FISHER DISCRIMINANT ANALYSIS WITH KERNELS

Sebastian Mika†, Gunnar Rätsch†, Jason Weston‡,
Bernhard Schölkopf†, and Klaus-Robert Müller†

†GMD FIRST, Rudower Chaussee 5, 12489 Berlin, Germany
‡Royal Holloway, University of London, Egham, Surrey, UK
{mika, raetsch, bs, klaus}@first.gmd.de, jasonw@dcs.rhbnc.ac.uk

Abstract. A non–linear classification technique based on Fisher's discriminant is proposed. The main ingredient is the kernel trick which allows the efficient computation of Fisher discriminant in feature space. The linear classification in feature space corresponds to a (powerful) non–linear decision function in input space. Large scale simulations demonstrate the competitiveness of our approach.

DISCRIMINANT ANALYSIS

In classification and other data analytic tasks it is often necessary to utilize pre–processing on the data before applying the algorithm at hand and it is common to first extract features suitable for the task to solve.

Feature extraction for classification differs significantly from feature extraction for describing data. For example PCA finds directions which have minimal reconstruction error by describing as much variance of the data as possible with m orthogonal directions. Considering the first directions they need not (and in practice often will not) reveal the class structure that we need for proper classification. Discriminant analysis addresses the following question: Given a data set with two classes, say, which is the best feature or feature set (either linear or non–linear) to discriminate the two classes? Classical approaches tackle this question by starting with the (theoretically) optimal Bayes classifier and, by assuming normal distributions for the classes, standard algorithms like quadratic or linear discriminant analysis, among them the famous Fisher discriminant, can be derived (e.g. [5]). Of course any other model different from a Gaussian for the class distributions could be assumed, this, however, often sacrifices the simple closed form solution. Several modifications towards more general features have been proposed (e.g. [6]); for an introduction and review on existing methods see e.g. [3, 5, 8, 11].

In this work we propose to use the kernel idea [1], originally applied in Support Vector Machines [19, 14]), kernel PCA [16] and other kernel based algorithms (cf. [14]) to define a non–linear generalization of Fisher's discriminant. Our method uses kernel feature spaces yielding a highly flexible algorithm which turns out to be competitive with Support Vector Machines.

Note that there exists a variety of methods called *Kernel Discriminant Analysis* [8]. Most of them aim at replacing the parametric estimate of class conditional distributions by a non–parametric kernel estimate. Even if our approach might be viewed in this way too, it is important to note that it goes one step further by interpreting the kernel as a dot–product in another space. This allows a theoretically sound interpretation together with an appealing closed form solution.

In the following we will first review Fisher's discriminant, apply the kernel trick, then report classification results and finally present our conclusions. In this paper we will focus on two–class problems only and discriminants linear in the feature space.

FISHER'S LINEAR DISCRIMINANT

Let $\mathcal{X}_1 = \{\boldsymbol{x}_1^1, \ldots, \boldsymbol{x}_{\ell_1}^1\}$ and $\mathcal{X}_2 = \{\boldsymbol{x}_1^2, \ldots, \boldsymbol{x}_{\ell_2}^2\}$ be samples from two different classes and with some abuse of notation $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2 = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell\}$. Fisher's linear discriminant is given by the vector $\boldsymbol{w}$ which maximizes

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T S_B \boldsymbol{w}}{\boldsymbol{w}^T S_W \boldsymbol{w}} \tag{1}$$

where

$$S_B := (\boldsymbol{m}_1 - \boldsymbol{m}_2)(\boldsymbol{m}_1 - \boldsymbol{m}_2)^T \text{ and} \tag{2}$$

$$S_W := \sum_{i=1,2} \sum_{\boldsymbol{x} \in \mathcal{X}_i} (\boldsymbol{x} - \boldsymbol{m}_i)(\boldsymbol{x} - \boldsymbol{m}_i)^T \tag{3}$$

are the between and within class scatter matrices respectively and $\boldsymbol{m}_i$ is defined by $\boldsymbol{m}_i := \frac{1}{\ell_i} \sum_{j=1}^{\ell_i} \boldsymbol{x}_j^i$. The intuition behind maximizing $J(\boldsymbol{w})$ is to find a direction which maximizes the projected class means (the numerator) while minimizing the classes variance in this direction (the denominator). But there is also a well known statistical way to motivate (1):

Connection to the optimal linear Bayes classifier: The optimal Bayes classifier compares the a posteriori probabilities of all classes and assigns a pattern to the class with the maximal probability (e.g. [5]). However, the a-posteriori probabilities are usually unknown and have to be estimated from a finite sample. For most classes of distributions this is a difficult task and often it is impossible to get a closed form estimate. However, by assuming normal distributions for all classes, one arrives at quadratic discriminant analysis (which essentially measures the Mahalanobis distance of a pattern towards the class center). Simplifying the problem even further and assuming equal covariance structure for all classes, quadratic discriminant analysis becomes linear. For two–class problems it is easy to show that the vector $\boldsymbol{w}$ maximizing (1) is in the same direction as the discriminant in the corresponding Bayes optimal classifier. Although relying on heavy assumptions which are not true in many applications, Fisher's linear discriminant has proven very powerful. One reason is certainly that a linear model is rather robust against noise and most likely will not overfit. Crucial, however, is the estimation of the scatter matrices, which might be highly biased. Using simple "plug–in" estimates as in (2) when the number of samples is small compared to the dimensionality will result in a high variability. Different ways to deal with such a situation by regularization have been proposed (e.g. [4, 7]) and we will return to this topic later.

FISHER'S DISCRIMINANT IN THE FEATURE SPACE

Clearly, for most real–world data a linear discriminant is not complex enough. To increase the expressiveness of the discriminant we could either try to use more sophisticated distributions in modeling the optimal Bayes classifier or look for non–linear directions (or both). As pointed out before, assuming general distributions will cause trouble. Here we restrict ourselves to finding non–linear directions by first mapping the data non–linearly into some feature space $\mathcal{F}$ and computing Fisher's linear discriminant there, thus thus implicitly yielding a non–linear discriminant in input space.

Let Φ be a non–linear mapping to some feature space $\mathcal{F}$. To find the linear discriminant in $\mathcal{F}$ we need to maximize

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T S_B^\Phi \boldsymbol{w}}{\boldsymbol{w}^T S_W^\Phi \boldsymbol{w}} \tag{4}$$

where now $\boldsymbol{w} \in \mathcal{F}$ and S_B^Φ and S_W^Φ are the corresponding matrices in $\mathcal{F}$, i.e.

$$\begin{aligned} S_B^\Phi &:= (\boldsymbol{m}_1^\Phi - \boldsymbol{m}_2^\Phi)(\boldsymbol{m}_1^\Phi - \boldsymbol{m}_2^\Phi)^T \text{ and} \\ S_W^\Phi &:= \sum_{i=1,2} \sum_{\boldsymbol{x} \in \mathcal{X}_i} (\Phi(\boldsymbol{x}) - \boldsymbol{m}_i^\Phi)(\Phi(\boldsymbol{x}) - \boldsymbol{m}_i^\Phi)^T \end{aligned}$$

with $m_i^\Phi := \frac{1}{\ell_i} \sum_{j=1}^{\ell_i} \Phi(\boldsymbol{x}_j^i)$.

Introducing kernel functions: Clearly, if $\mathcal{F}$ is very high– or even infinitely dimensional this will be impossible to solve directly. To overcome this limitation we use the same trick as in Kernel PCA [16] or Support Vector Machines. Instead of mapping the data explicitely we seek a formulation of the algorithm which uses only dot–products $(\Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{y}))$ of the training patterns. As we are then able to compute these dot–products efficiently we can solve the original problem without ever mapping explicitely to $\mathcal{F}$. This can be achieved using Mercer kernels (e.g. [12]): these kernels $\mathrm{k}(\boldsymbol{x}, \boldsymbol{y})$ compute a dot–product in some feature space $\mathcal{F}$, i.e. $\mathrm{k}(\boldsymbol{x}, \boldsymbol{y}) = (\Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{y}))$. Possible choices for k which have proven useful e.g. in Support Vector machines or Kernel PCA are Gaussian RBF, $\mathrm{k}(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|^2/c)$, or polynomial kernels, $\mathrm{k}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} \cdot \boldsymbol{y})^d$, for some positive constants c and d respectively.

To find Fisher's discriminant in the feature space $\mathcal{F}$, we first need a formulation of (4) in terms of only *dot products* of input patterns which we then replace by some kernel function. From the theory of reproducing kernels we know that any solution $\boldsymbol{w} \in \mathcal{F}$ must lie in the span of *all* training samples in $\mathcal{F}$. Therefore we can find an expansion for $\boldsymbol{w}$ of the form

$$\boldsymbol{w} = \sum_{i=1}^{\ell} \alpha_i \Phi(\boldsymbol{x}_i) \tag{5}$$

Using the expansion (5) and the definition of $\boldsymbol{m}_i^\Phi$ we write

$$\begin{aligned} \boldsymbol{w}^T \boldsymbol{m}_i^\Phi &= \frac{1}{\ell_i} \sum_{j=1}^{\ell} \sum_{k=1}^{\ell_i} \alpha_j \, \mathrm{k}(\boldsymbol{x}_j, \boldsymbol{x}_k^i) \\ &= \boldsymbol{\alpha}^T \boldsymbol{M}_i \end{aligned} \tag{6}$$

where we defined $(\boldsymbol{M}_i)_j := \frac{1}{\ell_i} \sum_{k=1}^{\ell_i} \mathrm{k}(\boldsymbol{x}_j, \boldsymbol{x}_k^i)$ and replaced the dot products by the kernel function. Now consider the numerator of (4). Be using the definition of S_B^Φ and (6) it can be rewritten as

$$\boldsymbol{w}^T S_B^\Phi \boldsymbol{w} = \boldsymbol{\alpha}^T M \boldsymbol{\alpha} \tag{7}$$

where $M := (\boldsymbol{M}_1 - \boldsymbol{M}_2)(\boldsymbol{M}_1 - \boldsymbol{M}_2)^T$. Considering the denominator, using (5), the definition of $\boldsymbol{m}_i^\Phi$ and a similar transformation as in (7) we find:

$$\boldsymbol{w}^T S_W^\Phi \boldsymbol{w} = \boldsymbol{\alpha}^T N \boldsymbol{\alpha} \tag{8}$$

where we set $N := \sum_{j=1,2} K_j (I - \mathbf{1}_{\ell_j}) K_j^T$, K_j is a $\ell \times \ell_j$ matrix with $(K_j)_{nm} := \mathrm{k}(\boldsymbol{x}_n, \boldsymbol{x}_m^j)$ (this is the kernel matrix for class j), I is the identity and $\mathbf{1}_{\ell_j}$ the matrix with all entries $1/\ell_j$.

Combining (7) and (8) we can find Fisher's linear discriminant in $\mathcal{F}$ by maximizing

$$J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^T M \boldsymbol{\alpha}}{\boldsymbol{\alpha}^T N \boldsymbol{\alpha}}. \tag{9}$$

This problem can be solved (analogously to the algorithm in the input space) by finding the leading eigenvector of $N^{-1}M$. We will call this approach (non–linear) *Kernel Fisher Discriminant* (KFD). The projection of a new pattern $\boldsymbol{x}$ onto $\boldsymbol{w}$ is given by

$$(\boldsymbol{w} \cdot \Phi(\boldsymbol{x})) = \sum_{i=1}^{\ell} \alpha_i \, \mathrm{k}(\boldsymbol{x}_i, \boldsymbol{x}). \tag{10}$$

Numerical issues and regularization: Obviously, the proposed setting is ill–posed: we are estimating ℓ dimensional covariance structures from ℓ samples. Besides numerical problems which cause the matrix N not to be positive, we need a way of capacity control in $\mathcal{F}$. To this end, we simply add a multiple of the identity matrix to N, i.e. replace N by N_μ where

$$N_\mu := N + \mu I. \tag{11}$$

This can be viewed in different ways: (i) it clearly makes the problem numerically more stable, as for μ large enough N_μ will become positive definite; (ii) it can be seen in analogy to [4], decreasing the bias in sample based estimation of eigenvalues; (iii) it imposes a regularization on $\|\boldsymbol{\alpha}\|^2$ (remember that we are maximizing (9)), favoring solutions with small expansion coefficients. Although the real influence in this setting of the regularization is not yet fully understood, it shows connections to those used in Support Vector Machines (see also [14]). Furthermore, one might use other regularization type additives to N, e.g. penalizing $\|\boldsymbol{w}\|^2$ in analogy to SVM (by adding the full kernel matrix $K_{ij} = \mathrm{k}(\boldsymbol{x}_i, \boldsymbol{x}_j)$).

EXPERIMENTS

Figure 1 shows an illustrative comparison of the feature found by KFD and the first and second (non–linear) feature found by Kernel PCA [16] on a toy data set. For both we used a polynomial kernel of degree two and for KFD the regularized within class scatter (11) where $\mu = 10^{-3}$. Depicted are the two classes (crosses and dots), the feature value (indicated by grey level) and contour lines of identical feature value. Each class consists of two noisy parabolic shapes mirrored at the x and y axis respectively. We see, that the KFD feature discriminates the two classes in a nearly optimal way, whereas the Kernel PCA features, albeit describing interesting properties of the data set, do not separate the two classes well (although higher order Kernel PCA features might be discriminating, too).

To evaluate the performance of our new approach we performed an extensive comparison to other state-of-the-art classifiers. The experimental setup was chosen in analogy to [10] and we compared the Kernel Fisher Discriminant to AdaBoost, regularized AdaBoost (also [10]) and Support Vector Machines (with Gaussian kernel). For KFD we used Gaussian kernels, too, and the regularized within-class scatter from (11). After the optimal direction $\boldsymbol{w} \in \mathcal{F}$ was found, we computed projections onto it by using (10). To estimate an optimal threshold on the extracted feature, one may use any classification technique, e.g. as simple as fitting a sigmoid [9]. Here we used a linear Support Vector Machine (which is optimized by gradient descent as we

Figure 1: Comparison of feature found by KFD (left) and those found by Kernel PCA: first (middle) and second (right); details see text.

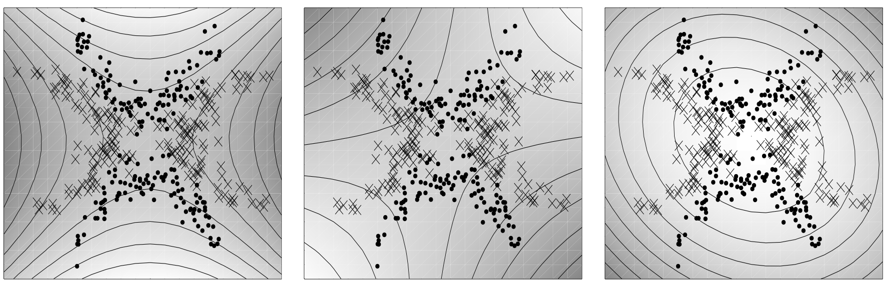

Table 1: Comparison between KFD, a single RBF classifier, AdaBoost (AB), regularized AdaBoost (AB_R) and Support Vector Machine (SVM) (see text). Best method in bold face, second best emphasized.

	RBF	AB	AB_R	SVM	KFD
Banana	**10.8±0.6**	12.3±0.7	*10.9±0.4*	11.5±0.7	**10.8±0.5**
B.Cancer	27.6±4.7	30.4±4.7	26.5±4.5	*26.0±4.7*	**25.8±4.6**
Diabetes	24.3±1.9	26.5±2.3	23.8±1.8	*23.5±1.7*	**23.2±1.6**
German	24.7±2.4	27.5±2.5	24.3±2.1	**23.6±2.1**	*23.7±2.2*
Heart	17.6±3.3	20.3±3.4	16.5±3.5	**16.0±3.3**	*16.1±3.4*
Image	3.3±0.6	**2.7±0.7**	**2.7±0.6**	*3.0±0.6*	4.8±0.6
Ringnorm	1.7±0.2	1.9±0.3	*1.6±0.1*	1.7±0.1	**1.5±0.1**
F.Sonar	34.4±2.0	35.7±1.8	34.2±2.2	**32.4±1.8**	*33.2±1.7*
Splice	*10.0±1.0*	10.1±0.5	**9.5±0.7**	10.9±0.7	10.5±0.6
Thyroid	4.5±2.1	*4.4±2.2*	4.6±2.2	4.8±2.2	**4.2±2.1**
Titanic	23.3±1.3	*22.6±1.2*	*22.6±1.2*	**22.4±1.0**	23.2±2.0
Twonorm	2.9±0.3	3.0±0.3	*2.7±0.2*	3.0±0.2	**2.6±0.2**
Waveform	10.7±1.1	10.8±0.6	**9.8±0.8**	*9.9±0.4*	*9.9±0.4*

only have 1-d samples). A drawback of this, however, is that we have another parameter to control, namely the regularization constant in the SVM.

We used 13 artificial and real world datasets from the UCI, DELVE and STATLOG benchmark repositories (except for banana).[1] The problems which are not binary were partitioned into two-class problems. Then 100 partitions into test and training set (about 60%:40%) were generated. On each of these data sets we trained and tested all classifiers (see [10] for details). The results in table 1 show the average test error over these 100 runs and the standard deviation. To estimate the necessary parameters we ran 5-fold cross validation on the first five realizations of the training sets and took the model parameters to be the median over the five estimates.[2]

Furthermore, we conducted preliminary experiments with KFD on the USPS dataset of handwritten digits where we restricted the expansion of $\boldsymbol{w}$ in (5) to run only over the first 3000 training samples. We achieved a 10–class error of 3.7% with a Gaussian kernel of width $0.3 \cdot 256$, which is slightly superior to a SVM with Gaussian kernel (4.2% [13]).

Experimental results: The experiments show that the Kernel Fisher Discriminant (plus a Support Vector Machine to estimate the threshold) is competitive or in some cases even superior to the other algorithms on almost all data sets (an exception being **image**). Interestingly, both SVM and KFD construct an (in some sense) optimal hyperplane in $\mathcal{F}$, while we notice that the one given by the solution $\boldsymbol{w}$ of KFD is often superior to the one of SVM.

[1] The breast cancer domain was obtained from the University Medical Center, Inst. of Oncology, Ljubljana, Yugoslavia. Thanks to M. Zwitter and M. Soklic for the data.

[2] In fact we did two such runs, first with a coarse and then with a finer stepping over parameter space. The data sets can be obtained via `http://www.first.gmd.de/~raetsch/`.

DISCUSSION AND CONCLUSIONS

Fisher's discriminant is one of the standard linear techniques in statistical data analysis. However, linear methods are often too limited and there have been several approaches in the past to derive more general class separability criteria (e.g. [6, 8, 5]). Our approach is very much in this spirit, however, due to the fact that we are computing the discriminant function in some feature space $\mathcal{F}$ (which is non–linearly related to input space), we are still able to find closed form solutions and maintain the theoretical beauty of Fisher's discriminant analysis. Furthermore different kernels allow for high flexibility due to the wide range of non–linearities possible.

Our experiments show that KFD is competitive to other state of the art classification techniques. Furthermore, there is still much room for extensions and further theory as linear discriminant analysis is an intensively studied field and many ideas previously developed in the input space carry over to feature space.

Note that while the complexity of SVMs scales with the number of Support Vectors, KFD does not have a notion of SVs and its complexity scales with the number of training patterns. On the other hand, we speculate, that some of the superior performance of KFD over SVM might be related to the fact, that KFD uses *all* training samples in the solution, not only the difficult ones, i.e. the Support Vectors.

Future work will be dedicated to finding suitable approximation schemes (e.g. [2, 15]) and numerical algorithms for obtaining the leading eigenvectors of large matrices. Further fields of study will include the construction of multi-class discriminants, a theoretical analysis of generalization error bounds of KFD, and the investigation of the connection between KFD and Support Vector Machines (cf. [18, 17]).

Acknowledgment. This work was supported in part by grants of the DFG (JA 379/5-1, 5-2, 7-1) and EC STORM project number 25387. We thank A. Smola for helpful comments and discussions.

REFERENCES

[1] M. Aizerman, E. Braverman and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning." **Automation and Remote Control**, vol. 25, pp. 821 – 837, 1964.

[2] C. Burges, "Simplified support vector decision rules," in L. Saitta, ed., **Prooceedings, 13th ICML**, San Mateo, CA, 1996, pp. 71–77.

[3] P. Devijver and J. Kittler, **Pattern Recognition: A Statistical Approach**, Prentice Hall, 1982.

[4] J. Friedman, "Regularized discriminant analysis," **Journal of the American Statistical Association**, vol. 84, no. 405, pp. 165–175, 1989.

[5] K. Fukunaga, **Introduction to Statistical Pattern Recognition**, San Diego: Academic Press, 2nd edn., 1990.

[6] T. Hastie and R. Tibshirani, "Discriminant analysis by gaussian mixtures," **Journal of the Royal Statistical Society**, December 1994.

[7] T. Hastie, R. Tibshirani and A. Buja, "Flexible discriminant analysis by optimal scoring," **Journal of the American Statistical Association**, December 1994.

[8] G. McLachlan, **Discriminant Analysis and Statistical Pattern Recognition**, John Wiley & Sons, 1992.

[9] J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods," Submitted to *Advances in Large Margin Classifiers*, A.J. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans, eds., MIT Press, 1999, to appear.

[10] G. Rätsch, T. Onoda and K.-R. Müller, "Soft margins for adaboost," Tech. Rep. NC-TR-1998-021, Royal Holloway College, University of London, UK, 1998, Submitted to Machine Learning.

[11] B. Ripley, **Pattern Recognition and Neural Networks**, Cambridge University Press, 1996.

[12] S. Saitoh, **Theory of Reproducing Kernels and its Applications**, Longman Scientific & Technical, Harlow, England, 1988.

[13] B. Schölkopf, **Support vector learning**, Oldenbourg Verlag, 1997.

[14] B. Schölkopf, C. Burges and A. Smola, eds., **Advances in Kernel Methods – Support Vector Learning**, MIT Press, 1999.

[15] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch and A. Smola, "Input space vs. feature space in kernel-based methods," **IEEE Transactions on Neural Networks**, 1999, Special Issue on VC Learning Theory and Its Applications, to appear.

[16] B. Schölkopf, A. Smola and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," **Neural Computation**, vol. 10, pp. 1299–1319, 1998.

[17] A. Shashua, "On the relationship between the support vector machine for classification and sparsified fisher's linear discriminant," **Neural Processing Letters**, vol. 9, no. 2, pp. 129–139, April 1999.

[18] S. Tong and D. Koller, "Bayes optimal hyperplanes → maximal margin hyperplanes," Submitted to IJCAI'99 Workshop on Support Vector Machines (`robotics.stanford.edu/~koller/`).

[19] V. Vapnik, **The nature of statistical learning theory**, New York: Springer Verlag, 1995.