Discriminant Subspace Learning Based on Support Vectors Machines

Nikolaos Pitelis[1] and Anastasios Tefas[2]

[1] School of Electronic Engineering and Computer Science,
Queen Mary, University of London, UK
nikolaos.pitelis@eecs.qmul.ac.uk
[2] Department of Informatics, Aristotle University of Thessaloniki, Greece
tefas@aiia.csd.auth.gr

Abstract. A new method for dimensionality reduction and feature extraction based on Support Vector Machines and minimization of the within-class data dispersion is proposed. An iterative procedure is proposed that successively applies Support Vector Machines on perpendicular subspaces using the deflation transformation in such a way that the within-class variance is minimized. The proposed approach is proved to be a successive SVM using deflation kernels. The normal vectors of the successive hyperplanes contain discriminant information and they can be used as projection vectors for feature extraction and dimensionality reduction of the data. Experiments on various datasets are conducted in order to highlight the superior performance of the proposed algorithm.

1 Introduction

In pattern recognition and machine learning problems with high-dimensional data have always been difficult to cope with. That is, the so-called "curse of dimensionality", which constitutes motivation for the development of dimensionality reduction methods. Simple classification algorithms which are very commonly used in a variety of disciplines, like *k-Nearest Neighbor* (KNN) [1] or *Nearest Centroid* (NC) [2], favor greatly when they have to treat the same problem in a lower-dimensional space, especially when it is redundant.

The benefits lie in reducing computational complexity, since the size of the problem is reduced, and improving classification accuracy. The first gives the possibility to deal with more complex problems that cannot be treated in their original form. In order for the latter to be succeeded, the dimensionality reduction has to take place in such a way that will augment discriminant information and remove information that does not contribute discriminability, e.g noise.

A closely related term to dimensionality reduction is feature extraction, which entails the transformation of the data from the high-dimensional space to the lower-dimensional one. This transformation can be either linear or non-linear and although linear transformations have a more solid mathematical background, non-linear transformations, which are usually extensions of previously proposed linear ones, are usually more powerful. These non-linear generalizations are usually achieved using the *kernel trick* [3], which gives us the opportunity to compute

P. Perner (Ed.): MLDM 2012, LNAI 7376, pp. 198–212, 2012.

only the dot products of the inpute patterns, rather to explicitly compute the mapping, i.e. their projection onto a very high-dimensional space.

Feature extraction can also be used for visualization tasks so as to get better understanding and an overview of a problem. In the framework of this paper we are interested both in visualization and classification tasks. In the following we shortly describe the most commonly used dimensionality reduction techniques that are related to our proposed method.

1.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) [4], also known as Karhunen-Loeve transformation, was fisrt developed by Pearson [5] (1901) and Hotelling [6] (1933). It is one of the most widely used dimensionality reduction technique in problems as data compression and clustering, pattern recognition and visualization. The main idea is to reduce the dimensionality of a data population trying to keep its spatial characteristics. This is achieved with a linear transformation to the space of principal components, which are ordered in such a way that the first few retain most of the data variation. The principal components are obtained by performing eigenanalysis of the data covariance matrix.

More specifically, if $\boldsymbol{A}_k$ is the matrix of the k eigenvectors that correspond to the k largest eigenvalues of the covariance matrix and $\boldsymbol{X}$ is the initial data matrix, then the transformed data of dimensionality k is given by $\boldsymbol{Z} = \boldsymbol{A}_k\boldsymbol{X}$. PCA has been generalized into kernel-PCA [7] using the kernel trick. Since PCA is an unsupervised learning method, i.e. class label information is not taken into account, it is not always suitable for classification tasks.

1.2 Linear Discriminant Analysis (LDA)

On the contrary to PCA, *Linear Discriminant Analysis* (LDA), [8], also known as *Fisher's Discriminant Analysis* (FDA or FLDA), is a supervised learning technique, which exploits the class label information in order to maximize the classes discriminality in the extracted space. This is achieved by maximizing Fisher's discriminant ratio, that is, the ratio of between-class variance to within-class variance. For a training set of d-dimensional samples $\boldsymbol{x}_i, i = 1, \ldots, N$ that belong to two classes these notions are expressed by the following quantities

$$\begin{aligned} \boldsymbol{S}_i &= \sum_{\boldsymbol{x}\in\omega_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T, \\ \boldsymbol{S}_W &= \boldsymbol{S}_1 + \boldsymbol{S}_2 \quad \text{and} \\ \boldsymbol{S}_B &= (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \end{aligned} \tag{1}$$

By $\boldsymbol{\mu}_i$ we denote the mean value of class ω_i. We call $\boldsymbol{S}_W$ *within-class scatter matrix* and $\boldsymbol{S}_B$ *between-class scatter matrix*. The quantity that LDA seeks to maximize is defined as

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w}}, \tag{2}$$

where $\boldsymbol{w}$ is the projection vector that transforms the data to the one-dimensional subspace. If the number of classes is more than two, then the reduced dimensionality can be at most equal to the number of classes minus one. The performance of LDA is optimal provided that the data distributions are normal for all classes with the same covariance matrix. LDA was also extended for the non-linear case in Kernel-FDA [9], similarly to PCA, using the kernel trick.

1.3 Margin Maximizing Discriminant Analysis (MMDA)

Margin Maximizing Discriminant Analysis (MMDA) [10] investigated the possibility of projecting the input data onto the normal of a hyperplane that separates two classes in a binary problem. This hyperplane should provide good generalization for future data and make no assumptions regarding the distribution of the input patterns. The authors proposed a deflation approach to be able to perform this process in subsequent orthogonal subspaces, by projecting onto the space spanned by the normal of such a margin maximizing hyperplane. The first hyperplane is obtained solving the *Maximum Margin Separation* (MMS) problem, which is expressed as a quadratic programming problem:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}||\boldsymbol{w}||_2^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{subject to} \quad y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned} \tag{3}$$

The resulting weighting vector of the hyperplane is normalized and used for projecting the data by $\boldsymbol{x}_i' = \boldsymbol{x}_i - (\boldsymbol{w}^T\boldsymbol{x}_i)\boldsymbol{w}$. Then, problem 3 is solved again for the projected data.

1.4 Proposed Approach

In this paper we propose a novel supervised learning technique that seeks to exploit *Support Vector Machines* (SVMs) in a dimensionality reduction scheme. More specifically we intend to use the discriminant information contained in the resulting hyperplane of SVMs to perform feature extraction and the corresponding normal vector of the hyperplane can be used as projection vector. Thus, the first step of the proposed method is the standard SVM optimization that generates the first dimension/feature. In order to be able to extract additional discriminant information we adopt a deflation procedure similar to MMDA. On the same time, inspired by the maximization of Fisher's discriminant ratio, we desire to minimize the within-class variance similarly to [11] and [12]. This results to the definition of a new optimization problem incorporating both the deflation procedure and the within-class variance minimization. This approach can be regarded as a modification of the standard SVMs optimization, employing a deflation kernel.

The novelty of our work lies in three different aspects. The first is the idea of combining SVMs for maximizing the between-class margin and Fisher's discriminant ratio for minimizing the within class variance in one dimensionality

reduction technique. The second is the iterative generation of successive orthonormal projections onto deflated subspaces, according to this criterion, for feature extraction. And the third is the incorporation of the within-class variance minimization and the deflation procedure in the SVMs optimization, using the kernel trick.

The manuscript is organized as follows: The proposed approach is described in Sec. 2. In Sec. 2.1, we discuss in detail the deflation procedure that we adopt and in Sec. 2.2 we show how the deflation of the within-class scatter matrix can be included in the same procedure. The modified optimization problem is presented in Sec. 2.3 and in Sec. 2.4 we show how this problem can be efficiently solved using the kernel trick. The way we perform the feature extraction and the final form of the algorithm are presented in Sec.2.5. In Sec. 3.1 we demonstrate the visualization capability of our method and in Sec. 3.2 we present the experimental results for classification tasks. Finally, conclusions are drawn in Sec. 4.

2 Definition and Derivation of the Problem

In the proposed approach, our goal is to use the information regarding the distribution of the data in space, which is contained in the resulting hyperplane of a classification task with SVMs minimizing in parallel the within-class variance. Moreover, we want to do that in an iterative way so as each iteration of the procedure will provide us with a new feature, which will contain additional discriminant information for our data with respect to the preceding steps. In order to achieve that we need to apply the SVMs in successive subspaces, which are pairwise perpendicular. The proposed method is called *Within Support Vector Discriminant Analysis* (WSVDA) and the algorithm can be overseen in Table 2.

Table 1. The main steps of the iterative procedure of WSVDA

1: compute the within scatter matrix $\boldsymbol{S}_W$ for the data
2: solve SVMs for the data minimizing $\boldsymbol{S}_W$
3: compute weighting vector $\boldsymbol{w}$
4: compute projection matrix $\boldsymbol{P}$ using the normalized weighting vector of previous step
5: deflate the data along the direction of $\boldsymbol{w}$
6: iterate from step 1 to step 5 for as many times as the desired reduced dimensionality
7: use the normalized weighting vectors $\boldsymbol{w}$ for feature extraction

2.1 Deflation Procedure

Let us present this idea with a simple example. If we think of a three-dimensional example of a binary problem, the resulting hyperplane (actually a plane) of linear SVMs would be as depicted in Fig. 1. The projection of the data onto the hyperplane is additionally a transformation to a space perpendicular to the initial one. Consequently if we apply SVMs to these transformed data, projected

onto the hyperplane, the resulting hyperplane (actually a line) will be perpendicular to the initial one. That means that the two hyperplanes contain exclusive information regarding the data distribution. The orthogonality property stands for the corresponding normal vectors of the hyperplanes, which can be used for the feature extraction.

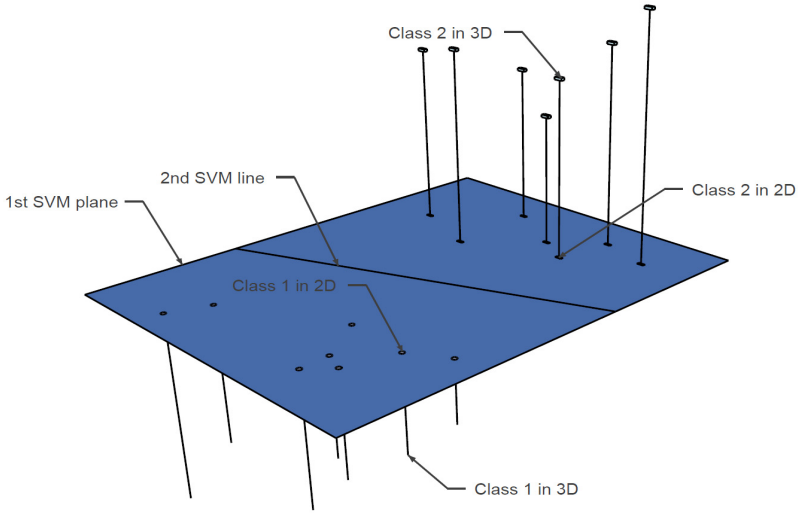

Fig. 1. The resulting hyperplanes of linear SVMs for a three-dimensional example of a binary problem. The three-dimensional data are projected onto the plane, which is the decision surface when the three-dimensional data are the input to the SVMs. The deflated two-dimensional data are the input to the SVMs in the second iteration of the method, resulting to a separating line. Best viewed in color.

Suppose the training set with finite number of elements $\boldsymbol{x}_i, i = 1, \ldots, N$, of dimensionality d, which can be separated into two different classes ω_1 and ω_2. The corresponding labels for these training samples are denoted by y_i with a value equal to 1, if $\boldsymbol{x}_i \in \omega_1$ or -1 if $\boldsymbol{x}_i \in \omega_2$. We also use the notation $\boldsymbol{X}$ for the data matrix, which contains the vectors $\boldsymbol{x}_i$ in its columns, i.e. $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$.

In order to project the data onto the successive hyperplanes we use a deflation transformation algorithm similar to the one in [13], which is used for deflating the data in the space of principal components. Similarly, if $\boldsymbol{w}^k$ is the normal vector of the hyperplane in iteration k of the procedure, then $\boldsymbol{P}_{\boldsymbol{w}^k} = \boldsymbol{I}_{d\times d} - \boldsymbol{w}^k \boldsymbol{w}^{kT}$, where $\boldsymbol{P}_{\boldsymbol{w}^k}$ is the projection matrix along the direction of vector $\boldsymbol{w}^k$ and $\boldsymbol{I}_{d\times d}$ is the identity matrix of dimension d. It is important to mention here, that the weighting vector $\boldsymbol{w}^k$, which is the result of SVMs in our algorithm, has to be normalized before used for the deflation process. Consequently, the data matrix $\boldsymbol{X}$ can be deflated along the direction of $\boldsymbol{w}^k$, that is, projected onto the hyperplane of iteration k, by multiplying it with the corresponding projection matrix $\boldsymbol{P}_{\boldsymbol{w}^k}$, $\boldsymbol{X}^k = \boldsymbol{P}_{\boldsymbol{w}^k} \boldsymbol{X}$.

Since in each iteration we transform the data to a subspace having removed a dimension, the deflation should be done on all the directions of the normal vectors of the previously computed hyperplanes. This multiple deflation can be done in a successive way: $\boldsymbol{X}^1 = \boldsymbol{P}_{\boldsymbol{w}^1}\boldsymbol{X}, \boldsymbol{X}^2 = \boldsymbol{P}_{\boldsymbol{w}^2}\boldsymbol{X}^1, \ldots, \boldsymbol{X}^k = \boldsymbol{P}_{\boldsymbol{w}^k}\boldsymbol{X}^{k-1}$, but it can also be applied on the initial data matrix using the product of all the projection matrices of the previous steps: $\boldsymbol{P}^k = \boldsymbol{P}_{\boldsymbol{w}^1}\boldsymbol{P}_{\boldsymbol{w}^2}\ldots\boldsymbol{P}_{\boldsymbol{w}^k}$. In this way, the matrix $\boldsymbol{X}^k = \boldsymbol{P}^k\boldsymbol{X}$ has been simultaneously deflated along the multiple directions of the normal vectors.

If we express the product of the successive projection matrices using the weighting normal vectors we have

$$\boldsymbol{P}^k = (\boldsymbol{I}_{d\times d} - \boldsymbol{w}^1\boldsymbol{w}^{1T})(\boldsymbol{I}_{d\times d} - \boldsymbol{w}^2\boldsymbol{w}^{2T})\ldots(\boldsymbol{I}_{d\times d} - \boldsymbol{w}^k\boldsymbol{w}^{kT}), \tag{4}$$

it is shown that the order of multiplication has no effect to the final result and because of the orthogonality property we conclude to

$$\boldsymbol{P}^k = \boldsymbol{I}_{d\times d} - \sum_{i=1}^{k} \boldsymbol{w}^k\boldsymbol{w}^{kT}. \tag{5}$$

However, for the implementation task, the first form proves to be numerically more stable. Finally, as a result of the symmetry of all the projection matrices, they are equivalent to their respective transposed matrices, e.g. $\boldsymbol{P}^k = \boldsymbol{P}^{k^T}$.

2.2 Within-Class Variance Deflation

We have already discussed in the previous section what is the input data, i.e. the deflated data to the successive SVMs, but we also need to provide them with the within scatter matrix of the deflated data. In order to avoid the computation of this matrix for the deflated data in each iteration we investigate the possibility of 'projecting' the within scatter matrix onto the subspace of each iteration. Indeed,

$$\begin{aligned}
\boldsymbol{S}_W^k &= \boldsymbol{S}_1^k + \boldsymbol{S}_2^k \\
&= \sum_{\boldsymbol{x}_i^k\in\omega_1} (\boldsymbol{x}_i^k - \boldsymbol{\mu}_1^k)(\boldsymbol{x}_i^k - \boldsymbol{\mu}_1^k)^T + \sum_{\boldsymbol{x}_j^k\in\omega_2} (\boldsymbol{x}_j^k - \boldsymbol{\mu}_2^k)(\boldsymbol{x}_j^k - \boldsymbol{\mu}_2^k)^T \\
&= \sum_{\boldsymbol{x}_i\in\omega_1} \boldsymbol{P}^k(\boldsymbol{x}_i - \boldsymbol{\mu}_1)(\boldsymbol{x}_i - \boldsymbol{\mu}_1)^T\boldsymbol{P}^{kT} + \sum_{\boldsymbol{x}_j\in\omega_2} \boldsymbol{P}^k(\boldsymbol{x}_j - \boldsymbol{\mu}_2)(\boldsymbol{x}_j - \boldsymbol{\mu}_2)^T\boldsymbol{P}^{kT} \\
&= \boldsymbol{P}^k \sum_{\boldsymbol{x}_i\in\omega_1} (\boldsymbol{x}_i - \boldsymbol{\mu}_1)(\boldsymbol{x}_i - \boldsymbol{\mu}_1)^T\boldsymbol{P}^k + \boldsymbol{P}^k \sum_{\boldsymbol{x}_j\in\omega_2} (\boldsymbol{x}_j - \boldsymbol{\mu}_2)(\boldsymbol{x}_j - \boldsymbol{\mu}_2)^T\boldsymbol{P}^k \\
&= \boldsymbol{P}^k(\boldsymbol{S}_1 + \boldsymbol{S}_2)\boldsymbol{P}^k \\
\boldsymbol{S}_W^k &= \boldsymbol{P}^k\boldsymbol{S}_W\boldsymbol{P}^k,
\end{aligned} \tag{6}$$

where $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are the within-class scatter matrices for class ω_1 and ω_2 respectively. Similarly, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are the corresponding mean values.

2.3 Deflated Within-Class Support Vector Machines

According to the aforementioned we can modify the standard SVMs [15] optimization problem and define a new one, which simultaneously will maximize the margin and minimize the within-class variance in deflated subspaces. For the separable case [14] would be expressed as

$$\text{minimize} \quad \boldsymbol{w}^T \boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P} \boldsymbol{w}, \quad \boldsymbol{w}^T \boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P} \boldsymbol{w} > \boldsymbol{0} \tag{7}$$

subject to the separability constraints

$$y_i(\boldsymbol{w}^T \boldsymbol{P} \boldsymbol{x}_i + b) \geq 1, \quad i = 1, \ldots, N. \tag{8}$$

The solution to this problem is given by the saddle point of the Lagrangian

$$\boldsymbol{\mathcal{L}}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \boldsymbol{w}^T \boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P} \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i [y_i(\boldsymbol{w}^T \boldsymbol{P} \boldsymbol{x}_i - b) - 1], \tag{9}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots \alpha_N]^T$ is the vector of Lagrange multipliers. The Karush-Kuhn-Tucker (KKT) conditions [16] imply that for the saddle point

$$\begin{aligned}
\nabla_{\boldsymbol{w}} \boldsymbol{\mathcal{L}}(\boldsymbol{w}_o, b_o, \boldsymbol{\alpha}_o) = \boldsymbol{0} \Leftrightarrow \boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P} \boldsymbol{w}_o &= \frac{1}{2} \sum_{i=1}^{N} \alpha_{i,o} y_i \boldsymbol{P} \boldsymbol{x}_i \\
\frac{\partial}{\partial b} \boldsymbol{\mathcal{L}}(\boldsymbol{w}_o, b_o, \boldsymbol{\alpha}_o) = 0 \Leftrightarrow \sum_{i=1}^{N} \alpha_{i,o} y_i &= 0 \\
y_i(\boldsymbol{w}_o^T \boldsymbol{P} \boldsymbol{x}_i - b_o) - 1 \geq 0, \quad i &= 1, \ldots, N \\
\alpha_{i,o} \geq 0, \quad i &= 1, \ldots, N \\
\alpha_{i,o}[y_i(\boldsymbol{w}_o^T \boldsymbol{P} \boldsymbol{x}_i - b_o) - 1] \geq 0, \quad i &= 1, \ldots, N,
\end{aligned} \tag{10}$$

where subscript o denotes the optimal solution.

The KKT conditions show that the weighting vector is a linear combination of the support vectors in the training set multiplied by the inverse of the 'projection' of matrix $\boldsymbol{S}_W$, that is $\boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P}$. More specifically the optimal weighting vector normal to the separating hyperplane is given by

$$\boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P} \boldsymbol{w}_o = \frac{1}{2} \sum_{i=1}^{N} \alpha_{i,o} y_i \boldsymbol{P} \boldsymbol{x}_i \Leftrightarrow \boldsymbol{w}_o = \frac{1}{2} (\boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P})^{-1} \sum_{i=1}^{N} \alpha_{i,o} y_i \boldsymbol{P} \boldsymbol{x}_i. \tag{11}$$

By replacing (11) into (9) and using the KKT conditions, we obtain the Wolfe-dual problem

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{P} (\boldsymbol{P} \boldsymbol{S}_W \boldsymbol{P})^{-1} \boldsymbol{P} \boldsymbol{x}_j, \tag{12}$$

which is equivalent to the optimization problem

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{H}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha} \\ &\text{subject to} \quad \alpha_i \geq 0, i = 1, \ldots, N \quad \text{and} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \end{aligned} \tag{13}$$

where $\boldsymbol{H}_{ij} = \frac{1}{2} y_i y_j \boldsymbol{x}_i^T \boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{x}_j$ is the ijth element of the Hessian matrix.

The corresponding separating hyperplane is defined by

$$\begin{aligned} g(\boldsymbol{x}) &= sgn(\boldsymbol{w}^T\boldsymbol{P}\boldsymbol{x} + b) \\ &= sgn\left(\frac{1}{2}\sum_{i=1}^{N} \alpha_{i,o}\left(\boldsymbol{x}_i^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{x}\right) + b\right), \end{aligned} \tag{14}$$

where $b_o = \frac{1}{2}\boldsymbol{w}_o^T\boldsymbol{P}(\boldsymbol{x}_i + \boldsymbol{x}_j)$ for any pair of support vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ such that $y_i = 1$ and $y_j = -1$.

In the non-separable case [15], we relax the separability constraints (8) by introducing non-negative slack variables ξ_i, $i = 1, \ldots, N$. The new optimization problem is expressed as

$$\text{minimize} \quad \boldsymbol{w}^T\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P}\boldsymbol{w} + C\sum_{i=1}^{N}\xi_i, \quad \boldsymbol{w}^T\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P}\boldsymbol{w} > \mathbf{0} \tag{15}$$

subject to the separability constraints

$$y_i(\boldsymbol{w}^T\boldsymbol{P}\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N, \tag{16}$$

where by C we denote the cost of violating the constraints, i.e. the cost of misclassification.

The solution to this problem is given by the saddle point of the Lagrangian

$$\boldsymbol{\mathcal{L}}(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}) = \boldsymbol{w}^T\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P}\boldsymbol{w} + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{P}\boldsymbol{x}_i - b) - 1 + \xi_i] - \sum_{i=1}^{N}\beta_i\xi_i, \tag{17}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots \alpha_N]^T$ and $\boldsymbol{\beta} = [\beta_1, \ldots \beta_N]^T$ are the vectors of Lagrange multipliers. The modified Karush-Kuhn-Tucker (KKT) conditions [16] for the non-separable case imply that for the saddle point

$$\begin{aligned}
\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}_o, b_o, \boldsymbol{\alpha_o}, \boldsymbol{\beta_o}, \boldsymbol{\xi_o}) = \mathbf{0} \Leftrightarrow \boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P}\boldsymbol{w}_o = \frac{1}{2}\sum_{i=1}^{N}\alpha_{i,o}y_i\boldsymbol{P}\boldsymbol{x}_i \\
\frac{\partial}{\partial b}\mathcal{L}(\boldsymbol{w}_o, b_o, \boldsymbol{\alpha}_o, \boldsymbol{\beta}_o, \boldsymbol{\xi}_o) = 0 \Leftrightarrow \sum_{i=1}^{N}\alpha_{i,o}y_i = 0 \\
\frac{\partial}{\partial \xi_i}\mathcal{L}(\boldsymbol{w}_o, b_o, \boldsymbol{\alpha}_o, \boldsymbol{\beta}_o, \boldsymbol{\xi}_o) = 0 \Leftrightarrow \beta_{i,o} = C - \alpha_{i,o} \\
y_i(\boldsymbol{w}_o^T\boldsymbol{P}\boldsymbol{x}_i - b_o) - 1 + \xi_{i,o} \geq 0, \quad i = 1, \ldots, N \\
\beta_{i,o} \geq 0, 0 \leq \alpha_{i,o} \leq C, \xi_{i,o} \geq 0, \beta_{i,o}\xi_{i,o} = 0 \quad i = 1, \ldots, N \\
\alpha_{i,o}[y_i(\boldsymbol{w}_o^T\boldsymbol{P}\boldsymbol{x}_i - b_o) - 1 + \xi_{i,o}] \geq 0, \quad i = 1, \ldots, N,
\end{aligned} \tag{18}$$

The Wolfe-dual problem as well as the hyperplane are the same as in the separable case, i.e equations (12), (13) and (14), since the slack variables and their Lagrange multipliers do not appear in it.

2.4 Deflation Kernel

Instead of solving the optimization problem of the previous section for the deflated data in every iteration as our algorithm required in order to extract the desired knowledge, the formulation of the optimization problem in the previous section allows us to incorporate the deflation transformation of each iteration in the existing optimization problem. This is possible if we consider the deflation as a kernel function, which we define as

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{x}_j \tag{19}$$

and the feature map as

$$\Phi(\boldsymbol{x}) = (\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}\boldsymbol{P}\boldsymbol{x}. \tag{20}$$

So

$$\begin{aligned}
K(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \langle\Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j)\rangle \\
&= \left((\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}\boldsymbol{P}\boldsymbol{x}_i\right)^T\left((\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}\boldsymbol{P}\boldsymbol{x}_j\right) \\
&= \boldsymbol{x}_i^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}\boldsymbol{P}\boldsymbol{x}_j \\
K(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \boldsymbol{x}_i^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{x}_j
\end{aligned} \tag{21}$$

This notation gives us the advantage that the explicit computation of all the training samples is no longer needed, but we only need to compute the dot product of the vectors in the feature space, i.e. the Hessian matrix of (13), using the kernel trick. If we use matrix notation for the data instead of vectors, as defined in Sec.2.1, the above functions are expressed as $\Phi(\boldsymbol{X}) = (\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1/2}\boldsymbol{P}\boldsymbol{X}$ and $K(\boldsymbol{X}) = \boldsymbol{X}^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{X}$.

2.5 Feature Extraction

The method described in the previous sections consists a dimensionality reduction technique. In each iteration of the procedure we obtain a new weighting vector, orthogonal to all the previously obtained ones, which is used as a projection vector in the feature extraction schemes. The number of iterations and subsequently the number of obtained weighting vectors defines the number of the resulting dimensionality. The fact that the weighting vectors are pairwise orthogonal implies that in each iteration we acquire new discriminant information regarding our data. The extracted data consist of samples of which each feature is the projection of the initial vector onto the corresponding weighting vector, $f_{\boldsymbol{w}^k}(\boldsymbol{x}) = \boldsymbol{w}^{kT}\boldsymbol{x}$.

If we denote by $\boldsymbol{W}^k$ the augmented projection matrix which contains in its columns the weighting vector of each iteration, i.e. $\boldsymbol{W}^k = (\boldsymbol{w}^1, \boldsymbol{w}^2, \ldots, \boldsymbol{w}^k)$ we can express the feature extraction of the whole procedure in a compact way using the expression $\boldsymbol{X}_D^k = \boldsymbol{W}^{kT}\boldsymbol{X}$, where by $\boldsymbol{X}_D^k$ we denote the final matrix of the extracted data after k iterations and with a reduced dimensionality of k as well. The final form of the algorithm of WSVDA, which is implemented for the experiments in this paper is shown in Table 2.

Table 2. The implemented algorithm of WSVDA

input: training set data matrix $\boldsymbol{X}$ with N samples and corresponding labels y_i
output: extracted data matrix $\boldsymbol{X}_D^k$
1: compute the within scatter matrix $\boldsymbol{S}_W$ for the initial data
2: initial projection matrix $\boldsymbol{P} = \boldsymbol{I}_{d\times d}$
3: **for** k=1 to reduced dimensionality
4: check the condition number of $(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})$, regularize by adding a small quantity to the diagonal elements if needed in order to achieve numerical stability
5: compute $(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}$
6: train SVMs using $\boldsymbol{H} = \boldsymbol{X}^T\boldsymbol{P}(\boldsymbol{P}\boldsymbol{S}_W\boldsymbol{P})^{-1}\boldsymbol{P}\boldsymbol{X}$
7: compute weighting vector $\boldsymbol{w}^k$ from (11)
8: normalize weighting vector $\boldsymbol{w}^k$
9: concatenate normalized $\boldsymbol{w}^k$ into $\boldsymbol{W}^T$
10: update projection matrix $\boldsymbol{P}$ using the normalized weighting vector of previous step, according to $\boldsymbol{P} = \boldsymbol{P}(\boldsymbol{I} - \boldsymbol{w}^k\boldsymbol{w}^{kT})$
11: **end**
12: use the normalized weighting vectors $\boldsymbol{w}$ for feature extraction, according to $\boldsymbol{X}_D^k = \boldsymbol{W}^{kT}\boldsymbol{X}$

3 Experimental Results

In this section we present the results of the experiments performed to assess the performance of WSVDA and compare it with the most commonly used techniques as PCA and LDA as well as state of the art methods as MMDA. After the dimensionality reduction of the datasets with the aforementioned techniques, classification is performed using KNN and NC algorithms. It was also considered valuable to compare these results with SVMs classification applied on the initial data.

In order to achieve higher credibility for our results we perform various instances of k-fold cross validation, with 1 fold being used as a training set and the rest $(k-1)$ folds being used as the test set. This approach offers the opportunity to use a small number of samples for the training phase, which is comparable or sometimes smaller than the number of features. That is, the dimensionality of the training set is higher than its cardinality, which is often the case for small sample size (SSS) problems. In such occasions we expect and we show that WSVDA has better performance. It is also important to mention that all the datasets were scaled uniformly to $[-1, 1]$.

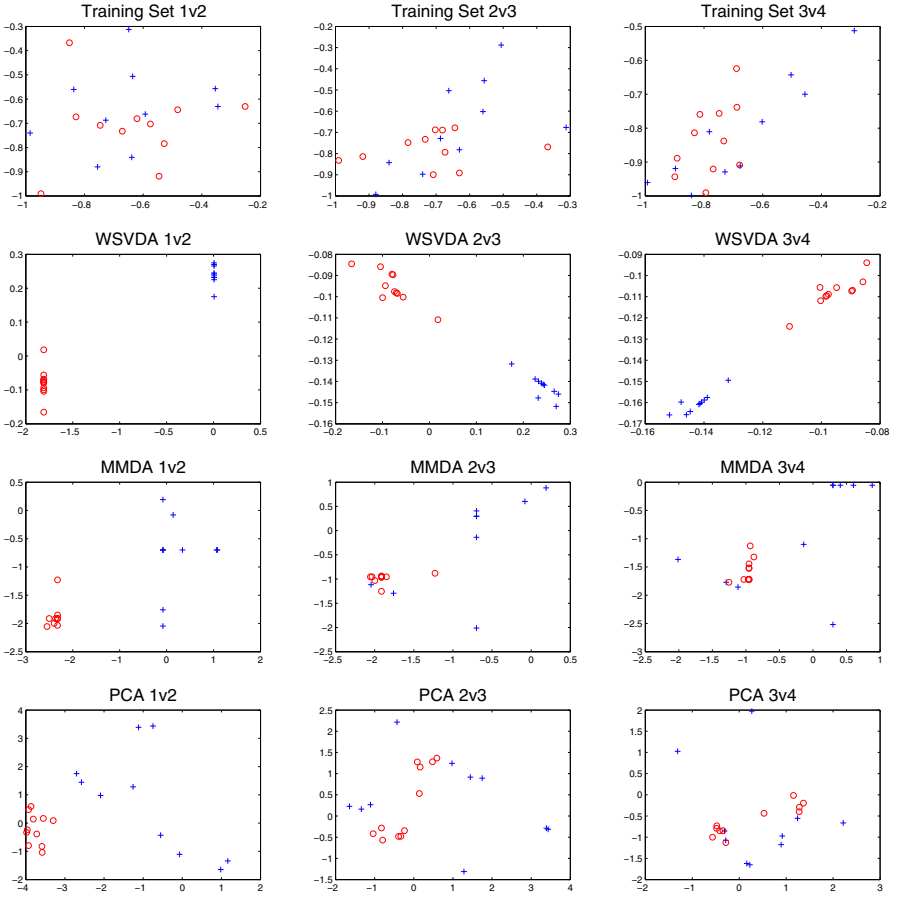

Fig. 2. Projection of pairs of features of Connectionist Bench (Sonar) dataset onto two-dimensional subspaces. In the first row three pairs of features of the initial data are shown. In the following three rows we can see the first three pairs of extracted features for WSVDA, MMDA and PCA. Best viewed in color.

3.1 Use of WSVDA for Visualization Purposes

One important attribute of WSVDA is the visualization capability, which is particularly useful for high-dimensional datasets. To demonstrate this attribute we use the Connectionist Bench dataset from the UCI Machine Learning Repository. In Fig. 2 the first three pairs of features are shown for the initial training data and after reducing their dimensionality with WSVDA, MMDA and PCA. Since the problem is binary LDA could not be used for two-dimensional visualization purposes.

We can observe that all three methods are capable of extracting discriminant information from the data and make the classification task easier compared to the initial data. However, it is important to note that only in the case of WSVDA the two classes are linearly separable for all the extracted features depicted on the figure. This means that except for the first extracted feature, the succeeding features provide additional and new discriminant information.

3.2 Dimensionality Reduction and Classification Results

In this section we present the experimental results for classification purposes using four different datasets. The experimental scenario includes a dimensionality reduction step using one of the following four techniques WSVDA, MMDA,

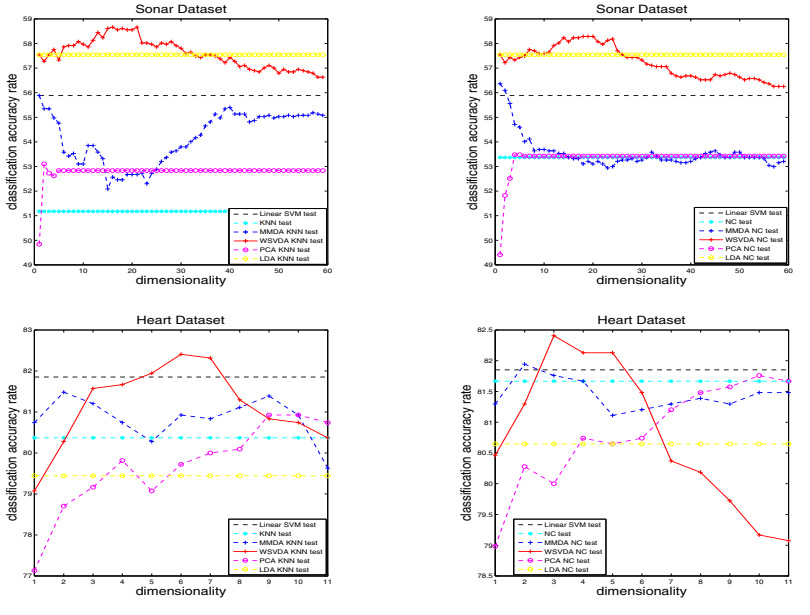

Fig. 3. Classification accuracy rates for Sonar and Heart datasets. WSVDA outperforms the other methods in all occasions in terms of classification accuracy rate and it is observable that it gains discriminant information from the successive subspaces where the optimization problem is solved. The number of neighbors for the KNN algorithm is 5 for this set of experiments, whereas the number of folds for the cross validation is 10 and 5 for Sonar and Heart datasets, respectively. Best viewed in color.

PCA, and LDA, and a classification step using either KNN or NC classification algorithms. Classification with Linear SVMs and KNN applied on the initial data is also performed, using the following datasets: 1. *Connectionist Bench (Sonar)* dataset contains 208 samples of 60 attributes that correspond to measurements of a sonar device for signals that are reflected on two different surfaces. 2. *Statlog (Heart)* dataset consists of 270 samples with 13 attributes that correspond to medical data related to heart. 3. *Wine Recognition Data* dataset contains 178 samples of 13 features which correspond to the chemical analysis of three varieties of wine. 4. *Splice-junction Gene Sequences* dataset consists of 3190 samples with 61 attributes that correspond to DNA sequences.

In Fig. 3 the average classification accuracy rates for the four dimensionality reduction techniques followed by k-Nearest Neighbor or Nearest Centroid classification are shown for Sonar and Heart datasets, first and second row respectively. On the horizontal axis we have the number of reduced dimensions. Since the problems are binary, LDA results to one-dimensional extracted data, so only one accuracy rate is available.

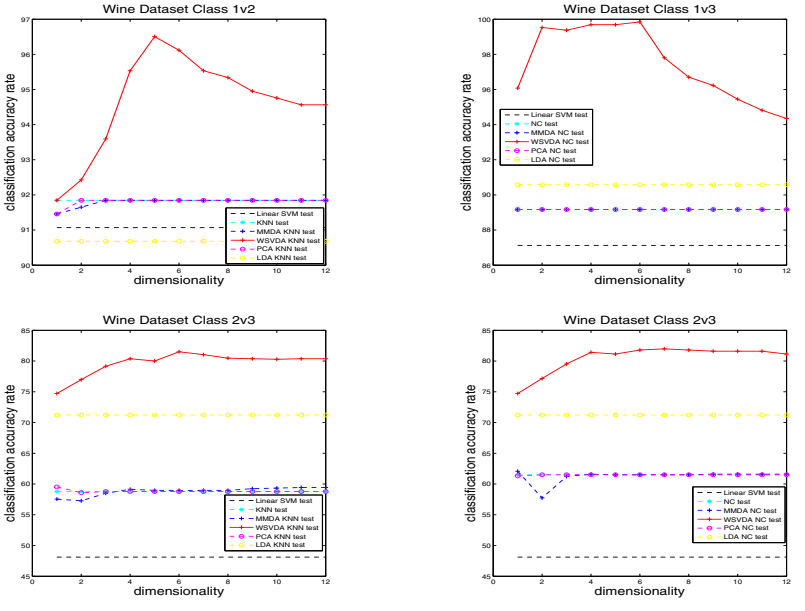

Fig. 4. Classification accuracy rates for Wine dataset. The top left subfigure corresponds to the binary problem between classes 1v2 for 5-NN classification and 5-fold cross-validation, whereas the top right subfigure corresponds to the binary problem between classes 1v3 for NC classification and 7-fold cross-validation. The second row corresponds to the binary problem between classes 2v3, which are more difficult to discriminate and due to 10-fold cross-validation, which results to very small training sample, we observe very low classification rates. WSVDA outperforms the other methods in all occasions in terms of classification accuracy rate showing that is a suitable technique for small sample size problems. Best viewed in color.

The comparison between the different methods highlights the superior performance of WSVDA in comparison to MMDA, PCA, LDA and classification with Linear SVMs and KNN or NC, applied on the initial data. In the left column we see the results for KNN and in the right one we see the results for NC. The results for the Wine Dataset are shown in Fig. 4.

Table 3 offers a detailed view over the classification results for all the datasets examined and all the approaches followed. The accuracy rates correspond to the highest value over the average accuracy rates of the cross validation and for every possible reduced dimensionality. They are instances from experiments with different parameters such as the number of folds, the number of nearest neighbors and the regularization, but same for each line of the table. This is the reason for big differences observed in classification rates. For the Wine dataset for example, the low classification rates for class 2 against 3 are due to the biggest overlapping between these classes in comparison to the other combination and due to the larger number of folds which has as a result a very small training set. This fact results to a more difficult classification task, for which WSVDA proves to be quite robust.

Table 3. Classification Results

Data sets	KNN	NC	SVM	PCA +KNN	LDA +KNN	MMDA +KNN	WSVDA +KNN	PCA +NC	LDA +NC	MMDA +NC	WSVDA +NC
Sonar	51.18	53.37	55.88	53.10	57.54	55.88	**58.66**	53.48	57.54	56.36	58.29
Heart	80.37	81.67	81.85	80.93	79.44	81.48	**82.41**	81.76	80.65	81.94	**82.41**
Wine											
1vs2	91.84	90.68	91.07	91.84	90.68	91.84	**96.50**	90.68	90.29	90.68	96.12
1vs3	89.32	89.17	87.13	89.32	90.42	89.32	**100**	89.17	90.58	89.17	99.84
2vs3	58.77	61.51	48.11	59.53	71.23	59.43	81.51	61.51	71.23	62.08	**81.98**
Splice	68.17	80.74	76.81	68.42	77.44	78.88	79.16	72.02	78.36	81.40	**81.43**

4 Conclusions

A novel dimensionality reduction method has been proposed that combines the minimization of the within class scatter matrix with the maximization of the margin between the classes in each projection. The proposed approach uses an iterative feature extraction with deflation kernels that transform the original data to perpendicular subspaces where a quadratic optimization problem is solved. Thus, the discriminant information that lie in the subspace which is perpendicular to the only dimension that standard SVM extract is exploited for better discriminality and classification. Experimental results on several datasets illustrate the superiority of the proposed approach against other popular dimensionality reduction methods.

Acknowledgments. This research has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) ERC Starting Grant agreement 204871-HUMANIS.

References

1. Cover, T.M., Hart, P.E.: Nearest Neighbor Pattern Classification. IEEE Transactions In Information Theory, 21–26 (1967)
2. Duda, O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley (2001)
3. Scholkopf, B., Smola, A.: Learning with Kernels. MIT, Cambridge (2002)
4. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer (2002)
5. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine 2, 559–572 (1901)
6. Hotelling, H.: Analysis of a Complex of Statistical Variables into Principal Components. Journal of Educational Psychology 24, 417–441 (1933)
7. Scholkopf, B., Smola, A., Muller, K.R.: Nonlinear component analysis as a Kernel eigenvalue problem. Neural Computation 10, 1299–1319 (1998)
8. Alpaydin, E.: Introduction to Machine Learning. MIT Press (2004)
9. Juwei, L., Plataniotis, K.N., Venetsanopoulos, A.N.: Face recognition using Kernel direct discriminant analysis algorithms. IEEE Transactions on Neural Networks 14, 117–126 (2003)
10. Kocsor, A., Kovács, K., Szepesvári, C.: Margin Maximizing Discriminant Analysis. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 227–238. Springer, Heidelberg (2004)
11. Tefas, A., Kotropoulos, C., Pitas, I.: Using Support Vector Machines to Enhance the Performance of Elastic Graph Matching for Frontal Face Authentication. IEEE Transactions on Pattern Analalysis and Machine Intelligence 23(7), 735–746 (2001)
12. Zafeiriou, S., Tefas, A., Pitas, I.: Minimum Class Variance Support Vector Machines. IEEE Transactions on Image Processing 16(10), 2551–2564 (2007)
13. Kung, S.Y., Diamantaras, K.I.: Neural networks for extracting unsymmetric principal components. In: Neural Networks for Signal Processing, pp. 50–59. IEEE Press, New York (1991)
14. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. In: Data Minining Knowledge Discovery, vol. 2, pp. 121–167 (1998)
15. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
16. Fletcher, R.: Practical Methods of Optimization, 2nd edn. Wiley, New York (1987)