Fusion of Gaussian Kernels Within Support Vector Classification

Javier M. Moguerza[1], Alberto Muñoz[2], and Isaac Martín de Diego[1]

[1] University Rey Juan Carlos, c/ Tulipán s/n, 28933 Móstoles, Spain
{javier.moguerza, isaac.martin}@urjc.es
[2] University Carlos III de Madrid, c/ Madrid 126, 28903 Getafe, Spain
alberto.munoz@uc3m.es

Abstract. In this paper we propose some methods to build a kernel matrix for classification purposes using Support Vector Machines (SVMs) by fusing Gaussian kernels. The proposed techniques have been successfully evaluated on artificial and real data sets. The new methods outperform the best individual kernel under consideration and they can be used as an alternative to the parameter selection problem in Gaussian kernel methods.

1 Introduction

It is well known that the choice of kernel parameters is often critical for the good performance of Support Vector Machines (SVMs). Nevertheless, to find optimal values in terms of generalization performance for the kernel parameters is an open and hard to solve question. An a priori kernel selection for SVM is a difficult task [1]. The Gaussian kernel (or radial basis function (RBF) kernel) function is one of the most popular classical SVM kernels. The effect of RBF kernels parameter within a SVM framework has been studied from a theoretical point of view [5]. Several practical proposals to choose the RBF kernel parameter have been made [14,7,3,13]. However, there is not a simple and unique technique to select the best set of parameters to build a kernel matrix. Our proposal is based on the fusion of the different RBF kernel matrices that arise with the use of a range of values for the unkown parameters. Fusing kernels provides a solution that minimizes the effect of a bad parameter choice. An intuitive and usual approach to build this fusion is to consider linear combinations of the matrices. This is the proposal in [6], which is based on the solution of a semi-definite programming problem to calculate the coefficients of the linear combination. Nevertheless, the solution of this kind of optimization problem is computationally very expensive [16].

In this paper we propose several methods to build a kernel matrix from a collection of RBF kernels generated from different values of the unkown parameters in the RBF kernel function. The functions involved in the proposed methods take advantage of class conditional probabilities and nearest neighbour techniques.

The paper is organized as follows. The general framework for the methods is presented in Section 2. The proposed methods are described in Section 3. The experimental setup and results on artificial and real data sets are described in Section 4. Section 5 concludes.

2 General Framework

Consider the general expression of the RBF kernel function:

$$K(x_i, x_j) = exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right), \tag{1}$$

where $\sigma > 0$ is the kernel parameter, and x_i and x_j are data points in the sample. The kernel parameter controls the flexibility of the kernel. Small values of σ gradually reduce the kernel to the identity matrix. On the other hand, large values of σ imply that the kernel matrix become close to a constant function. Notice that the RBF kernel matrix defines a similarity measure. As already mentioned, our proposal is based on the generation of a collection of kernel matrices using a wide range of values for the unkown RBF kernel parameter. Once the collection has been built, we will fuse the matrices in order to build a unique kernel.

In order to fuse the kernel matrices we make use of the concept of functional fusion of matrices. This concept is based on the one introduced originally in [10]. Let $K_1, K_2, ...K_M$ be a set of M normalized input RBF kernel matrices defined from (1) on a data set X, and denote by K^* the desired output combination. Let y denote the label vector, where for simplicity $y_i \in \{-1, +1\}$ (the extension to the multiclass case is straightforward).

Consider the following (functional) weighted sum:

$$K^* = \sum_{m=1}^{M} W_m \otimes K_m\,, \tag{2}$$

where $W_m = [w_m(x_i, x_j)]$ is a matrix whose elements are nonlinear functions $w_m(x_i, x_j)$, and '$\otimes$' denotes the element by element product between matrices (Hadamard product). Notice that if $w_m(x_i, x_j) = \mu_m$, where $\mu_m\,, m = 1, \dots M$ are constants, then the method reduces to calculate a simple linear combination of matrices:

$$K^* = \sum_{m=1}^{M} \mu_m K_m\,. \tag{3}$$

Several methods have been suggested to learn the coefficients μ_m of the linear combination [2,6]. Thus, the formulation used in these papers is a particular case of the formula we propose. For instance, if we take $\mu_m = \frac{1}{M}$, the average of the input matrices is obtained.

Regarding our proposals, consider the (i, j) element of the matrix K^* in (2):

$$K^*(x_i, x_j) = \sum_{m=1}^{M} w_m(x_i, x_j) K_m(x_i, x_j)\,. \tag{4}$$

This is the general formula of our approximation. In this way, we will generate a particular weight for each pair of elements under consideration.

An aspect that has to be treated before describing the methods is the fact that the kernel matrix arising from the combination has to be a positive semi-definite matrix. Since this can not be guaranteed in advance, we make use of some of the several solutions that have been proposed to solve this difficulty [12]. For instance, consider the spectral decomposition $K^* = Q\Lambda Q^T$, where Λ is a diagonal matrix containing (in decreasing order) the eigenvalues of K^*, and Q is the matrix of the corresponding eigenvectors. Assume that Λ has at least p positive eigenvalues. We can consider a p-dimensional representation by taking the first p columns of Q: $Q_p\Lambda_p Q_p^T$. We will refer to this technique as 'Positive Eigenvalue Transformation'. A computationally cheaper solution is to consider the definition of a new kernel matrix as K^{*2}. Notice that, in this case, the new kernel matrix is: $Q\Lambda^2 Q^T$. We call this method 'Square Eigenvalue Transformation'. In practice, there seems not to be a universally best method to solve this problem [11].

3 Some Specific Proposals

The next section describes a common feature to the methods we will propose: The use of conditional class probabilities in order to build the weights $w_m(x_i, x_j)$ introduced in the previous section.

3.1 Conditional Class Probabilities

Consider the pair (x_i, y_i) and an unlabelled observation x_j. Given the observed value x_j, define $P(y_i|x_j)$ as the probability of x_j being in class y_i. If x_i and x_j belong to the same class this probability should be high. Unfortunately, this probability is unknown and has to be estimated. In our proposals, we will estimate it by:

$$P(y_i|x_j) = \frac{n_{ij}}{n}, \tag{5}$$

where n_{ij} is the number of the n-nearest neighbours of x_j belonging to class y_i. Notice that each kernel matrix induces a different type of neighborhood. In fact, there is an explicit relation between a kernel matrix and a distance matrix. For instance, consider a matrix K of inner products in an Euclidean space $\mathcal{F}$ (a kernel). Then $D^2 = ve^T + ev^T - 2K$ is a matrix of square Euclidean distances in $\mathcal{F}$ [4], where v is a vector made up of the diagonal elements of K. Hence, it is advisable to estimate this probability for each representation, that is, for the matrix K_m we will estimate the conditional probabilities $P_m(y_i|x_j)$ using the induced distances matrix D_m^2. We will need the average of this conditional probabilities over the kernel matrices:

$$\bar{\rho}(x_i, x_j) = \frac{\bar{P}(y_i|x_j) + \bar{P}(y_j|x_i)}{2}, \tag{6}$$

where $\bar{P}(y_i|x_j) = \frac{1}{M}\sum_{m=1}^{M} P_m(y_i|x_j)$.

To estimate the conditional class probabilities, the appropriate size of the neighbourhood has to be determined. We propose a dynamic and automatic method: given two points x_i and x_j, we look for the first common neighbour. For each data point (x_i and x_j), the size k of the neighbourhood will be determined by the number of neighbours nearer than the common neighbour. To be more specific, let $R(x_i, n) = \{\textit{n-nearest neighbours of } x_i\}$, then $k = \text{argmin}_n \{R(x_i, n) \cap R(x_j, n) \neq \emptyset\}$. Obviously, the size k of the neighbourhood depends on the particular pair of points under consideration.

At this point, we have the tools to implement some particular proposals of combination methods.

3.2 The 'MaxMin' Method

The 'MaxMin' method (first used in [10]) produces a functional fusion of two kernel matrices, namely, the maximum and the minimum of the ordered sequence of similarities, being zero the weight assigned to the rest of the similarities. Consider the ordered sequence:

$$\min_{1\leq m\leq M} K_m(x_i,x_j) = K_{[1]}(x_i,x_j) \leq \ldots \leq K_{[M]}(x_i,x_j) = \max_{1\leq m\leq M} K_m(x_i,x_j)\,,$$

where the subscript $[\cdot]$ denotes the position induced by the order. This method builds each element of K^* using the formula:

$$K^*(x_i,x_j) = \bar{\rho}(x_i,x_j)K_{[M]}(x_i,x_j) + (1-\bar{\rho}(x_i,x_j))K_{[1]}(x_i,x_j)\,. \tag{7}$$

If x_i and x_j belong to the same class then the conditional class probabilities $\bar{\rho}(x_i,x_j)$ will be high and the method guarantees that $K^*(x_i,x_j)$ will be large. On the other hand, if x_i and x_j belong to different classes the conditional class probabilities $\bar{\rho}(x_i,x_j)$ will be low and the method will produce a value close to the minimum of the similarities. In the following, this method will be refered as **MaxMin**.

3.3 The Percentile-In Method

Next we propose a method whose assignment of positive weights $w_m(x_i,x_j)$ is based on the order induced by the similarities. The method builds each element of K^* using the following formulae:

$$K^*(x_i,x_j) = K_{\lceil \bar{\rho}(x_i,x_j)M \rceil}\,, \tag{8}$$

where the subscript $\lceil\cdot\rceil$ denotes the upper rounding of the argument.

We denote this method by **'Percentile-in'** method [10] . If the class probability $\bar{\rho}(x_i,x_j)$ is high, we can expect a high kernel between x_i and x_j and the method will guarantee a high $K^*(x_i,x_j)$. If the class probability $\bar{\rho}(x_i,x_j)$ is low, $K^*(x_i,x_j)$ will be also low.

3.4 The Percentile-Out Method

As in the previous method, the last proposed technique is based on the order induced by the similarities. However, in this case two similarities are considered. Each element of the K^* matrix is built as follows:

$$K^*(x_i, x_j) = \frac{1}{2}\left(K_{\lceil \bar{P}(y_i|x_j)M \rceil} + K_{\lceil \bar{P}(y_j|x_i)M \rceil}\right), \tag{9}$$

where the subscript $\lceil \cdot \rceil$ denotes the upper rounding of the argument. We denote this method by **'Percentile-out'** method [10] .

If the conditional class probabilities $\bar{P}(y_i|x_j)$ and $\bar{P}(y_j|x_i)$ are high, we can expect a high kernel between x_i and x_j and both methods will guarantee a high $K^*(x_i, x_j)$. If the conditional class probabilities $\bar{P}(y_i|x_j)$ and $\bar{P}(y_j|x_i)$ are both low, $K^*(x_i, x_j)$ will be also low.

4 Experiments

To test the performance of the proposed methods, a SVM (with the upper bound on the dual variables fixed to 1) has been trained on several real data sets using the output matrix K^* constructed.

In order to classify a non-labelled data point x, $K^*(x, i)$ has to be evaluated. We calculate two different values for $K^*(x, i)$, the first one assumming x belongs to class $+1$ and the second assumming x belongs to class -1. For each assumption, we compute the distance between x and the SVM hyperplane and assign x to the class corresponding to the largest distance from the hyperplane.

Since our technique is based on the calculation of the nearest neighbours, we have compared the proposed methods with the k-Nearest Neighbour classification (k-NN, using the optimal value $k = l^{\frac{4}{p+4}}$, where l is the sample size and p is the data dimension [15]). In order to evaluate the improvement provided by our proposals, we have carried out a Wilcoxon signed-rank test (see for instance [8]). This nonparametric test is used to compare the median of the results for different runs of each method. So, the null hypothesis of the test is that our methods do not improve the individual kernels.

4.1 Two Areas with Different Scattering Matrices

This data set, shown in Figure 1, is made up of 400 points in $\mathbb{R}^2$. Visually there are two areas of points (80% of the sample is in area A_1 and 20% is in area A_2). Each area A_i corresponds to a circle with radio σ_i. Here $\sigma_1 = 10^{-2}\sigma_2$, with $\sigma_2 = 1$. The first group center is $(0, 1)$ and the second group center is $(1, 1)$. Nevertheless, the areas do not coincide with the classes $\{-1, +1\}$ that are to be learned. Half of the points in each class belongs to aread A_1, and the other half to area A_2. Within each area, the classes are linearly separable. Therefore the only way to built a classifier for this data set is to take into account the area each point belongs to. We use 50% of the data for training and 50% for testing.

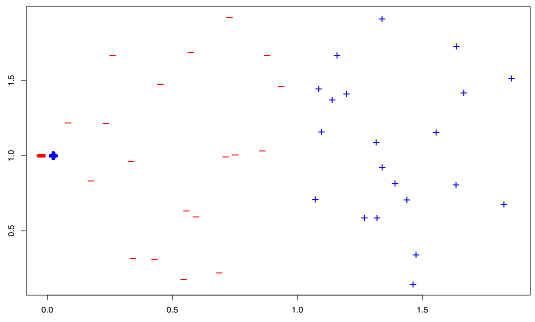

Fig. 1. Two areas with different scattering matrices. The first area center is $(0, 1)$ and the second area center is $(1, 1)$. The areas do not coincide with the classes $\{-1, +1\}$.

Let $\{K_1, \ldots, K_5\}$ be a set of five RBF kernels with parameters σ =0.5, 2.5, 5, 7.5 and 10 respectively. We normalize the kernel matrices: $K(x, z) = \frac{K(x,z)}{\sqrt{K(x,x)}\sqrt{K(y,y)}}$. In order to get a positive semi-definite kernel matrix K^*, we use the Square Eigenvalue Transformation technique described in Section 2.

Table 1 shows the performance of our proposals for this data set. The results have been averaged over 10 runs. Given the geometry of the data, it is clear that is not possible to choose a unique best σ for the whole data set. As σ grows, the test error increases for the data contained in area A_1, and decreases within area A_2. The LC method seems to work fairly. Nevertheless, the MaxMin method achieves the best results on classification. Regarding the Wilcoxon signed-rank test for the comparison of our methods with the LC technique, the p-value is smaller than 0.001 for the MaxMin method.

4.2 Cancer Data Set

In this section we have dealt with a database from the UCI Machine Learning Repository: the Breast Cancer data set [9]. The data set consists of 683 observations with 9 features each. Let $\{K_1, \ldots, K_{12}\}$ be a set of RBF kernels with parameters σ =0.1, 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 respectively. We use the Positive Eigenvalue Transformation to solve the problem of building a positive semi-definite matrix.

Table 2 shows the performance of the proposed methods when combining all these kernel matrices. Again, the results have been averaged over 10 runs. The MaxMin method, the Percentile-in method, and the Percentile-out method improve the best RBF kernel under consideration (test errors of 2.8% for the three methods vs. 3.1%). The results provided by all the combination methods are not degraded by the inclusion of kernels with a bad generalization performance. Our methods clearly outperform the SVM classifier using an RBF kernel with $\sigma = \sqrt{d}/2$, where d is the data dimension (see [14] for details). Regarding the Wilcoxon signed-rank test for the comparison of our methods with the SVM technique, the p-values are smaller than 0.05 for the MaxMin and the Percentile-out

Table 1. Percentage of missclassified data and percentage of support vectors for the two different scattering data set: A_1 stands for the less scaterring group, A_2 stands for the most dispersive one

	Train Error			**Test Error**			**Support Vectors**		
Method	Total	A_1	A_2	Total	A_1	A_2	Total	A_1	A_2
RBF$_{\sigma=0.5}$	2.1	2.6	0.0	13.5	4.1	51.0	39.6	25.1	97.5
RBF$_{\sigma=2.5}$	4.8	6.0	0.0	13.5	6.5	41.5	62.2	53.4	97.5
RBF$_{\sigma=5}$	6.6	8.2	0.0	14.0	10.1	29.5	82.8	79.2	97.0
RBF$_{\sigma=7.5}$	16.0	19.9	0.5	22.2	22.6	20.5	94.6	94.2	96.0
RBF$_{\sigma=10}$	30.7	38.2	0.5	37.3	44.1	10.0	94.2	95.4	89.5
MaxMin	0.3	0.4	0.0	4.9	0.9	21.0	27.7	9.6	100.0
Percentile-in	4.2	5.1	0.5	9.0	3.1	32.5	35.9	20.1	99.0
Percentile-out	0.7	0.9	0.0	7.7	1.1	34.0	29.0	11.4	99.5
k-**NN**	14.5	3.5	58.5	15.5	3.5	63.5	—	—	—
LC	1.6	2.0	0.0	8.1	2.5	29.5	46.6	33.2	100.0

Table 2. Percentage of missclassified data, sensitivity (Sens.), specificity (Spec.) and percentage of support vectors for the cancer data using a battery of RBF kernels. Standard deviations in brackets.

	Train			**Test**			**Support**
Method	Error	Sens.	Spec.	Error	Sens.	Spec.	**Vectors**
Best RBF	2.3 (0.3)	0.979	0.976	3.1 (1.6)	0.976	0.966	13.6 (1.3)
Worst RBF	0.0 (0.0)	1.000	1.000	24.7 (2.3)	1.000	0.627	74.0 (2.4)
MaxMin	0.1 (0.1)	0.999	0.998	2.8 (1.6)	0.963	0.975	14.2 (1.5)
Percentile-in	2.0 (0.4)	0.982	0.979	2.8 (2.8)	0.975	0.969	7.8 (0.7)
Percentile-out	0.2 (0.1)	0.999	0.997	2.8 (1.7)	0.964	0.975	19.2 (4.5)
k-**NN**	2.7 (0.5)	0.961	0.980	3.4 (1.5)	0.949	0.974	— (—)
LC	0.0 (0.0)	1.000	1.000	3.2 (1.6)	0.976	0.964	41.5 (4.4)
SVM	0.1 (0.1)	1.000	0.999	4.2 (1.4)	0.989	0.942	49.2 (1.0)

methods, and smaller than 0.1 for the Percentile-in method. Again, the improvement obtained by the use of our proposals is statistically significant.

5 Conclusions

In this paper, we have proposed some methods for the fusion of RBF kernels in order to improve their classification ability. The proposed techniques are specially usefull when does not exist an overall and unique best RBF kernel. The suggested kernel fusion methods compare favorably to the single use of one of the RBF kernels involved in the combination. Further research will focus on the theoretical properties of the methods. In particular, the methods shown in this paper do not take full advantage of the concept of the functional weighted sum described in (2): we think that there is room for improvement and more sophisticated ways for the calculus of the weights for the particular case of RBF kernel matrices may be designed. There are two natural extensions of this work. Firstly, the application of this methodology to kernels not defining a similarity (for instance, polynomial kernels). In this case, care has to be taken when transforming the kernel into a similarity. A second extension would be the generalization for the fusion of different types of kernels. In this case, the normalization of the kernels has to be carefully studied.

Acknowledgments. This work was partially supported by Spanish grants TIC2003-05982-C05-05, MTM2006-14961-C05-05, and SEJ 2004-03303.

References

1. S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12:783–789, 1999.
2. O. Bousquet and D.J.L. Herrmann. On the complexity of learning the kernel matrix. In S. Becker, S. Thurn, and K. Obermayer, editors, *Advances in Neural Information Processing Systems, 15*, pages 415–422. Cambridge, MA: The MIT Press, 2003.
3. O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1/3):131–159, 2002.
4. J. C. Gower and P. Legendre. Metric and euclidean properties of dissimilarity coefficients. *Journal of Classification*, 3:5–48, 1986.
5. S.S. Keerthi and C. Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation*, 15:1667–1689, 2003.
6. G. R. G. Lanckriet, N. Cristianini, P. Barlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5(Jan):27–72, 2004.
7. J.-H. Lee and C.-J. Lin. Automatic model selection for support vector machines. Technical report, National Taiwan University, 2000.
8. E. L. Lehmann. *Nonparametrics: Statistical Methods Based on Ranks.* McGraw-Hill, 1975.
9. O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1–18, 1990.
10. J. M. Moguerza, I. Martín de Diego, and A. Muñoz. Improving support vector classificacion via the combination of multiple sources of information. In *Proc. of the IAPR International Workshops SSPR 2004 and SPR 2004, Vol. 3138 of LNCS*, pages 592–600. Berlin: Springer, 2004.
11. E. Pękalska, R. P. W. Duin, S. Günter, and H. Bunke. On not making dissimilarities euclidean. In *Proc. of the IAPR International Workshops SSPR 2004 and SPR 2004, Vol. 3138 of LNCS*, pages 1145–1154. Berlin: Springer, 2004.
12. E. Pękalska, P. Paclík, and R. P. W. Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research, Special Issue on Kernel Methods*, 2(12):175–211, 2001.

13. K. Schittkowski. Optimal parameter selection in support vector machines. *Journal of Industrial and Management Optimization*, 1(4):465–476, 2005.
14. B. Schölkopf, S. Mika, C. J.C. Burges, K.-R. Müller P. Knirsch, G. Rätsch, and A. J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 1999.
15. B. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London, 1986.
16. L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.