Functional Learning of Kernels for Information Fusion Purposes

Alberto Muñoz and Javier González

Universidad Carlos III de Madrid, c/ Madrid 126, 28903 Getafe, Spain
{alberto.munoz,javier.gonzalez}@uc3m.es

Abstract. When there are several sources of information available in pattern recognition problems, the task of combining them is most interesting. In the context of kernel methods it means to design a single kernel function that collects all the relevant information of each kernel for the classification task at hand. The problem is then solved by training a Support Vector Machine (SVM) on the resulting kernel. Here we propose a consistent method to produce kernel functions from kernel matrices created by any given kernel combination technique. Once this fusion kernel function is available, it will be possible to evaluate the kernel at any data point. The performance of the proposed fusion Kernel is illustrated on several classification and visualization tasks.

Keywords: Mercer Kernel, Support Vector Machines, Kernel Combination, Classification problems.

1 Introduction

In pattern recognition sometimes happens that there are several sources of information available and we want to fusion all the available information to create an optimized classifier [6]. This is the starting point for techniques such as boosting or bagging of classifiers. In this paper we are interested in the particular problem of constructing an optimal classifier from a given collection of kernels [10,7]. The aim is to design a single kernel function that collects all the relevant information of each kernel for the classification task at hand. With this resulting kernel the problem is solved by training a Support Vector Machine (SVM).

The best available techniques use the classification labels to combine kernel matrices and produce another kernel matrix [10,12], but not a kernel function. As a consequence, there is no way to evaluate the combination kernel at points for which we do not know its label.

The concrete goal of this paper is to propose a consistent method to produce kernel functions from kernel matrices produced by any given kernel combination technique. Once the kernel function is available, it will be possible to evaluate the kernel at any data point.

Consider a two-class classification problem and let $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be a sample of n observations with $\mathbf{x}_i \in X$ (some input space) and $y_i \in \{1, -1\}$. Suppose there are m kernel matrices $K_{Sl} = (K_l(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$, $l = 1, \ldots m$, corresponding to m kernel functions $K_1, \ldots, K_m$, evaluated at the points of S.

J. Ruiz-Shulcloper and W.G. Kropatsch (Eds.): CIARP 2008, LNCS 5197, pp. 277–283, 2008.

The starting point will be a kernel combination matrix:

$$K_S^* = F(K_1, ..., K_m, \{y_i\}_{i=1}^n) \tag{1}$$

produced using any technique for combining the matrices K_i, and taking into account the labels of the data points in S (see [9,12,13] for some examples).

In Section 2 we propose the Fusion Kernel, a particular type of Mercer kernel able to parametrize the underlying kernel function that gives rise to K_S^*; that is, the Fusion kernel function evaluated at S gives K_S^*. To this aim we estimate a set of functions that have the property that give the eigenvectors of K_S^* , when evaluated at S. These functions will work as eigenfunctions for the Fusion Kernel. The whole process is collected into an algorithmic procedure. In Section 3 we show the performance of the proposed methodology in two cases. In the first case we check the accuracy of the Fusion Kernel function to replicate the behavior of several combination techniques in a real classification problem. In the second experiment we reconstruct a given kernel from projection kernels and check the validity of our approach.

2 Learning the Kernel Eigenfunctions from Multiple Information Sources

Each kernel K_l defines a RKHS function space H_{K_l}, and the elements of H_{K_l} can be expressed as finite linear combinations of the form $h(\mathbf{x}) = \sum_s \lambda_{sl} K_l(\mathbf{x}_s, \mathbf{x})$ where $\lambda_s \in \mathbb{R}$ and $\mathbf{x}_s \in X$ or, equivalently, $h(\mathbf{x}) = \sum_j \mu_j \phi_{jl}(\mathbf{x})$, where the $\{\phi_{jl}\}$ are the eigenfunctions of the integral operator T_{K_l} associated to K_l defined by $T_{K_l}(f) = \int_X K_l(\cdot, s) f(s) ds$ (see [1,5] for details). By Mercer theorem [1] each kernel function K_l admits a functional expansion of the form:

$$K_l(\mathbf{x}, \mathbf{y}) = \sum_j \lambda_j \phi_{jl}(\mathbf{x}) \phi_{jl}(\mathbf{y}) \tag{2}$$

where λ_{jl} is the eigenvalue corresponding to the eigenfunction ϕ_{jl}.

Definition 1 (Fusion Kernel). *Let $K_1, \ldots, K_m$ be a set of m kernel functions. A kernel function K is a Fusion Kernel for the set $K_1, \ldots, K_m$ when it can be expressed as*

$$K(\boldsymbol{x}, \boldsymbol{y}) = \sum_{h=1}^{d} \lambda_h \phi_h(\boldsymbol{x}) \phi_h(\boldsymbol{y}), \tag{3}$$

where $\{\lambda_h\} \in \mathbb{R}^+$ and $\phi_h \in Span\langle \underbrace{\phi_{11}, \ldots, \phi_{1d_1}}_{K_1}, ..., \underbrace{\phi_{m1}, \ldots, \phi_{md_m}}_{K_n} \rangle$ and ϕ_{jl} represents the jth eigenfunction of the lth kernel.

In this way, the eigenfunctions ϕ_h of any fusion kernel are defined by linear combinations of the eigenfunctions of $K_1, \ldots, K_m$:

$$\phi_h(\mathbf{x}) = \sum_{l=1}^{m} \sum_{j=1}^{d_m} c_{jl} \phi_{jl}(\mathbf{x}) \quad for \quad h = 1, \ldots, d. \tag{4}$$

Let $K^*(x, y)$ a kernel function such that K^* evaluated at the sample S gives the objective matrix (1). It may occur that there is more than one kernel function with this property because there can exist many Mercer kernels which induce the same metric [4]. In order to determine the values $\{c_{jl}\}$ in (6) we impose that each ϕ_h is the orthogonal projection onto $Span\{\phi_{jl}\}$ of each eigenfunctions ϕ_h^* of K^*. If we knew the ϕ_h^* functions, we could determine the fusion kernel in eq. (3) by minimizing $\|\phi_h^*(\mathbf{x}) - \phi_h(\mathbf{x})\|^2$.

In practice, we do not know neither the eigenfunctions ϕ_h^* of K^*, nor the eigenfunctions ϕ_{jl} of the K_l. We only can compute $\hat{\gamma}_h$, the hth eigenvector of K_S^* and $\hat{\phi}_{jl}$ the jth eigenvector of the matrix K_{Sl}. The good news are that the eigenfunctions the kernel K^* can be approximated, up to a normalization factor, by the eigenvectors of the matrix K_S^*, as Proposition 1 shows. Then we can estimate the coefficients $\{c_{il}\}$ by minimizing the following expression:

$$\widehat{J}_h = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\gamma}_l - \sum_{l=1}^{m}\sum_{j=1}^{d_m} c_{jl}\hat{\phi}_{jl}\right)^2 \quad for \;\; h = 1, \ldots d. \tag{5}$$

where, $d_l = min(n, rank(K_{Sl}))$ and $d = min(n, rank(K_S^*))$, which is a simple quadratic convex optimization problem. Denote $\{\hat{c}_{il}\}$ the solution to problem (5). In order to evaluate $\phi_h(\mathbf{x})$ for any data point using eq. (6) we need to calculate $\phi_{jl}(\mathbf{x})$. Using Proposition 1:

$$\hat{\phi}_h(\mathbf{x}) = \sum_{l=1}^{m}\sum_{j=1}^{d_m}\hat{c}_{jl}\hat{\phi}_{jl}(\mathbf{x}) = \sum_{l=1}^{m}\sum_{j=1}^{d_m}\hat{c}_{jl}\frac{1}{\sqrt{\hat{\lambda}_h}}\sum_{i=1}^{n}\hat{\phi}_{jl}K_l(\mathbf{x}, \mathbf{x}_i). \tag{6}$$

Regarding the eigenvalues λ_h of K^* in eq. (3), they can be estimated using $\hat{\lambda}_h$, the eigenvalues of the matrix K_S^*. The reason is that the eigenvalues of any Mercer kernel matrix K_S converge to the eigenvalues of the corresponding kernel function K [2,14,8]. Substituting the obtained $\phi_h(\mathbf{x})$ and $\hat{\lambda}_h$ into eq. (3) we will be able to evaluate the fusion kernel at any pair of data points, including points not in the sample S and even in the case we do not know the label of the points.

The steps to calculate a Fusion kernel for a combination matrix K_S^* are summarized in Table 1. We conclude the Section with the following proposition, used above.

Proposition 1. *[3] Let $K(\boldsymbol{x}, \boldsymbol{z})$ a kernel function, $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ a sample and K_S the $n \times n$ matrix $K_S = (K(\boldsymbol{x}_i, \boldsymbol{x}_j))$. Let $(\phi_j, \lambda_j)_{j=1}^n$ the pairs of eigenfunction and eigenvalues of the kernel function K. Let $(\hat{\phi}_j, \widehat{\lambda}_j)_{j=1}^n$ the pairs of eigenvectors and eigenvalues of the matrix K_S. Then, for any $\boldsymbol{x}$:*

$$\phi_j(\boldsymbol{x}) = \frac{\sqrt{n}}{\widehat{\lambda}_j}\sum_{i=1}^{n}\hat{\phi}_{ji}K(\boldsymbol{x}, \boldsymbol{x}_i) \tag{7}$$

$$For \;\; any \;\; \boldsymbol{x}_i \in S \;, \quad \phi_j(\boldsymbol{x}_i) = \sqrt{n}\hat{\phi}_{ji} \tag{8}$$

Table 1. Fusion Kernel estimation algorithm

INPUT		
	K_S^*:	kernel matrix of a given (possible unknown) combination.
	$K_{S1}, ..., K_{Sm}$	kernel matrices involved in the combination.
OUTPUT		
	K:	Fusion Kernel of K_S^*
STEPS		
		1.- Decompose $K_S^* = U\Lambda U'$, with $U = [u_1, \ldots, u_d]$ and $\Lambda = diag(\hat{\lambda}_1, ..., \hat{\lambda}_d)$.
		2.- Decompose $K_S l = V_l Q_l V_l'$ for $l = 1, ..., m$.
		3.- Create the matrix V=$[V_1, ..., V_m]$.
		4.- For $h = 1, \ldots, d$ solve the linear system $Ac_h = b$, where $A = V'V$ and $b = V'u_h$.
		5.- Estimate every ϕ_h by eq. (6) .
		6.- Consider eq. (3) for $\lambda_h = \hat{\lambda}_h$.

3 Experiments

3.1 Visualization Example

Given a kernel matrix K_S, it is immediate to obtain a distance matrix D_S by letting $d(x, y) = k_{xx} + k_{yy} - 2k_{xy}$, and we can plot the sample S in $\mathbb{R}^2$ applying Multidimensional Scaling on D_S. In this example we will consider three different kernels K_i (that give rise to three different representations in $\mathbb{R}^2$) and will produce a combining matrix K_S^* using different combination algorithms. We will see that the Fusion Kernel is able to position correctly new data points in the 2-dimensional plot. Notice that for new data points x we can not evaluate $K(x, x_i)$, for $x_i \in S$ and we do not have a direct way to position them in the plot.

The data base consists of radar data consisting of a phased array of 16 high-frequency antennas. The targets are the free electrons in the ionosphere [15] and the two clases are labeled as "Good" – radar returns showing evidence of some type of structure in the ionosphere – and "Bad" (returns without structure). There are 351 data points.

We consider three kernels, $K_1(\mathbf{x}, \mathbf{y}) = \mathbf{x}'\mathbf{y}$,$K_2(\mathbf{x}, \mathbf{y}) = (\mathbf{x}'\mathbf{y} + 1)^2$ and $K_3(\mathbf{x}, \mathbf{y}) = \exp\{-\gamma\|\mathbf{x} - \mathbf{y}\|^2\}$ for $\gamma = 1$ and two combinations schemes. Let K_{S1}, K_{S2} and K_{S3} the three kernel matrices calculated over a training sample S of size n, $\overline{K}_S = \frac{1}{m}\sum_{l=1}^{m} K_{Sl}$ the mean of the kernels, 1_n the column vector of ones and n elements, and Y a diagonal matrix whose non-null elements are the labels of the problem. Then, the two combinations schemes (see [9] for details) are given by:

1. $K_{SM} = \overline{K} + \tau Y \sum_{l=1}^{m} (K_{Sl} - \overline{K}_S)^2 Y$
2. $K_{MAKM} = \overline{K} + \tau Y 1_n 1_n' Y$

These methods need the labels to build the combined kernel matrix and, thus, K_S^* cannot be evaluated for testing points, for which we do not know the label.

Now, we consider 346 data points and produce two different combination matrices using, respectively, SM and MAKM. Then we obtain the two fusion kernels using the algorithm in Table 1. Finally, we evaluate $K(x, \cdot)$ for the 5 remaining points we did not use for building the kernel combinations.

In Figure 1 we show the ionosphere data set representations corresponding to the kernels given by SM and MAKM. If we used the whole data set to build the combination kernels, we had obtained the location of the 5 test points shown in the Figure by black dense circles. The squares show the location of these points using the fusion kernel. It is apparent that the fusion kernel is able to locate the data points at their right places.

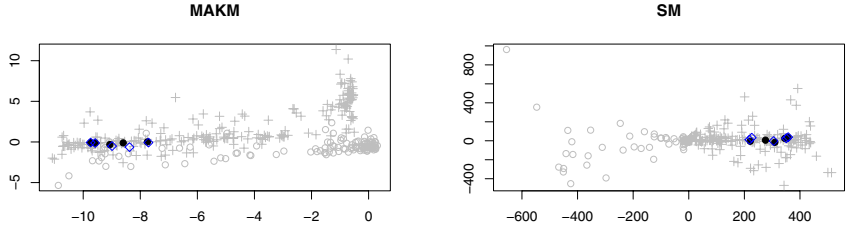

Fig. 1. Representation of the two kernel combinations. Crosses and circuinferences represent the to classes for the train data. Black circles represent the theoretical embedding for the test data and the blue circunferences their approximations.

3.2 Classification Example

In this example we show the performance of the method in a controlled two-class classification example. We generate 200 train point and 50 test points in $\mathbb{R}^2$. In Class 1 $(x, y) = (u + 1, u^2 + e)$ and $(x, y) = (u + 7/5, -u^2 + 1 + e)$ for Class 2, where $u \sim U(-1, 1)$ is a uniform random variable and $e \sim N(0, 0.1)$ a Gaussian one. The sample is shown in Figure 2.

We consider with two kernels based on the projections of the data onto the two coordinate axis: $K_1(\mathbf{x}, \mathbf{y}) = \pi_x(\mathbf{x})\pi_x(\mathbf{y}) = x_1 y_1$ and $K_2(\mathbf{x}, \mathbf{y}) = \pi_y(\mathbf{x})\pi_y(\mathbf{y}) = x_2 y_2$. In order to estimate the accuracy of the procedure, we consider the battery of kernels of increasing complexity $K^*(d) = (K_1 + K_2)^d$ for $d = 1, \ldots, 15$.

Let $K_S^*(d)$ denote the matrices obtained by applying the real combination kernels $K^*(d)$ to the points in the sample of 250 points. Let $K_F(d)$ denote the fusion kernel and $K_L(d) = \sum_{i=1}^{2} \lambda_i(d) K_i$ a linear combination of kernels defined as the best approximation (using the Frobenius norm) to $K^*(d)$ using a linear combination of the K_i kernels. We want to compare the reference matrices $K_S^*(d)$ with the matrices obtained applying $K_F(d)$ and $K_L(d)$ to the sample.

To this aim we use two goodness of fit measures. First we classify the test data using $K_F(d)$ and $K_L(d)$ and compare with the results obtained using the reference matrix $K_S^*(d)$. In the second case we measure the difference between matrices using the Frobenius norm. The results are shown in Figure 3. Fig. 3 left shows that, for every d, the fusion kernel $K_F(d)$ obtains a quite similar performance to that obtained by the kernel it tries to reproduce (that is, $K^*(d)$).

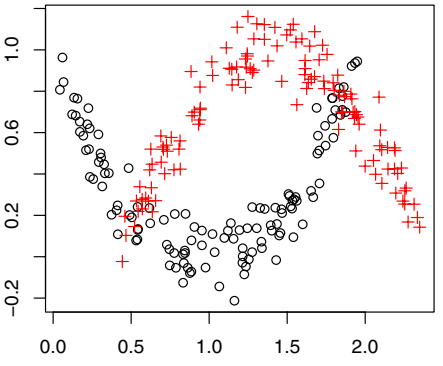

Fig. 2. Plot of the generated data

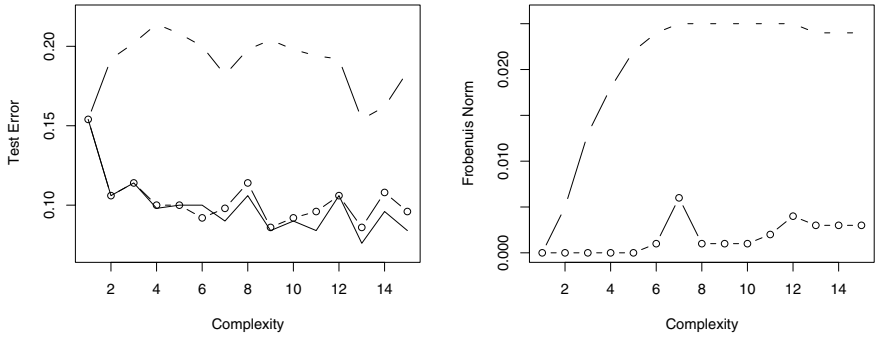

Fig. 3. Comparison of the proposed method (line with circles) and the linear combination (dotted line) in an example of increasing complexity. Classification errors (left) and differences in terms of the frobenius norm (right) are shown.

This is not the case for $K_L(d)$, a linear combination of kernels (instead a linear combination of eigenfunctions, as the fusion kernel is). Thus, in a real case where we do not know in advance the results for the test data, we can expect a good behavior for the fusion kernel. Fig. 3 right shows the adjustment of K_F and K_L to the true combination kernel as d increases. Again, the fusion kernel K_F remains similar to $K^*(d)$.

4 Conclusions

Combining kernels is a promising line of research when solving pattern recognition tasks. The best combination techniques available combine kernel matrices to produce another kernel matrices, but it is not possible to evaluate the kernel at new data points. In this work we present a parametrized kernel, the fusion kernel, able to approximate the function kernel that generates the combination kernel for any given fusion technique. In the experiments we show that the fusion kernel is able to approximate correctly the underlying function that generates

the combination kernel in each case. In particular, we show the ability of the fusion kernel to approximate highly non-linear combination of kernel functions using a nonlinear classification problem.

Acknowledgments

This work was partially supported by Spanish grant SEG 2007/64500.

References

1. Aroszajn, N.: Theory of Reproducing Kernels. Transactions of the American Mathematical Society 68(3), 337–404 (1950)
2. Bengio, Y., Delalleau, O., Le Roux, N., Paiement, J.-F., Vincent, P., Ouimet, M.: Learning eigenfunctions links spectral embedding and kernel PCA. Neural Computation 16, 2197–2219 (2004)
3. Bengio, Y., Vincent, P., Paiement, J.: Learning eigenfunctions of similarity: Linking spectral clustering and kernel PCA. Technical Report 1232, Département d'informatique et recherche opérationnelle, Université de Montréal (2003)
4. Burges, C.J.C., Platt, J.C., Jana, S.: Distortion discriminant analysis for audio fingerprinting. EEE Transactions on Speech and Audio (2003)
5. Cucker, F., Smale, S.: On the Mathematical Foundations of Learning. Bulletin of the American Mathematical Society 39(1), 1–49 (2002)
6. Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine intelligence 20(3), 226–239 (1998)
7. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., El Ghaoui, L., Jordan, M.I.: Learning the Kernel Matrix with Semidefinite Programming. Journal of Machine Learning Research 5, 27–72 (2004)
8. Mikio, L.B.: Accurate Error Bounds for the Eigenvalues of the Kernel Matrix. Journal of Machine Learning Research 7, 2303–2328 (2006)
9. Martin de Diego, I., Moguerza, J.M., Muñoz, A.: Combining Kernel Information for Support Vector Classification. In: Roli, F., Kittler, J., Windeatt, T. (eds.) MCS 2004. LNCS, vol. 3077, pp. 102–111. Springer, Heidelberg (2004)
10. Moguerza, J., Muñoz, A.: Support Vector Machines with Applications. Statistical Science 21(3), 322–336 (2006)
11. Moguerza, J.M., Muñoz, A., de Diego, I.M.: Improving Support Vector Classification via the Combination of Multiple Sources of Information. In: Fred, A., Caelli, T.M., Duin, R.P.W., Campilho, A.C., de Ridder, D. (eds.) SSPR&SPR 2004. LNCS, vol. 3138, pp. 592–600. Springer, Heidelberg (2004)
12. Muñoz, A., de Diego, I.M.: From indefinite to Semi-Definite Matrices. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) SSPR 2006 and SPR 2006. LNCS, vol. 4109, pp. 764–772. Springer, Heidelberg (2006)
13. Muñoz, A., González, J.: Joint Diagonalization of Kernels for Information Fusion. In: Rueda, L., Mery, D., Kittler, J. (eds.) CIARP 2007. LNCS, vol. 4756, pp. 556–563. Springer, Heidelberg (2007)
14. Schlesinger, S.: Approximating Eigenvalues and Eigenfunctions of Symmetric Kernels. Journal of the Society for Industrial and Applied Mathematics 6(1), 1–14 (1957)
15. Sigillito, V.G., Wing, S.P., Hutton, L.V., Baker, K.B.: Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Technical Digest 10, 262–266 (1989)