Generalized Augmentation of Multiple Kernels [⋆]

Wan-Jui Lee, Robert P.W. Duin and Marco Loog

Pattern Recognition Laboratory,
Delft University of Technology, The Netherlands

Abstract. Kernel combination is meant to improve the performance of single kernels and avoid the difficulty of kernel selection. The most common way of combining kernels is to compute their weighted sum. Usually, the kernels are assumed to exist in independent empirical feature spaces and therefore were combined without considering their relationships.
To take these relationships into consideration in kernel combination, we propose the generalized augmentation kernel which is extended by all the single kernels considering their correlations. The generalized augmentation kernel, unlike the weighted sum kernel, does not need to find out the weight of each kernel, and also would not suffer from information loss due to the average of kernels.
In the experiments, we observe that the generalized augmentation kernel usually can achieve better performances than other combination methods that do not consider relationship between kernels.

1 Introduction

The selection of kernel functions, the model, and the parameters, is one of the most difficult problem of designing a kernel machine. Recently, an interesting development seeks to construct a good kernel from a series of kernels. Most approaches in the literature aim to derive a weighted sum kernel, and the main concern is to find out the weight of each kernel [2, 4, 6, 8–10, 13, 18, 19]. In [10], semi-definite programming (SDP) was used to optimize over the coefficients in a linear combination of different kernels with respect to a cost function. The optimization worked in a transductive setting, and therefore all the information of training and testing patterns was required in the process. To prevent over-fitting, the search space of possible combined kernel matrices is constrained by bounding the kernel matrices with a fixed trace. If kernel target alignment is used as the cost function, this method could be seen as a generalization of the kernel matrix learning method in [7]. Instead of learning the combined kernel matrix in a transductive setting, hyperkernel methods [18] directly learned the combined kernel function in the inductive setting by minimizing a regularized risk functional. In these optimization methods, not only the cost function and constraints were considered, much effort was also spent to speed up the optimization procedure. Sometimes, a SDP problem might be able to cast to a second-order cone program (SOCP) [19] or quadratically constrained quadratic program (QCQP) [10] problem to achieve lower complexity. In [2], gradient decent was used to propose an even faster method. These methods mainly

[⋆] We acknowledge financial support from the FET programme within the EU FP7, under the SIMBAD project (contract 213250).

focus on finding the best weight of each kernel, and then perform the weighted sum of these kernels in order to derive a combined kernel. In these settings, local information is easily averaged out, and therefore these methods might suffer from information loss and the abilities of single kernels also become weaker. For example, if the dataset has varying local distributions, different kernels will be good for different areas. Averaging the kernels of such a dataset would lose some capability to describe these local distributions.

To unfold the local characteristics of data, [11, 12, 21] all proposed to augment s kernels of size $m \times m$ into a kernel of size $(s \times m) \times (s \times m)$. These methods all put the original kernel matrices on the diagonal. The main difference is on the off-diagonal. The methods proposed in [12, 21], which we name as the augmented kernel in short, only put zeros on the off-diagonal due to lacking of knowledge about the cross terms. In this case, it implies that different kernels live in the different subspaces and there is no interaction between these subspaces. Also, the empirical feature functions of different kernels are unrelated. This is a rather constrained assumption, and does not provide much flexibility. These cross terms are however, defined in [11] as the inner product of square roots of kernel functions to meet the mercer condition. Unfortunately, one can only derive the square root of a kernel function if the function is self-similar. Therefore, the only adoptable kernel for computing the composite kernel proposed in [11] is the Radial Basis function (RBF). Moreover, the cross terms of composite kernels are very similar to the product of two RBF kernels and therefore usually results in very small values. In most cases, it is very much the same as putting zeros on the off-diagonal.

In this work, we propose a method to augment single kernels into a generalized augmentation kernel by considering the cross terms on the off-diagonal and these cross terms can be derived from any type of kernel. It duplicates empirical feature functions on the off-diagonal with scaling parameters which indicate how related the empirical feature functions from two different kernels should be. This scaling parameter also allows us to smoothly vary between the original augmented kernel and the direct sum kernel, and therefore we can have a generalized description of multiple kernels and enlarge the searching space of the optimal solution. The experimental results suggest that the generalized augmentation kernel usually can find a better solution than the sum kernel, the composite kernel and the augmented kernel.

The rest of the paper is organized as follows. In Section 2, the overview of support vector machine is recaped. The direct sum of kernels and the augmented kernel and their empirical feature spaces are described in Section 3, respectively. Our proposal for constructing the generalized augmentation kernel from single kernels is given in Section 4. Simulation results are presented in Section 5. Finally, a conclusion is given in Section 6.

2 Overview of Support Vector Machine

For convenience, we introduce the support vector classifier with d input variables x_{i1}, x_{i2}, . . ., x_{id} for 2-class problem with class labels $+1$ and -1 in this section. $\mathbf{x}_i$ and y_i represent i^{th} input datum (a vector) and its corresponding class label [20, 3]. Extension to multi-class problems can be achieved by training multiple support vector machines.

To control both training error and model complexity, the optimization problem of SVM is formalized as follows:

$$\begin{aligned} \text{minimize } & \frac{1}{2} < \mathbf{w}, \mathbf{w} > + C \sum_{i=1}^{n} \xi_i, \\ \text{subject to } & < \mathbf{w} \cdot \mathbf{x}_i > + b \geq +1 - \xi_i, \text{ for } y_i = +1 \\ & < \mathbf{w} \cdot \mathbf{x}_i > + b \leq -1 + \xi_i, \text{ for } y_i = -1 \\ & \xi_i \geq 0, \forall i. \end{aligned} \tag{1}$$

By using Lagrange multiplier techniques, Eq.(1) could lead to the following dual optimization problem:

$$\begin{aligned} \text{maximize } & \textstyle\sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j < \mathbf{x}_i, \mathbf{x}_j >, \\ \text{subject to } & \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \in [0, C]. \end{aligned} \tag{2}$$

Using Lagrange multipliers, the optimal desired weight vector of the discriminant hyperplane is $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$. Therefore the best discriminant hyperplane can be derived as

$$f(\mathbf{x}) =< \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \mathbf{x} > + b = (\sum_{i=1}^{n} \alpha_i y_i < \mathbf{x}_i, \mathbf{x} >) + b, \tag{3}$$

where b is the bias of the discriminant hyperplane.

2.1 Empirical Feature Function of Kernels

In Eq.(3), the only way in which the data appears is in the form of dot products, that is $< \mathbf{x}_i, \mathbf{x} >$. The discriminant hyperplane is thereby linear and can only solve a linearly separable classification problem. If the problem is nonlinear, instead of trying to fit a nonlinear model, the problem can be mapped to a new space by a nonlinear transformation using a suitably chosen kernel function. The linear model used in the new space corresponds to a nonlinear model in the original space. To make the above model nonlinear, consider a mapping $\phi(\mathbf{x})$ from the input space into some feature space as

$$\phi : \mathbb{R}^d \to \mathcal{H}. \tag{4}$$

The training algorithm only depends on the data through dot products in $\mathcal{H}$, i.e. on functions of the form $< \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) >$. Suppose a kernel function K defined by

$$K(\mathbf{x}_i, \mathbf{x}_j) =< \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) >, \tag{5}$$

is used in the training algorithm. Explicit knowledge of ϕ is thereby avoided. The dot product in the feature space can be expressed as a kernel function. Similar to Eq.(3) in linear problems, for a nonlinear problem, we will have the following discriminant function

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b. \tag{6}$$

3 Sum Kernel And Augmented Kernel

Most kernel combination methods try to average out the kernel matrices in one way or another [10, 1, 9, 8, 14, 18]. Suppose s original kernels are given as $K_1, K_2, ...,$ and K_s with size $m \times m$ and the empirical feature functions of these kernels are $\phi_1(\mathbf{x}), \phi_2(\mathbf{x})...,$ and $\phi_s(\mathbf{x})$. Combining kernels by summing up all the kernels is equivalent to taking the Cartesian product of their respect empirical feature spaces. And weighing the kernels is the same as scaling the empirical feature spaces. To make use of all the local charac-

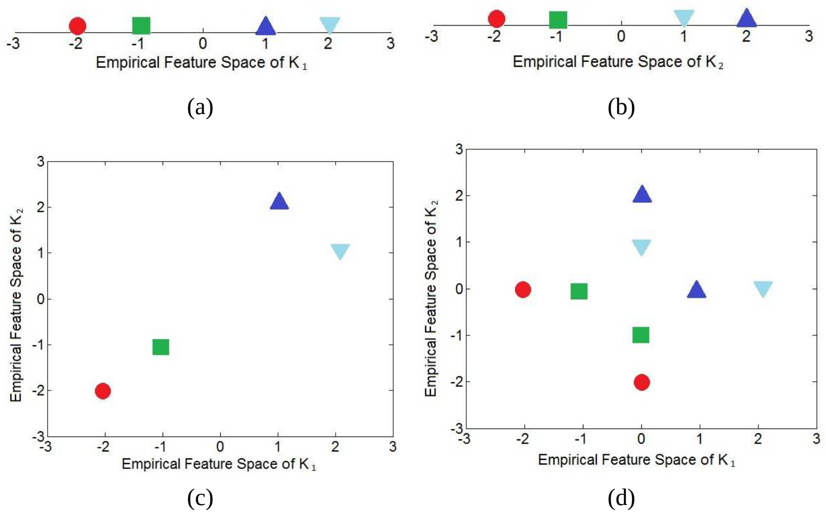

Fig. 1. Geometrical interpretation of (a) the empirical feature space of K_1 (b) the empirical feature space of K_2 (c) the empirical feature space of $K_1 + K_2$ (d) the empirical feature space of $K_1 \bigoplus K_2$.

teristics of each single kernel, the augmented kernel is proposed in [12, 21], which is of the form

$$K_1 \bigoplus K_2 \bigoplus \cdots \bigoplus K_s = \begin{pmatrix} K_1 & 0 & \cdots & 0 \\ 0 & K_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K_s \end{pmatrix}_{s\times m, s\times m.} \tag{7}$$

A direct sum kernel is with m coefficients while the augmented kernels is with $s \times m$ coefficients. Therefore, every object in every kernel can contribute during the training for augmented kernel.

By augmenting the empirical feature functions of kernels into one matrix $X_{\bigoplus}$ as

$$X_{\bigoplus} = \begin{pmatrix} \phi_1(\mathbf{x}) & 0 & \cdots & 0 \\ 0 & \phi_2(\mathbf{x}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \phi_s(\mathbf{x}) \end{pmatrix}_{s\times m, s\times m,} \tag{8}$$

one can show that the augmented kernel obeys the mercer theorem by taking the inner product of $X_{\bigoplus}$, that is $X_{\bigoplus}^T X_{\bigoplus} = K_1 \bigoplus K_2 \bigoplus \cdots \bigoplus K_s$.

Figure 1 is the illustration of the empirical feature spaces of direct sum kernel and the augmented kernel. From Figure 1(c) and Figure 1(d), we can see that instead of 4 objects as in the empirical feature spaces of the individual kernels and the direct sum kernel, there are duplicated objects (in total 8) in the empirical feature space of the augmented kernel. However, the augmented kernel assumes that all the empirical feature functions $\phi_1(\mathbf{x}), \phi_2(\mathbf{x})$..., and $\phi_s(\mathbf{x})$ are independent and uncorrelated. Therefore objects only live in their subspaces, and that is why we see two clear vertical and horizontal sets of objects in Figure 1(d). All the objects are therefore only allowed to be on the coordinates and the rest of the space is completely empty.

4 Generalized Augmentation Kernel in Multiple-Kernel Spaces

The augmented kernel assumes that the empirical feature spaces of kernels are all independent and uncorrelated. However, this assumption is not necessarily true, especially if kernels of the same type but different shapes are selected to combine. To combine kernels which are not necessarily uncorrelated, we first define the space which is expanded by these empirical feature functions to be the multiple-kernel space. In this space, the main components are not only these empirical feature functions but also the relationships among them. In the multiple-kernel space, the empirical feature functions are duplicated on the off-diagonal with scaling parameters which indicate how related the empirical feature functions from two different kernels should be. Thus, the multiple-kernel space $X_{\bigotimes}$ is in the following form:

$$X_{\bigotimes} = \begin{pmatrix} r_{11}\phi_1(\mathbf{x}) & r_{12}\phi_1(\mathbf{x}) & \cdots & r_{1s}\phi_1(\mathbf{x}) \\ r_{21}\phi_2(\mathbf{x}) & r_{22}\phi_2(\mathbf{x}) & \cdots & r_{2s}\phi_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ r_{s1}\phi_s(\mathbf{x}) & r_{s2}\phi_s(\mathbf{x}) & \cdots & r_{ss}\phi_s(\mathbf{x}) \end{pmatrix} \tag{9}$$

with size $s \times m, s \times m$ and r_{ij} is a parameter indicating how related $\phi_i(\mathbf{x})$ and $\phi_j(\mathbf{x})$ is. For simplicity, we assume that r_{11}, $r_{22}, ...,$ and r_{ss} are all equal to 1, and r_{ij} is equal to r_{ji}, for all i, j. By taking the inner product of $X_{\bigotimes}$, we can therefore define the generalized augmentation kernel $K_1 \bigotimes K_2 \bigotimes \cdots \bigotimes K_s$ as

$$\begin{pmatrix} r_{11}r_{11}K_1 + \cdots + r_{s1}r_{s1}K_s & r_{12}r_{11}K_1 + \cdots + r_{s2}r_{s1}K_s & \cdots & r_{1s}r_{11}K_1 + \cdots + r_{ss}r_{s1}K_s \\ r_{11}r_{12}K_1 + \cdots + r_{s1}r_{s2}K_s & r_{12}r_{12}K_1 + \cdots + r_{s2}r_{s2}K_s & \cdots & r_{1s}r_{12}K_1 + \cdots + r_{ss}r_{s2}K_s \\ \vdots & \vdots & \ddots & \vdots \\ r_{11}r_{1s}K_1 + \cdots + r_{s1}r_{ss}K_s & r_{12}r_{1s}K_1 + \cdots + r_{s2}r_{ss}K_s & \cdots & r_{1s}r_{1s}K_1 + \cdots + r_{ss}r_{ss}K_s \end{pmatrix}$$

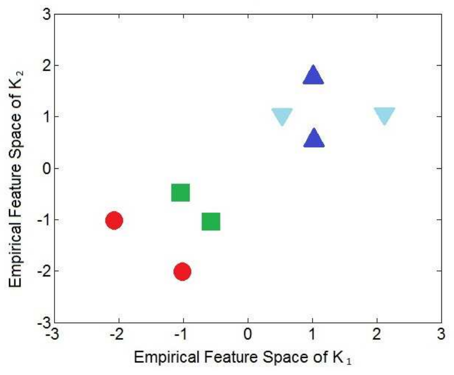

Fig. 2. Geometrical interpretation of the empirical feature space of $K_1 \bigotimes K_2$ with $r_{11} = 1$, $r_{22} = 1$, $r_{12} = 0.5$, and $r_{21} = 0.5$ given the empirical feature spaces of K_1 and K_2 as in Figure 1(a) and Figure 1(b).

with size $s \times m, s \times m$. Therefore the size of the generalized augmentation kernel matrix is $(s \times m) \times (s \times m)$ while the sizes of the original kernel matrices are $m \times m$. After the construction of the generalized augmentation kernel matrix, the support vector machine can proceed the learning of support vectors and their corresponding coefficients. Both training and testing objects have to be replicated as the generalized augmentation kernel matrix is s times larger than the base kernels.

The geometrical interpretation of the generalized augmentation kernel is given in Figure 2. By setting the degree of correlation between the empirical feature functions of two kernels as 0.5, the duplicated objects, unlike in the augmented kernel, can now position in different parts of the space as well. This obviously enlarges the searching space of optimal solutions for support vector machines. Moreover, the parameter r_{ij} allows us to smoothly vary between the original augmented kernel $(r = 1)$ and the direct sum kernel $(r = 0)$.

5 Experimental Results

In this section, we compare the experimental results obtained by our generalized augmentation kernel with those of other kernel combination methods and a classifier combination method. The kernel combination methods include the augmented kernel [12, 21], the direct sum kernel and the composite kernel [11]. The Fisher learning rule is used to derive the classifier combiner. One synthetic dataset and 7 benchmark datasets [15–17] are used in the experiments. To test whether the generalized augmentation kernel is more capable of describing data with different local distributions than the other kernel or classifier combination methods, two of the 8 datasets used in the experiments are with different local distributions, and the other 6 datasets are regular real datasets. The single kernel and the combined kernel SVM classifiers in the experiments are implemented with LIBSVM [5] and the classifier combiners are built with PRTOOLS

[17]. In every experiment, two RBF kernels are constructed, and kernel combination and classifier combination methods were used to combine these kernels or classifiers.

5.1 Experiment 1: Data With Varying Local Distributions

Spiral and sonar datasets are used in experiment 1. The SVM parameter C is set to 1 in all experiments to obtain a reasonable number of support vectors. The spiral dataset is a synthetic 2-dimensional dataset with 400 data patterns in 2 classes as shown in Figure 3. The sonar dataset contains information of 208 objects, 60 attributes, and two classes,

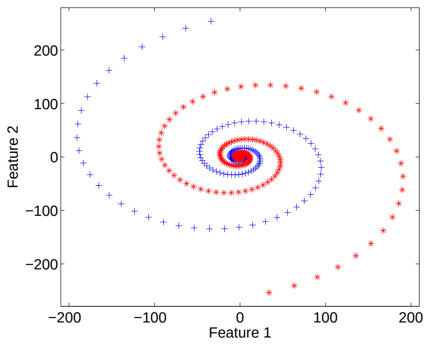

Fig. 3. Distribution of the spiral dataset.

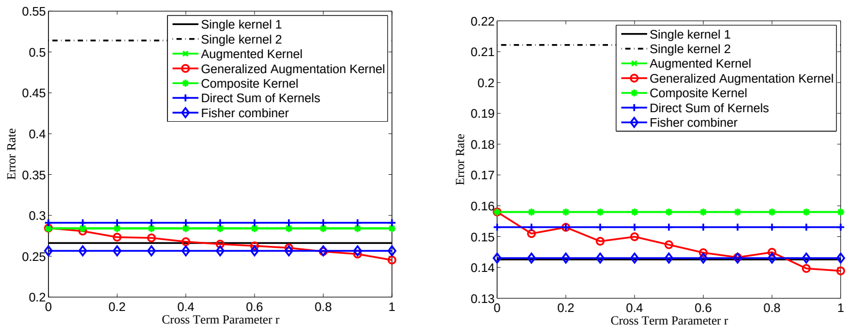

Fig. 4. Experiment results of (a) spiral dataset, and (b) sonar dataset with single kernels, generalized augmentation kernel, augmented kernel, composite kernel and the Fisher combiner.

rock and mine. The attributes represent the energy within a particular frequency band integrated over a certain period of time. In all the experiments, the dataset is randomly split into training and testing datasets with 80% and 20% ratio. For both datasets, two RBF kernels are built and different methods are used to combine these two kernels in each experiment. The results are averaged over 500 experiments. The sigma's of these single RBF kernels are assigned heuristically in the following way. The smallest sigma is the average distance of each data pattern to its nearest neighbor. The largest sigma is the average distance of each data pattern to its furthest neighbor.

The results for both single kernel classifiers, kernel combination methods and classifier combination methods with spiral dataset and sonar dataset are in Figure 4(a) and Figure 4(b), respectively. For these two datasets, the sum kernel, the augmented kernel and the composite kernel all have very similar performances. The Fisher classifier combiner is much better than kernel combination methods except for the generalized augmentation kernel under the situation that a high correlation factor is assigned during the construction of the generalized augmentation kernel. This suggests that the classifier combiner is more preferable than kernel combination methods when data is with varying local distributions, especially if the empirical feature spaces of the combined kernel is assumed to be uncorrelated.

Table 1. datasets.

dataset	biomed	IMOX	auto-mpg	heart	iris	wine
# features	5	8	6	13	4	13
# classes	2	4	2	2	3	3
# objects	194	192	398	297	150	178

5.2 Experiment 2: Benchmark Data

Table 2. Results of single kernels, augmented kernel, direct sum kernel, composite kernel and the Fisher combiner.

method	datasets					
	biomed	IMOX	auto-mpg	heart	iris	wine
	average error rate of 500 experiments					
single kernel 1	0.1245	0.0788	0.1704	0.3828	0.0692	0.2610
single kernel 2	0.1336	0.0794	0.1497	0.3041	0.0365	0.3147
direct sum kernel	**0.1206**	**0.0587**	0.1418	0.3346	0.0423	0.2803
augmented kernel	0.1362	0.0607	**0.1396**	**0.3258**	**0.0402**	0.2684
composite kernel	0.1362	0.0608	**0.1396**	**0.3259**	**0.0402**	0.2684
fisher combiner	0.1318	0.0893	0.1737	0.4242	0.0750	**0.2622**

Six real world datasets [15–17] as shown in Table 1 with the number of features, objects and classes, are used to have a more general investigation in experiment 2. The SVM parameter C is set to 1 in all experiments to obtain a reasonable number of support vectors. In all the experiments, the dataset is randomly split into training and testing datasets with 80% and 20% ratio. For all datasets, two RBF kernels are built with the heuristic mentioned above and different methods are used to combine these two kernels in each experiment, and the results are the averages of 500 repeated experiments. The results of all single kernel classifiers, kernel combination methods and classifier combination methods with all datasets are shown in Table 2. The results of the generalized augmentation kernel with different values of the cross term parameter r are given in Table 3.

From Table 2 and Table 3, we can see that given the right value to the parameter r, the generalized augmentation kernel can perform better than the best kernel combination (in bold) in all the datasets. Nevertheless, these r values seem to be dependant on the dataset and therefore there is a need to choose a good value beforehand. Also in these datasets, the classifier combiner is in general worse than the kernel combination methods.

Table 3. Results of the generalized augmentation kernel with different values of the cross term parameter r.

cross term parameter r	datasets					
	biomed	IMOX	auto-mpg	heart	iris	wine
	average error rate of 500 experiments					
0	0.1362	0.0607	0.1396	**0.3258**	0.0402	0.2684
0.2	0.1263	0.0576	0.1375	0.3370	0.0396	0.2491
0.4	0.1231	0.0524	**0.1365**	0.3388	0.0388	0.2435
0.6	0.1196	0.0458	0.1435	0.3339	0.0381	0.2377
0.8	0.1222	0.0449	0.1456	0.3416	0.0423	0.2390
1	**0.1189**	**0.0443**	0.1513	0.3580	**0.0367**	**0.2316**

6 Conclusions

In this study we proposed a method to construct the generalized augmentation kernel with multiple kernels. With the replication of empirical feature functions and adding a scaling factor to influence the correlation between kernels, we give more flexibility in positioning the duplicated objects in the combined empirical feature space.

However, this scaling factor, also called the cross term parameter, seems to be dependant on the dataset, and therefore it is more beneficial if this parameter can be well chosen before the process of learning. So far, we only choose the optimal parameter based on testing results and this is not really a practical solution. Moreover, our experiments only use two kernels so far in order to have a better insight on the cross term parameter. It is necessary, however, to include experiments with more kernels later on.

References

1. K. P. Bennett, M. Momma, and M. J. Embrechts, “MARK: A Boosting Algorithm for Heterogeneous Kernel Models,” in *Proc. 8th ACMSIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pp. 24-31, 2002.
2. O. Bousquet, and D. Herrmann, “On the Complexity of Learning the Kernel Matrix,” in *Proc. Advances in Neural Information Processing Systems*, pp. 415-422, 2003.
3. C. J. C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” *Knowledge Discovery and Data Mining*, vol. 2, no. 2, pp. 1-43, 1998.
4. G. Camps-Valls, L. Gomez-Chova, J. Muoz-Marí, J. Vila-Francés, and J. Calpe-Maravilla, “Composite Kernels for Hyperspectral Image Classification,” *IEEE Geoscience and Remote Sensing Letters*, Vol. 3, No. 1, pp. 93-97, 2006.
5. C. C. Chang and C. J. Lin, *LIBSVM : A Library for Support Vector Machines*, [http://www.csie.ntu.edu.tw/ cjlin/libsvm], Taiwan, National Taiwan University, 2001.
6. K. Crammer, J. Keshet, and Y. Singer, “Kernel Design Using Boosting,” in *Proc. of the Fifteenth Annual Conference on Neural Information Processing Systems*, 2002.
7. N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, “On Kernel Target Alignment,” *Technical Report NeuroColt*, 2001-099, Royal Holloway University, London, 2001.
8. I. M. de Diego, J. M. Moguerza, and A. Muoz, “Combining Kernel Information for Support Vector Classification,” in *Proc. Multiple Classifier Systems*, pp. 102-111, 2004.
9. G. Fung, M. Dundar, J. Bi, and B. Rao, “A Fast Iterative Algorithm for Fisher Discriminant Using Heterogeneous Kernels,” in *Proc. 21st Int. Conf. Machine Learning* , 2004.
10. G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, “Learning the Kernel Matrix with Semidefinite Programming,” *Journal of Machine Learning Research*, Vol. 5, pp. 27-72, 2004.
11. W.-J. Lee, S. A. Verzakov, and Robert P. W. Duin, “Kernel Combination Versus Classifier Combination,” in *Proc. Multiple Classifier Systems*, p.22-31, 2007.
12. C. T. Lin, C. M. Yeh, S. F. Liang, J. F. Chung and N. Kumar, “Support-Vector-Based Fuzzy Neural Network for Pattern Classification ,” *IEEE Trans. on Fuzzy Systems*, Vol. 14, No. 1, pp. 31-41, 2006.
13. C. A. Micchelli, and M. Pontil, “Learning the Kernel Function via Regularization,” *Journal of Machine Learning Research*, Vol. 6, pp. 1099-1125, 2005.
14. J. M. Moguerza, A. Muoz, and I. M. de Diego, “Improving Support Vector Classification via the Combination of Multiple Sources of Information,” SSPR/SPR, pp. 592-600, 2004.
15. A. Asuncion and D. J. Newman, UCI Machine Learning Repository [http://www.ics.uci.edu/mlearn/MLRepository.html], Irvine, CA: University of California, Department of Information and Computer Science, 2007.
16. A. K. Jain and M. D. Ramaswami, *Classifier design with Parzen window*, Pattern Recogition and Artificial Intelligence, Netherlands: Elsevier, 1988.
17. R. P. W. Duin, P. Juszczak, P. Paclik, E. Pękalska, D. de Ridder and D. M. J. Tax, *PRTOOLS4, A Matlab Toolbox for Pattern Recognition*, [http://www.prtools.org], the Netherlands, Delft University of Technology, Pattern Recognition Laboratory, 2004.
18. C. S. Ong, A. J. Smola, and R. C. Williamson, “Learning the Kernel with Hyperkernels,” *Journal of Machine Learning Research*, Vol. 6, pp. 1043-1071 , 2005.
19. I. W. H. Tsang, and J. T. Y. Kwok, “Efficient Hyperkernel Learning Using Second-Order Cone Programming,” *IEEE Trans. on Neural Networks*, Vol. 17, No. 1, pp. 48-58, 2006.
20. V. Vapnik, *The Nature of Statistical Learning Theory*, New York: Springer Verlag, 1995.
21. F. Yan, K. Mikolajczyk, J. Kittler, and M. A. Tahir, “Combining Multiple Kernels by Augmenting the Kernel Matrix,” in *Proc. Multiple Classifier Systems*, pp. 175-184, 2010.