Let us use the eigenvectors decomposition of the matrix K,

$$K = U\Gamma U^t.$$

Here, we consider Γ the diagonal matrix of nonzero eigenvalues and U the matrix of normalized eigenvectors associated to Γ. Thus Γ^{-1} exists. U is an orthonormal matrix, that is,

$$U^tU = I, \quad \text{where } I \text{ is the identity matrix.}$$

Substituting K in equation 3.4, we get:

$$\lambda = \frac{(\Gamma U^t\alpha)^t U^t W U(\Gamma U^t\alpha)}{(\Gamma U^t\alpha)^t U^t U(\Gamma U^t\alpha)}.$$

Let us proceed to variable modification using β such that:

$$\beta = \Gamma U^t\alpha. \tag{4.1}$$

Substituting in the latter formula, we get

$$\lambda = \frac{\beta^t U^t W U\beta}{\beta^t U^t U\beta}. \tag{4.2}$$

Therefore we obtain

$$\lambda U^t U\beta = U^t W U\beta.$$

As U is orthonormal, the latter equation can be simplified and gives

$$\lambda\beta = U^t W U\beta, \tag{4.3}$$

for which solutions are to be found by maximizing λ. For a given β, there exists at least one α satisfying equation 4.1 in the form $\alpha = U\Gamma^{-1}\beta$. α is not unique.

Thus the first step of the system resolution consists of finding β according to equation 4.3, which corresponds to a classical eigenvector system resolution. Once β are calculated, we compute α.

Note that one can achieve this resolution by using other decomposition of K or other diagonalization method. We refer to the QR decomposition of K (Wilkinson & Reinsch, 1971), which allows working in a subspace that simplifies the resolution.

The coefficients α are normalized by requiring that the corresponding vectors ν be normalized in F: $\nu^t\nu = 1$.

Using equation 3.2,

$$\nu^t\nu = \sum_{p=1}^{N}\sum_{q=1}^{n_p}\sum_{l=1}^{N}\sum_{h=1}^{n_l}\alpha_{pq}\alpha_{lh}\phi^t(x_{pq})\phi(x_{lh}) = 1$$

$$\nu^t\nu = \sum_{p=1}^{N}\sum_{l=1}^{N}\alpha_p^t K_{pl}\alpha_l = 1$$

$$\nu^t\nu = \alpha^t K\alpha = 1. \tag{4.4}$$

The coefficients α are divided by $\sqrt{\alpha^t K\alpha}$ in order to get normalized vectors ν.

Knowing the normalized vectors ν, we then compute projections of a test point z by

$$\nu^t\phi(z) = \sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}k(x_{pq}, z). \tag{4.5}$$

GDA procedure is summarized in the following steps:

1. Compute the matrices K (see equations 2.7 and 2.8) and W (see equation 2.9).
2. Decompose K using eigenvectors decomposition.
3. Compute eigenvectors β and eigenvalues of the system 4.3.
4. Compute eigenvectors ν using α (see equation 3.3) and normalize them (see equation 4.4).
5. Compute projections of test points onto the eigenvectors ν (see equation 4.5).

5 Experiments

In this section, two sets of simulated data are studied, and then the results are confirmed on Fisher's iris data (Fisher, 1936) and the seed classification.

The types of simulated data are chosen in order to emphasize the influence of the kernel function. We have used a polynomial kernel of degree d and a gaussian kernel to solve the corresponding classification problem. Other kernel forms can be used, provided that they fulfill the Mercer's theorem (Vapnik, 1995; Schölkopf et al., 1998). Polynomial and gaussian kernels are among the classical kernels used in the literature:

Polynomial kernel using a dot product: $k(x, y) = (x, y)^d$, where d is the polynomial degree.

Gaussian kernel: $k(x, y) = \exp(-\frac{\|x-y\|^2}{\sigma})$, where the parameter σ has to be chosen.

These kernel functions are used to compute K matrix elements: $k_{ij} = k(x_i, x_j)$.

5.1 Simulated Data.

5.1.1 Example 1: Separable Data. Two 2D classes are generated and studied in the feature space obtained with different type of kernel function. This example aims to illustrate the behavior of the GDA algorithm according to the choice of the kernel function.

For the first class (class1), a set of 200 points (x, y) is generated as in the following:

X is a normal variable with a mean equal to 0 and a standard variation equal to $\sqrt{2}$: $X \sim N(0, \sqrt{2})$.

Y is generated according to the following variable: $Y_i = X_i^* X_i + N(0, 0.1)$.

The second class (class2) corresponds to 200 points (x, y), where:

X is a variable such that $X \sim N(0, 0.001)$.

Y is a variable such that $Y \sim N(2, 0.001)$.

Note that the variables X and Y are independent here. Twenty examples by class are randomly chosen to compute the decision function. Both data and the decision function are shown in Figure 1.

We construct a decision function corresponding to a polynomial of degree 2. Suppose that the input vector $x = (x_1, x_2, \ldots, x_t)$ has t components, where t is termed the dimensionality of the input space. The feature space F has $\frac{t(t+1)}{2}$ coordinates of the form $\phi(x) = (x_1^2, \ldots, x_t^2, x_1x_2, \ldots, x_ix_j, \ldots)$ (Poggio, 1975; Vapnik, 1995). The separating hyperplane in F space is a second-degree polynomial in the input space. The decision function is computed on the learning set by finding a threshold such that the projection curves (see Figure 1b) are well separated. Here, the chosen threshold corresponds to the value 0.7. The polynomial separation is represented for the whole data in Figure 1a.

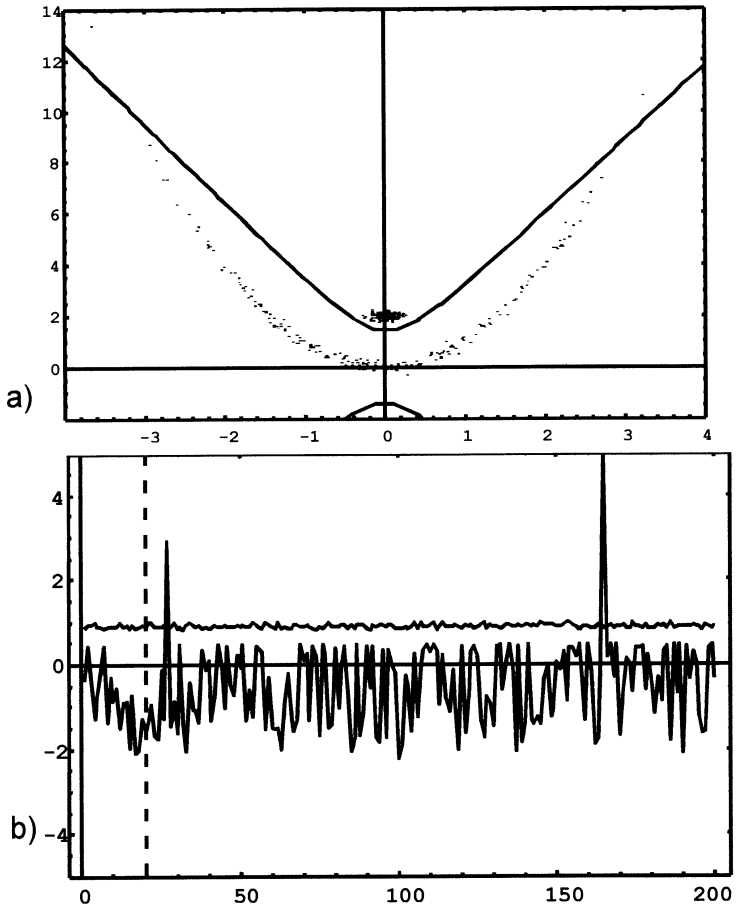

Figure 1: (a) The decision function for the two classes using the first discriminant axis. In the input space, the decision function is computed using a polynomial kernel type with $d = 2$. (b) Projections of all examples on the first axis with an eigenvalue λ equal to 0.765. The dotted line separates the learning examples from the others. The x-axis represents the samples number, and the y-axis is the value of the projection.

Notice that the nonlinear GDA produces a decision function reflecting the structure of the data. As for the LDA method, the maximal number of principal components with nonzero eigenvalues is equal to the number of classes minus one (Fukunaga, 1990). For this example, the first axis is sufficient to separate the two classes of the learning set.

It can be seen from Figure 1b that the two classes can clearly be separated using one axis, except for two examples, where the curves overlap. The two misclassified examples do not belong to the learning set. The vertical dotted line indicates the 20 examples of the learning set of each class. We can observe that almost all of the examples of class2 are projected on one point.

In the following, we give the results using a gaussian kernel. As previously, the decision function is computed on the learning set and represented for the data in Figure 2a.

In this case, all examples are well separated. When projecting the examples on the first axis, we obtain the curves given in Figure 2b, which are well-separated lines. The positive line corresponds to class2 and the negative one to class1. The line of the threshold zero separates all the learning examples as well as all the testing examples. The corresponding separation in the input space is an ellipsoid (see Figure 2a).

5.1.2 Example 2: Nonseparable Data. We consider two overlapped clusters in two dimensions, each cluster containing 200 samples. For the first cluster, samples are uniformly located on a circle of radius of 3. A normal noise with a variance of 0.05 is added to the X and Y coordinates. For the second cluster, the X and Y coordinates follow a normal distribution with a mean vector

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

and a covariance matrix

$$\begin{pmatrix} 5.0 & 4.9 \\ 4.9 & 5.0 \end{pmatrix}.$$

This example will illustrate the behavior of the algorithm on nonseparated data, and the classification results will be compared to SVM results. Therefore, 200 samples of each cluster are used for the learning step and 200 for the testing step. The GDA is performed using a gaussian kernel operator with a sigma equal to 0.5. The decision function and the data are represented on the Figure 3.

To evaluate the classification performance, we use the Mahalanobis distance to assign samples to the classes. The percentage of correct classification for the learning set is 98%, and for the testing set it is equal to 93.5%. The SVM classifier of free Matlab software (Gunn, 1997) has been used to classify these data with the same kernel operator and the same value of sigma (sigma $= 0.5$ and $C = \infty$). The percentage of correct classification for the learning set is 99%, and for the testing set it is 83%. By performing the parameter C of the SVM classifier with a gaussian kernel, the best results obtained (with sigma $= 1$ and $C = 1$) are 99% on the learning set and 88% on the testing set.

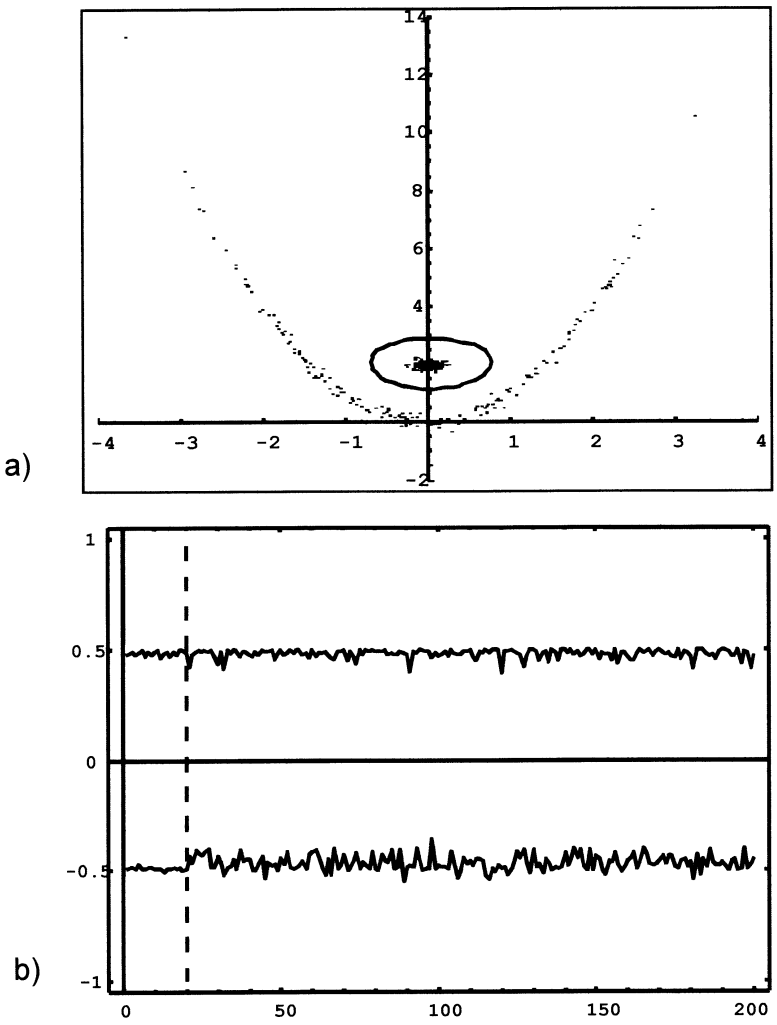

Figure 2: (a) The decision function on the data using a gaussian kernel with $\sigma = 1$. (b) The projection of all examples on the first discriminant axis with an eigenvalue λ equal to 0.999. The x-axis represents the samples number, and the y-axis is the value of the projection.

5.2 Fisher's Iris Data. The iris flower data were originally published by Fisher (1936), for example, in discriminant analysis and cluster analysis. Four parameters—sepal length, sepal width, petal length, and petal

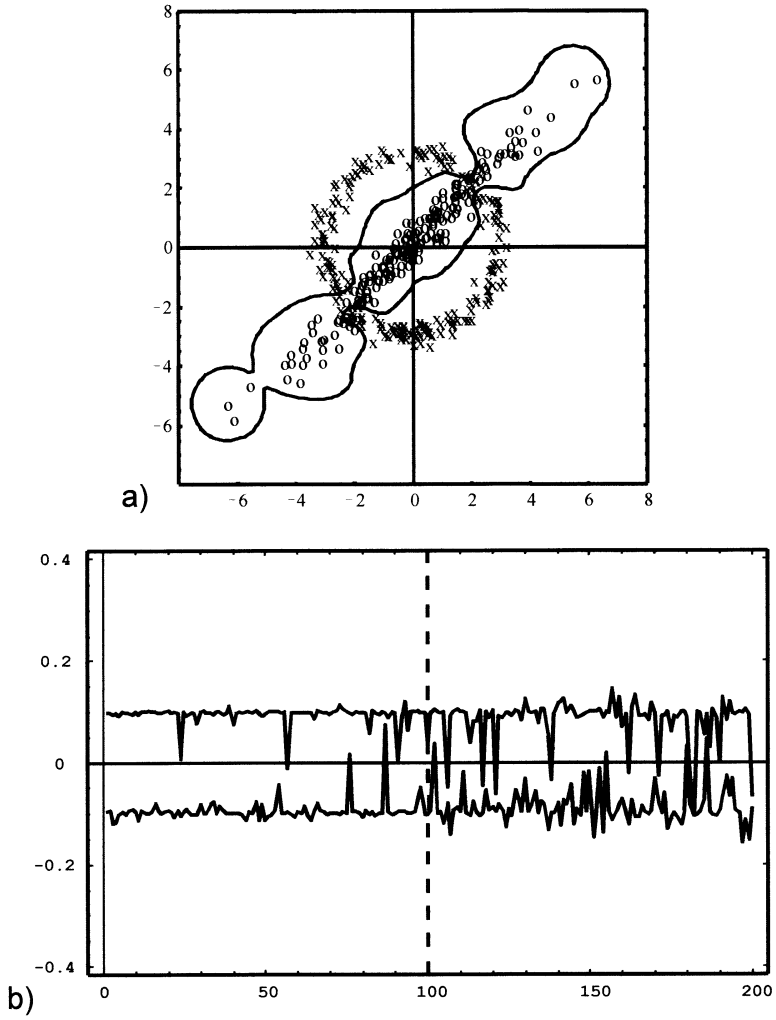

Figure 3: (a) 200 samples of the first cluster are represented by crosses and 200 samples of the second cluster by circles. The decision function is computed on the learning set using a gaussian kernel with sigma = 0.5. (b) Projections of all samples on the first axis with an eigenvalue equal to 0.875. The dotted vertical line separates the learning sample from the testing samples.

width—were measured in millimeters on 50 iris specimens from each of three species: *Iris setosa, I. versicolor,* and *I. virginica*. The set of data thus contains 150 examples with four dimensions and three classes.

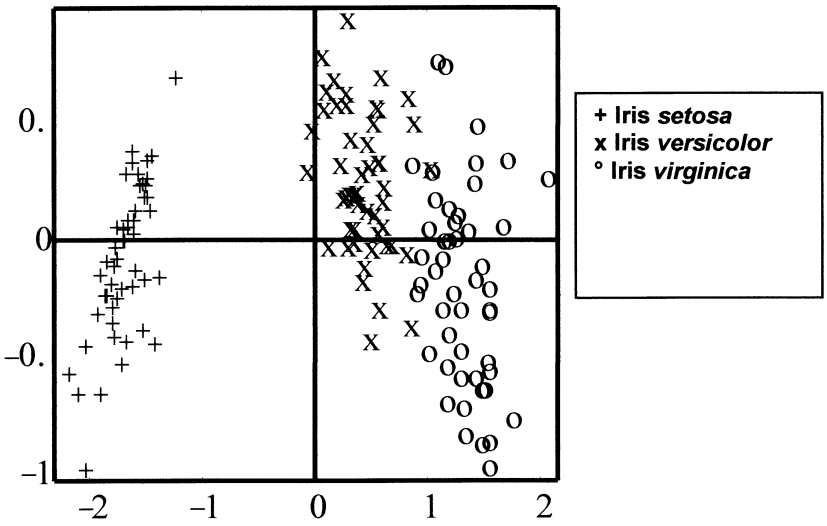

Figure 4: The projection of iris data on the first two axes using LDA. LDA is derived from GDA and associated with a polynomial kernel with degree one ($d = 1$).

One class is linearly separable from the two other; the latter are not linearly separable from each other. For the following tests, all iris examples are centered. Figure 4 shows the projection of the examples on the first two discriminant axes using the LDA method, a particular case of GDA when the kernel is a polynomial with degree one.

5.2.1 Relation to Kernel PCA. Kernel PCA proposed by Schölkopf et al. (1998) is designed to capture the structure of the data. The method reduces the sample dimension in a nonlinear way for the best representation in lower dimensions, keeping the maximum of inertia. However, the best axis for the representation is not necessarily the best axis for the discrimination. After kernel PCA, the classification process is performed in the principal component space. The authors propose different classification methods to achieve this task. Kernel PCA is a useful tool for unsupervised and nonlinear problems for feature extraction. In the same manner, GDA can be used for supervised and nonlinear problems for feature extraction and classification. Using GDA, one can find a reduced number of discriminant coordinates that are optimal for separating the groups. With two such coordinates, one can visualize a classification map that partitions the reduced space into regions.

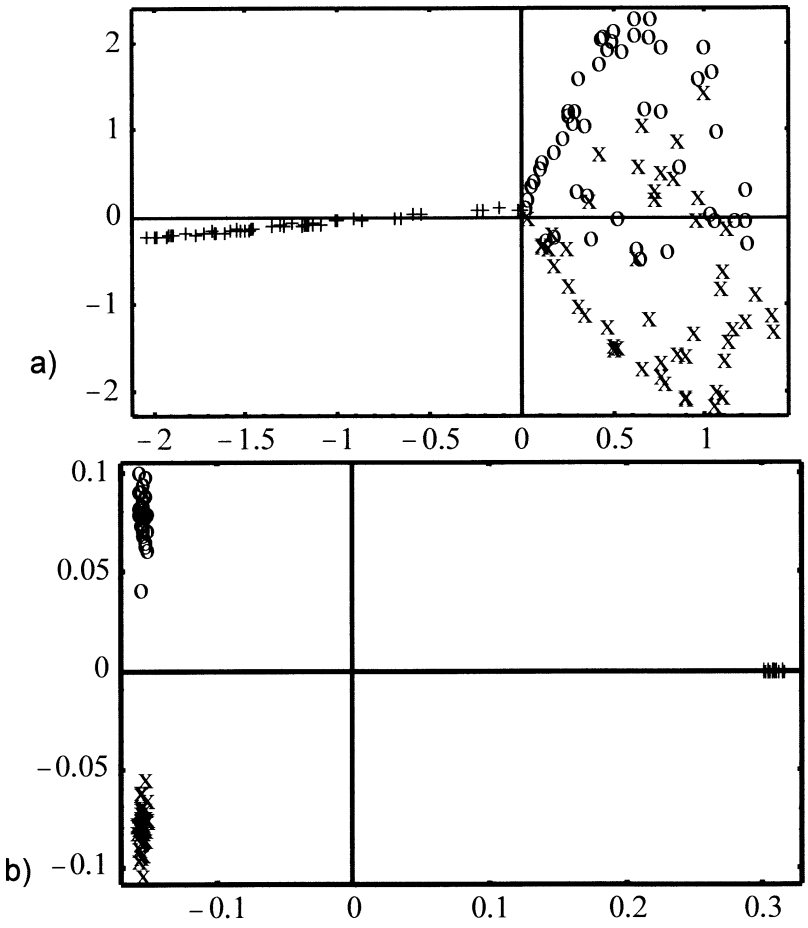

Figure 5: (a) The projection of the examples on the first two axes using nonlinear kernel PCA with a gaussian kernel and $\sigma = 0.7$. (b) The projection of the examples on the first two axes using the GDA method with a gaussian kernel $\sigma = 0.7$.

Kernel PCA and GDA can produce a very different representation, which is highly dependent on the structure of the data. Figure 5 shows the results of applying both the kernel PCA and GDA to the iris classification problem using the same gaussian kernel with $\sigma = 0.7$. The projection on the first two axes seems to be insufficient for kernel PCA to separate the classes; more than two axes will certainly improve the separation. With two axes, GDA algorithm produces better separation of these data because of the use of the interclasses inertia.

Table 1: Comparison of GDA and Other Classification Methods for the Discrimination of Three Seed Species: *Medicago sativa* L. (lucerne), *Melilotus sp.*, and *Medicago lupulina* L.

	Percentage of Correct Classification	
Method	Learning	Testing
k-nearest neighbors	81.7	81.1
Linear dicriminant analysis (LDA)	72.8	67.3
Probabilistic neural network (PNN)	100	85.6
Generalized dicriminant analysis (GDA) gaussian kernel (sigma $= 0.5$)	100	85.1
Nonlinear Support vectors machines gaussian kernel (sigma $= 0.7$, $C = 1000$)	99	85.2

As can be seen from Figure 5b, the three classes are well separated: each is nearly projected on one point, which is the center of gravity. Note that the first two eigenvalues are equal to 0.999 and 0.985.

In addition, we assigned the test examples to the nearest class according to the Mahalanobis distance and using the prior probability of each class. We apply the assignment procedure with the leave-one-out test method. We measured the percentage of correct classification. For GDA, the result is equal to 95.33% of correct classification. This percentage can be compared to those of radial basis function (RBF) network (96.7%) and the multilayer perceptron (MLP) network (96.7%) (Gabrijel & Dobnikar, 1997).

5.3 Seed Classification. Seed samples were supplied by the National Seed Testing Station of France. The aim is to perform seed classification methods in order to help analysts for the successful operation of national seed certification. Seed characteristics are extracted using a vision system and image analysis software. Three seed species are studied: *Medicago sativa* L. (lucerne), *Melilotus sp.*, and *Medicago lupulina* L. These species present the same appearance and are difficult for analysts to identify (see Figure 6).

In all, 224 learning seeds and 150 testing seeds were placed in random positions and orientations in the field of the camera. Each seed was represented by five variables extracted from image seeds and describing their morphology and texture. Different classification methods were compared in terms of the percentage of correct classification. The results are summarized in Table 1.

k-nearest neighbors classification was performed with 15 neighbors and gives better results than the LDA method. The SVM classifier was tested with different kernels and for several values of the upper-bound parameter C to relax the constraints. In this case the best results are 99% on the learning set and 85.2% on the testing set for $C = 1000$ and sigma $= 0.7$. For the same

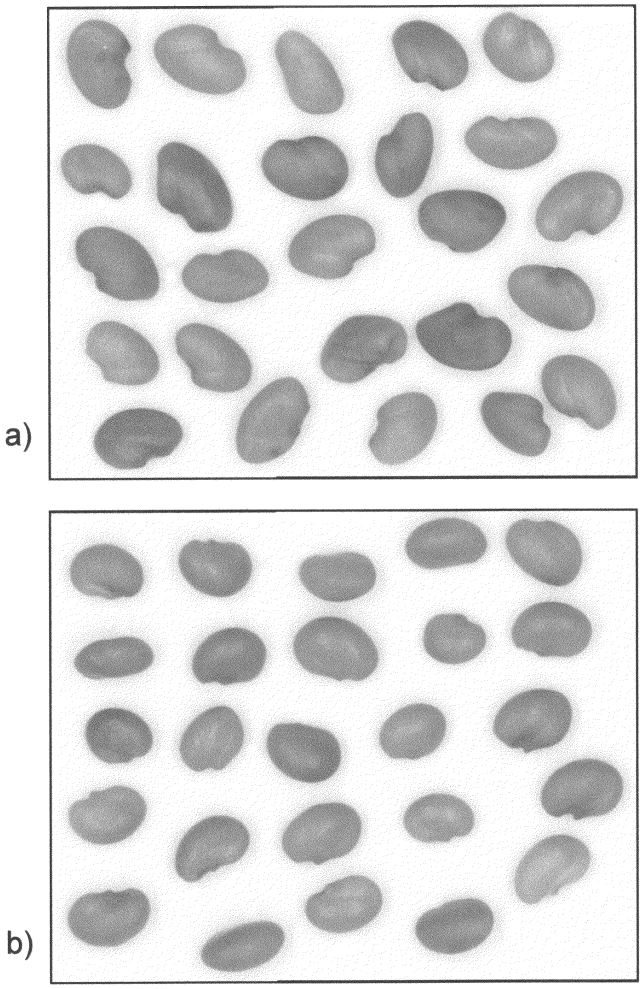

Figure 6: Examples of images of seeds: (a) *Medicago sativa* L. (lucerne). (b) *Medicago lupulina* L.

sigma as GDA (0.5), SVM results are 99% for the learning set and 82.5% for the testing set. The classification results obtained by the GDA method with a gaussian kernel, probabilistic neural network (PNN) (Specht, 1990; Musavi, Kalantri, Ahmed, & Chan, 1993), and SVM classifier are nearly the same. However, the advantage of the GDA method is that it is based on a formula calculation and not on an optimization approximation such as for the PNN classifier or SVM for which the user must choose and adjust some

parameters. Moreover, the SVM classifier was initially developed for two classes, and adapting it to multiclasses is time-consuming.

6 Discussion and Future Work

The dimensionality of the feature space is huge and depends on the size of the input space. A function that successfully separates the learning data may not generalize well. One has to find a compromise between the learning and the generalization performances. It was shown for SVM that the test error depends on only the expectation of the number of support vectors and the number of learning examples (Vapnik, 1995) and not on the dimensionality of the feature space. Currently we are seeking to establish the relationship between GDA resolution and SVM resolution. We can improve generalization, accuracy, and speed by using the wide studies of SVM technique (Burges & Schölkopf, 1997; Schölkopf et al., 1998). Nevertheless, the fascinating idea of using a kernel approach is that we can construct an optimal separating hyperplane in the feature space without considering this space in an explicit form. We have to calculate only the dot product. But the choice of the kernel type remains an open problem. However, the possibility of using any desired kernels (Burges, 1998) allows generalizing classical algorithms. For instance, there are similarities between GDA with a gaussian kernel and PNNs. Like the GDA method, PNN can be viewed as a mapping operator built on a set of input-output observations, but the procedure to define decision boundaries is different for the two algorithms. In GDA, the eigenvalue resolution is used to find a set of vectors that define a hyperplane separation and give a global minimum according to the inertia criterion. In PNN the separation is found by trial-and-error measurement on the learning set. The PNN and more general neural networks always find a local minimum.

The only weakness of the method, as for the SVM, is the need to store and process large matrices, as large as the number of the examples. The largest database on which we applied GDA is a real database with 1500 examples. Independently of the memory storage consumption, the complexity of the algorithm is cubic in the number of the example.

7 Conclusion

We have developed a generalization of discriminant analysis as nonlinear discrimination. We described the algebra formulation and the eigenvalue resolution. The motivation for exploring the algebraic approach was to develop an exact solution, not an approximate optimization. The GDA method gives an exact solution even if some points require further investigation, such as the choice of the kernel function. In terms of classification performance, for the small databases studied here, the GDA method competes with support vector machines and probabilistic neural network classifier.

Appendix A

Given two symmetric matrices A and B with the same size, B is supposed inversible. It shown that (Saporta, 1990) the quotient $\frac{v^tAv}{v^tBv}$ is maximal for v an eigenvector of $B^{-1}A$ associated with the largest eigenvalue λ. Maximizing the quotient requires that the derivative with respect to v vanish:

$$\frac{(v^tBv)(2Av) - (v^tAv)(2Bv)}{(v^tBv)^2} = 0,$$

which implies

$$B^{-1}Av = \left(\frac{v^tAv}{v^tBv}\right)v.$$

v is then an eigenvector of $B^{-1}A$ associated with the eigenvalue $\frac{v^tAV}{v^tBv}$. The maximum is reached for the largest eigenvalue.

Appendix B

In this appendix we rewrite equation 3.5 in a matrix form in order to obtain equation 3.4. We develop each term of equation 3.5 according to the matrices K and W. Using equations 2.6 and 3.2, the left term of 3.5 gives:

$$\begin{aligned} Vv &= \frac{1}{M}\sum_{l=1}^{N}\sum_{k=1}^{n_l}\phi(x_{lk})\phi^t(x_{lk})\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\phi(x_{pq}) \\ &= \frac{1}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\sum_{l=1}^{N}\sum_{k=1}^{n_l}\phi(x_{lk})[\phi^t(x_{lk})\phi(x_{pq})] \end{aligned}$$

$$\begin{aligned} \lambda\phi^t(x_{ij})Vv &= \frac{\lambda}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\phi^t(x_{ij})\sum_{l=1}^{N}\sum_{k=1}^{n_l}\phi(x_{lk})[\phi^t(x_{lk})\phi(x_{pq})] \\ &= \frac{\lambda}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\sum_{l=1}^{N}\sum_{k=1}^{n_l}\left[\phi^t(x_{ij})\phi(x_{lk})\right]\left[\phi^t(x_{lk})\phi(x_{pq})\right]. \end{aligned}$$

Using this formula for all class i and for all its element j, we obtain:

$$\begin{aligned} &\lambda(\phi^t(x_{11}),\ldots,\phi^t(x_{1n_1}),\ldots,\phi^t(x_{ij}),\ldots,\phi^t(x_{N1}),\ldots,\phi^t(x_{Nn_N}))Vv \\ &\quad = \frac{\lambda}{M}KK\alpha. \end{aligned} \tag{B.1}$$

According to equations 2.4, 2.5, and 3.3, the right term of equation 3.5 gives:

$$B\nu = \frac{1}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\phi(x_{pq})\sum_{l=1}^{N} n_l \left[\frac{1}{n_l}\sum_{k=1}^{n_l}\phi(x_{lk})\right]\left[\frac{1}{n_l}\sum_{k=1}^{n_l}\phi(x_{lk})\right]^t$$

$$= \frac{1}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\sum_{l=1}^{N}\left[\sum_{k=1}^{n_l}\phi(x_{lk})\right]\left[\frac{1}{n_l}\right]\left[\sum_{k=1}^{n_l}\phi^t(x_{lk})\phi(x_{pq})\right].$$

$$\phi^t(x_{ij})B\nu = \frac{1}{M}\sum_{p=1}^{N}\sum_{q=1}^{n_p}\alpha_{pq}\sum_{l=1}^{N}\left[\sum_{k=1}^{n_l}\phi^t(x_{ij})\phi(x_{lk})\right]\left[\frac{1}{n_l}\right]\left[\sum_{k=1}^{n_l}\phi^t(x_{lk})\phi(x_{pq})\right].$$

For all class i and for all its elements j, we obtain:

$$\left(\phi^t(x_{11}),\ldots,\phi^t(x_{1n_1}),\ldots,\phi^t(x_{ij}),\ldots,\phi^t(x_{N1}),\ldots,\phi^t(x_{Nn_N})\right)B\nu$$

$$= \frac{1}{M}KWK\alpha. \tag{B.2}$$

Combining equations B.1 and B.2, we obtain:

$$\lambda KK\alpha = KWK\alpha,$$

which is multiplied by α^t to obtain equation 3.4.

Acknowledgments

We are grateful to Scott Barnes (engineer at MEI, USA), Philippe Jard (applied research manager at MEI, USA), and Ian Howgrave-Graham (R&D manager at Landis & Gyr, Switzerland) for their comments on the manuscript for this article. We also thank Rodrigo Fernandez (research associate at the University of Paris Nord) for comparing results using his own SVM classifier software.

References

Aizerman, M. A., Braverman, E. M., & Rozonoér, L. I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control, 25*, 821–837.

Anouar, F., Badran, F., & Thiria, S. (1998). Probabilistic self organizing map and radial basis function. *Journal Neurocomputing, 20*, 83–96.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), *Fifth Annual ACM Workshop on COLT* (pp. 144–152). Pittsburgh, PA: ACM Press.

Burges, C. J. C. (1998). A tutorial on support vector machine for pattern recognition. Support vector web page available online at: http://svm.first.gmd.de.

Burges, C. J. C., & Schölkopf, B. (1997). Improving the accuracy and speed of support vector machines. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Neural information processing systems, 9*. Cambridge, MA: MIT Press.

Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Neural information processing systems, 9*. Cambridge, MA: MIT Press.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annual Eugenics, 7*, 179–188.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition* (2nd ed.). Orlando, FL: Academic Press.

Gabrijel, I., & Dobnikar, A. (1997). Adaptive RBF neural network. In *Proceedings of SOCO '97 Conference* (pp. 164–170).

Gunn, S. R. (1997). *Support vector machines for classification and regression* (Tech. rep.). Image Speech and Intelligent Systems Research Group, University of Southampton. Available online at: http://www.isis.ecs.soton.ac.uk/resource/svminfo/.

Hastie, T., Tibshirani, R., & Buja, A. (1993). *Flexible discriminant analysis by optimal scoring* (Res. Rep.). AT&T Bell Labs.

Kohonen, T. (1994). *Self-organizing maps*. New York: Springer-Verlag.

Musavi, M. T., Kalantri, K., Ahmed, W., & Chan, K. H. (1993). A minimum error neural network (MNN). *Neural Networks, 6*, 397–407.

Poggio, T. (1975). On optimal nonlinear associative recall. *Biological Cybernetics, 19*, 201–209.

Saporta, G. (1990). *Probabilites, analyse des données et statistique*. Paris: Editions Technip.

Schölkopf, B. (1997). *Support vector learning*. Munich: R. Oldenbourg Verlag.

Schölkopf, B., Smola, A., & Müller, K. R. (1996). *Nonlinear component analysis as a kernel eigenvalue problem* (Tech. Rep. No. 44). MPI fur biologische kybernetik.

Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation, 10*, 1299–1319.

Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks, 3*(1), 109–118.

Vapnik, V. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.

Vapnik, V., Golowich, S. E., & Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Neural information processing systems, 9*. Cambridge, MA: MIT Press.

Wilkinson, J. H., & Reinsch, C. (1971). *Handbook for automatic computation*, Vol. 2: *Linear algebra*. New York: Springer-Verlag.

Recent References Since the Article Was Submitted

Jaakkola, T. S., & Haussler, D. (1999). Exploiting generative models in discriminative classifiers. In M. S. Kearns, S. A. Solla, & D. A. Cohn (Eds.), *Advances in neural information processing systems, 11*. Cambridge, MA: MIT Press.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K. R. (1999). Fisher discriminant analysis with kernels. In *Proc. IEEE Neural Networks for Signal Processing Workshop, NNSP*.

Received December 4, 1998; accepted October 6, 1999.