Using Kernel Basis with Relevance Vector Machine for Feature Selection

Frédéric Suard and David Mercier

CEA, LIST, Laboratoire Intelligence Multi-capteurs et Apprentissage,
F-91191 Gif sur Yvette, France
{frederic.suard,david.mercier}@cea.fr

Abstract. This paper presents an application of multiple kernels like Kernel Basis to the Relevance Vector Machine algorithm. The framework of kernel machines has been a source of many works concerning the merge of various kernels to build the solution. Within these approaches, Kernel Basis is able to combine both local and global kernels. The interest of such approach resides in the ability to deal with a large kind of tasks in the field of model selection, for example the feature selection. We propose here an application of RVM-KB to a feature selection problem, for which all data are decomposed into a set of kernels so that all points of the learning set correspond to a single feature of one data. The final result is the selection of the main features through the relevance vectors selection.

Keywords: Relevance Vector Machine, Multiple Kernel, Kernel Basis, Feature Selection.

1 Introduction

During the last decade, Kernel Machines have been developed significantly, because of their ability to deal with a large variety of data, for example Support Vector Machines [12], kernel PCA [9] or kernel Fisher discriminant [7]. Approach involved by many kernel machines aims at defining a prediction function according to a weighted linear combination of kernel functions : $f(\mathbf{x}) = \sum_{i=1}^{n} w_i \cdot K(\mathbf{x}, \mathbf{x}_i) + w_0$. Among the cited algorithms, the SVM are particularly well known for their performance and their ability to face a large variety of dataset problems.

However, using a single kernel can be a limitation for some tasks, since all features are merged into a single kernel. To face this limitation, multiple kernel framework aims at using a set of kernels, instead of a single one, [5], by using a set of kernels. Lanckriet and Bach [1] have adapted this framework has been particularly to SVM thanks to composite kernel, by proposing to build a linear combination of kernels coming from various descriptors or a subset of data or a set of kernels with different parameters. The prediction function is then of the form : $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \sum_{j=1}^{k} \beta_j \cdot K_j(\mathbf{x}, \mathbf{x}_i) + b$.

A probabilistic model has also been developped to deal with such composite kernel [3], which shows the interest of such framework.

C. Alippi et al. (Eds.): ICANN 2009, Part II, LNCS 5769, pp. 255–264, 2009.

However, using a composite kernel have some restrictions, since the same set of kernels is used for all vectors of the solution, so that we can not merge global and local kernels to define the solution.

We consider here an other definition of multiple kernel, namely the Kernel Basis, which has been adapted recently to the *Least Angle Regression Stepwise* algorithm [2] by Guigue et al. [13]. The aim is to define the original kernel by concatenating a set of kernels, so that the algorithm can adapt a specific subset of kernels for each vector of the solution. Compared with the composite kernel, the kernel basis is able to deal both with local and global kernel. The prediction function of the kernel basis is then defined by : $f(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{k} w_{i,j} \cdot K_j(\mathbf{x}, \mathbf{x}_i) + w_0$.

But the LARS algorithm imposes a regularization parameter which helps to tune the complexity of the solution. The interest of such parameter is to limit the number of vectors, but can also impose a cross validation step.

In this paper we are looking at the Relevance Vector Machine algorithm proposed by Tipping et al. [11]. This method is based on a Bayesian approach which expects to maximize the distribution probability according to the linear combination of the relevance vectors. We propose here to define a multiple kernel like Kernel Basis, and to adapt it to the RVM algorithm. The interest is to go deeper in the model selection task, by applying a specific kernel for each relevance vector, so that we can modeling the data distribution with more efficiency.

To show the interest of our approach, we will present some preliminary results obtained for a feature selection task. The set of kernels is composed here by computing one kernel for each feature of each data. The feature selection is then operated by selecting the best kernels. We will compare the performance obtained on three different benchmark datasets with the state of the art. Results will show that the RVM-Kernel Basis is very promising, since it performs as well as SVM composite kernel but implies also a more sparse solution. The sparsity of the solution is a major advantage in such field of real time computation, for example. A sparse solution has also a higher generalization capacity.

2 Relevance Vector Machine

In the first part, we will present briefly the RVM algorithm and the way to apply a multiple kernel strategy. The relevance vector machine is a probabilistic sparse kernel model that has been introduced by M. Tipping in 2000 [11,14].

The aim is to reveal the underlying distribution of a set of data $\{\mathbf{x}_i, y_i\}_{i=1\ldots n}$, where $\mathbf{x} \in \mathbb{R}^d$. Each data is associated with a label y defined by $p(y|\mathbf{x}) \sim \mathcal{N}(f(\mathbf{x}), \sigma^2)$, with the standard deviation coming from the addition of a gaussian noise : $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The decision function is of the form :

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i \cdot \Phi(\mathbf{x}, \mathbf{x}_i) + w_0 \tag{1}$$

with Φ a kernel function, $\mathbf{w}$ the coefficient associated to each support vector.

So we can rewrite the probability of the data according to the parameters :

$$p(\mathbf{y}|\mathbf{w},\sigma^2) = \frac{1}{(\sigma\sqrt{2\pi})^n} \exp^{-\frac{1}{2\sigma^2}\|\mathbf{y}-\Phi\mathbf{w}\|^2} \tag{2}$$

with Φ a $n \times (n+1)$ matrix containing the kernel and a bias : $\Phi = \begin{bmatrix} 1 & \\ \vdots & K \\ 1 & \end{bmatrix}$.

The key of this approach is to define a prior on each coefficient w_i. According to the *Automatic Relevance Determination*[6] mechanism, all coefficient which are unecessary are pruned. This mechanism can explained the sparsity of the solution, since it prunes all parameters that add complexity to the probabilistic model. By pruning coefficients, the likelihood is then maximised regarding the input data. For further information the reader can refer to [11], this paper also presents the adaptation of this algorithm for the classification case.

3 Extension of RVM to Multiple Kernel - Kernel Basis

A recent approach in SVM pushed by Lankriet and Bach [1] , introduced the notion of multiple kernel learning, more precisely a composite kernel, by constituting a set of kernels, where each kernel has been obtained with different kernel formulation or parameters. The prediction function is then written :

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \sum_{j=1}^{k} \beta_j \cdot K_j(\mathbf{x}, \mathbf{x}_i) + b. \tag{3}$$

The solution is build around a composite kernel, that is to say that only one kernel is applied to all vectors. This approach has been proposed to tackle the descriptor fusion problem, by merging in a single kernel a set of kernels coming from different descriptors. It can also solve the problem of kernel parameter setting, since we can propose kernels from the same data but with a different parametrization.

However, this method assigns the same kernel for all support vectors without taking into account the impact of local kernels. Our problematic is to be able to assign specific kernel for each vector, which can be written:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{k} w_{i,j} \cdot \Phi_j(\mathbf{x}, \mathbf{x}_i) + w_0. \tag{4}$$

Instead of using a couple of weighting coefficients ($\mathbf{w} \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^d$), this formulation is based on a weighting coefficient $w \in \mathbb{R}^{n \times d}$. We can then apply for each vector a specific set of kernels.

The difference between composite kernel and kernel basis is illustrated on figure 1, where one can see that the kernel basis is able to assign for each vector a specific kernel which considers only one feature (RV 1 and 2) or two features

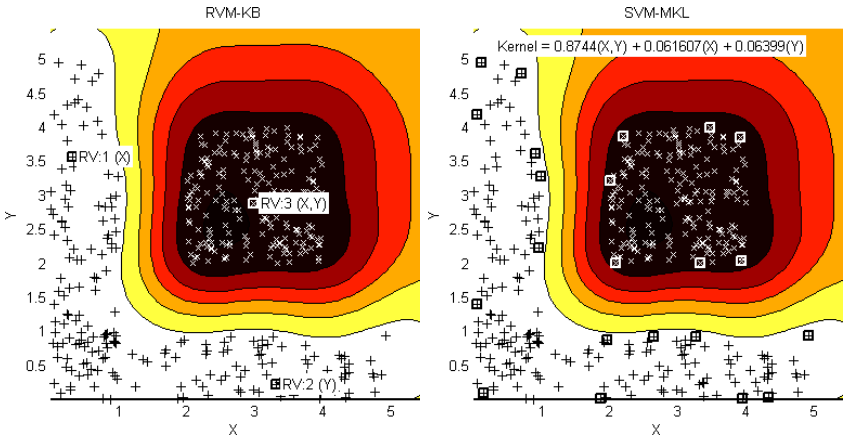

Fig. 1. This example shows the difference between RVM Kernel Basis (left) and SVM Composite Kernel (right). The Kernel Basis is able to adapt a specific kernel for each vector, namely a kernel based on 'x', a second on 'y' and the third combines 'x' and 'y'. The composite kernel aims at defining a unique kernel for all vectors.

(RV3), whereas a composite kernel can only consider the same kernel for all vectors.

One possible way to write this function is to define a kernel basis as described in [4,13]. This definition aims at decompose the kernel Φ into different blocks. The multiple kernel is then composed like a Kernel Basis :

$$\Phi = \begin{bmatrix} 1 \\ \vdots \; K_1 \; K_2 \ldots \; K_k \\ 1 \end{bmatrix}, \tag{5}$$

for k kernels when a bias is added. This matrix has a size of $n \times (n \times k)$, so we can associate a weight w_i with each column. If we consider that all columns are independant, we can finally write the prediction function with :

$$f(\mathbf{x}) = \sum_{i=1}^{n*k} w_i \cdot \phi_i(\mathbf{x}) + w_0. \tag{6}$$

This formulation shows that we can deal with relevance vectors, each one defined by at least one kernel function. Since the formulation of Tipping can deal with non-Mercer kernels, the extension of RVM to multiple kernel is well adapted with a Kernel Basis approach.

4 Experimental Results

In the last section, we will present some results to illustrate our approach. We propose here to use the RVM-Kernel Basis for a task of feature selection. Starting

from a dataset $\{\mathbf{x}_i, y_i\}_{i=1..n}$ of n data $\mathbf{x} \in \mathbb{R}^d$, the aim is to build a kernel Φ which has a size of n rows and $n \times d$ columns. Applying the RVM algorithm helps us to select the vectors to build the solution.

The more pertinent features are those corresponding to the selected vectors, so that we can simply count the number of times a features has been selected.

We compare the performance with a composite kernel approach, namely the SVM-MKL. The kernel set is constituted with d kernels of size $n \times n$. The feature selection is operated here by considering the value of the weighting coefficients β (see eq. 3) associated with each kernel in the linear combination. The weight of a feature is directly linked with the β value. We are using the SVM-MKL implementation for Matlab proposed by Rakotomamonjy et al. [8], that we already used for a task of image descriptors fusion [10].

We are using some datasets available on the UCI website (`http://archive.ics.uci.edu/ml/`) which are frequently used for machine learning benchmark : Boston Housing, Auto Mpg and Blood transfusion.

For each dataset, we first evaluated the performance of a single kernel, that is to say when only one kernel is used for all features. We tested different parameters for the kernel, namely a gaussian kernel for different bandwidths and a polynomial kernel of different orders. So that we obtained a reference to assess the multiple kernel approach. We carried a 4 fold cross validation for each kernel parameter.

In order to make our results comparable, we organized randomly the cross validation but used the same data for all tests. Concerning the multiple kernel, we set the same kernel parameter for all features, which can be considered as sub-optimal, but since our data are normalized to obtain a mean of 0 and a standard deviation of 1, this compromise is justified. At the moment, we are limited by the memory aspect of such algorithms, but this aspect is a perspective for our future works, by integrating the kernel parameter choice during the optimization.

Since the SVM algorithm involves some parameters within the optimization procedure, we have to set the weight of the misclassified points of the learning set (C), and the width of the regression tube (ϵ). We tried an exhaustive list of values for each parameter and retained the parameters corresponding to the best performance.

To estimate the performance, we have computed the mean squared error, for the regression case, and the AUC value, that is to say the area under the roc curve for the classification case.

The multiple kernel set is built differently in SVM and RVM. For the SVM, we define one kernel for each feature (which produces a kernel of size $n \times n \times d$), so that the SVM will select a subset of kernels and a set of support vectors. For the RVM the kernel basis has an original size of $n \times (n \times d)$, and the algorithm selects a subset of vectors among the $n \times d$ available. This can explained the fact that the percentage of relevance vectors have no correspondence between the single and kernel basis cases.

Auto MPG. The first dataset : AUTO MPG estimates the pollution emission of a car, by considering some mechanical characteristics of a vehicle. The database contains 398 cars, each one described by 7 features.

We have reported in table 1 the performance obtained for RVM and SVM approach for both single and multiple kernel. We report only the performance corresponding to the best parameter set (C, ϵ) for the SVM.

One can see that when using a single kernel, the SVM obtained a better score with an error of 2.74 against 2.84 for the RVM. When we use a multiple kernel, both algorithms improved their performance, with a significant improvement for the RVM-KB, which reached an error of 2.7.

However, we have to note that, due to its sparse solution, the RVM requires only 101 vectors, that is to say 101 scalars, to build the solution. The SVM requires a larger solution that implies 6 kernels and 292 points, that is to say 1752 scalars. In terms of percentage, the RVM retains 4.2% of the learning set against 97% for the SVM. The RVM percentage is computed as follows : $\frac{\#RV}{\#Data \times \%CrossValidation \times \#kernels} = \frac{101}{398 \times 3/4 \times (7+1)}$.

Table 1. Performance obtained for AUTO dataset. We report here for each algorithm, for both single and multiple kernel the error and the size of the solution, according to different kernel parameter setting.

AUTO	RVM				SVM				
	Single Kernel		Kernel Basis		Single Kernel $\epsilon = 0.2$, $C = 100$		Composite Kernel $\epsilon = 0.1$, $C = 10$		
Kernel	error	#RV (%)	error	#RV (%)	error	#SV (%)	error	#SV (%)	#kernels
gaussian 0.1	24.69	82 (27.55%)	3.14	1128 (51.4)	7.68	293 (98.16)	3.11	293 (98.2)	7
gaussian 0.5	8.99	174 (58.29%)	3.06	549 (23)	4.00	279 (93.4)	2.84	291 (97.4)	7
gaussian 1	4.47	64 (20.85%)	2.74	352 (14.8)	3.34	273 (91.46)	**2.70**	292 (97.8)	6
gaussian 2	**2.84**	20 (6.70%)	2.74	222 (9.3)	2.78	272 (91.12)	2.8	293 (98.4)	1
gaussian 3	2.86	18 (6.20%)	2.77	186 (7.8)	**2.74**	272 (91.12)	2.82	294 (98.6)	1
gaussian 5	2.97	13 (4.27%)	2.9	664 (27.8)	2.79	268 (89.78)	2.9	294 (98.6)	1
gaussian 10	3.20	18 (6.20%)	2.86	175 (7.3)	2.89	277 (92.80)	3.25	292 (97.8)	1
gaussian 20	5.20	8 (2.76%)	2.84	254 (10.6)	3.27	283 (94.81)	3.78	294 (98.6)	1
poly 1	23.80	1 (0.34%)	23.8	0.75 (0.03)	3.46	280 (93.80)	3.47	295 (98.8)	1
poly 2	2.93	18 (6.20%)	2.83	118 (4.9)	2.82	273 (91.46)	2.81	294 (98.6)	1
poly 3	3.40	44 (14.74%)	**2.70**	101 (4.2)	5.48	275 (92.13)	4.96	294 (98.6)	1

We have detailed on table 2 the mutiple kernel solution for RVM and SVM. We have reported the number of relevance vectors for the RVM and the total weight according to each feature. We simply added all the weights for all vectors using a given feature. Concerning the SVM, we report the weighting coefficient of each kernel of the composite kernel. This coefficient is directly linked to the feature pertinence. As we can see, SVM put a more important weight to the last kernel which contains all features ($\beta = 0.75$), but we can observe some similarities by comparing RVM-KB and SVM-MKL. For example, kernels 5 and 7 are neglicted in both cases, whereas kernels 1, 3 and 4 have a major weight. Considering the features 5 and 7 : acceleration and origin, this selection seems

Table 2. Auto MPG : details of the RVM-KB and SVM-MKL solution. For each feature, we report the number of relevance vectors and the cumulated weights, and for the SVM-MKL we detail the weight of each kernel for the linear combination of kernels.

		RVM-KB		SVM-MKL
kernel	Feature(s)	#RV	$\sum_i \|w_i\|$	β
1	# cylinders	18.2500	0.1886	0.0081
2	displacement	13.2500	0.0596	0.0323
3	horsepower	10.2500	0.2772	0.0289
4	weight	7.7500	0.2242	0.1053
5	acceleration	1.0000	0.0029	0
6	model year	38.2500	0.2147	0.0724
7	origin	0	0	0
8	[1-7]	12.2500	0.0328	0.7530

to be valuable, since the objective is to estimate the consumption of the car, so the other attributes seem to be more significant, namely the weight, horsepower or cylinders.

Boston Housing. The second dataset : Boston Housing, has been originally performed in the RVM article of Tipping. This regression task aims at predicting the value of a residential house by considering a set of features describing the environment, local supplies or roads. 506 prices are given, each one described by 13 features.

As we said above, we tried different kernel parameters to estimate the performance with a single kernel. We then accomplished the same procedure with the multiple kernel approach.

We report the results on the table 3 for both single and multiple kernel. Using only a single kernel, SVM performs better than RVM with an error of 3.27 against 3.81. For the multiple kernel approach, both SVM and RVM increase their performance, but the RVM has a major gain and reached an error of 3.22. Another point is the sparsity of the solution, which is also in favor of the RVM due to the lower number of points necessary : 322 scalars for the RVM against 375×7 for the SVM.

We have also reported the details of the multiple kernel solution in table 4, which clearly shows that both algorithms are able to select the more significant

Table 3. Best performance for single and multiple kernel for both RVM and SVM. For each case, we report the kernel parameter, the performance (error) and the size of the solution.

BOSTON	Kernel	error	#RV (%)
RVM	Gaussian 5	3.81	30 (7.8)
RVM-KB	Gaussian 2	3.22	322 (6.1)

BOSTON	Parameters	Kernel	error	#SV (%)
SVM	$\epsilon = 0.05$,$C = 100$	Gaussian,3	3.27	373 (98)
SVM-MKL	$\epsilon = 0.01$,$C = 10$	Gaussian,1	3.26	375 (98)

Table 4. Details of the RVM-KB and SVM-MKL solution for the Boston dataset. For each feature, we report the number of relevant vectors and the cumulated weights, and for the SVM-MKL we detail the weight of each kernel for the linear combination of kernels.

		RVM-KB		SVM-MKL
Kernel	Feature(s)	#RV	$\sum_i \lvert w_i \rvert$	β
1	Criminal rate	5.7500	0.0050	0.0040
2	Residential %	74.2500	0.0005	0
3	Industrial %	13.0000	0.0014	0
4	River distance	11.7500	0.0006	0
5	% NOx	16.0000	0.0037	0.0104
6	# rooms	15.0000	0.0035	0.1271
7	Age	3.5000	0.0008	0
8	Distance	0	0	0
9	Road	61.0000	0.0200	0
10	Tax	21.2500	0.0027	0.0181
11	Teachers	10.7500	0.0016	0.0185
12	% blacks	1.2500	0.0003	0
13	Lower salary	14.7500	0.0085	0.0939
14	[1-13]	73.7500	0.9513	0.7281

features. To assess efficiently the multiple kernel, we added to the set of kernels a kernel computed from all data. We have to precise that the bandwidth of this kernel is applied for each element of the vector, so that the bandwidth is comparable between single feature kernels and all features kernels. It is interesting to note that SVM and RVM obtained some similarities beyond their selection, since the last kernel (14 containing all features) obtained an important weight, so that it can be assimilated as a single kernel, but the performance grows up because of adding some single features.

In terms of feature selection, the SVM-MKL selected 7 kernels out of the 14 originals, and the RVM retained 9 main kernels. The 5 last kernels got a minor weight (namely the kernels 2, 4, 7, 8 and 12). On top of that, SVM totally removes kernels 2, 3, 4, 7, 8, 9 and 12. We can then notice the interest of RVM-KB, which is able to add minor kernels, that is to say local kernels, in opposition to the SVM which generally prefers to suppress this kind of kernel ($\beta = 0$).

Blood transfusion. The last experiment is a classification task based on the Blood Transfusion dataset. The aim of such database is to predict if a person has donated or not its blood in March 2007. The set contains 748 persons which are characterized by 4 features, concerning the frequency and the volume.

Since it is a classification problem, we give the performance by considering the area under the roc curve (AUC). We also report the size of the solution.

As shown on table 5, the RVM algorithm performs better than the SVM, in both single and multiple kernel case. We have to note that the SVM-MKL does not really takes benefit from the multiple kernel approach, since the performance is not increasing at all. This can be explained by the fact that all features are

Table 5. Performance obtained on the dataset BLOOD. We give the AUC value for the best kernel parameter.

BLOOD	Kernel	AUC	#RV (%)
RVM	gaussian 10	0.744	3 (0.54)
RVM-KB	poly 3	0.75	247 (11.02)

BLOOD	Parameter	Kernel	AUC	#SV (%)
SVM	$C = 10$	gaussian 10	0.718	299 (55)
SVM-MKL	$C = 100$	linear	0.71	378(67)

necessary, as shown on table 6. On top of that, the RVM is sparse, since for a single kernel only 3 vectors are necessary. However, the multiple kernel lost this sparsity, since 247 vectors are used. It should be noticed that in this case, it means 247 scalars, since we have originally 2241 vectors in the learning set, one vector beeing associated with one feature only. The sparsity can be explained by the fact that this set seems to have few overlap, considering also the SVM that retains around 50% of the learning set.

Table 6. Details of the RVM-KB and SVM-MKL solution for the Blood dataset. For each feature, we report the number of relevant vectors and the cumulated weights, and for the SVM-MKL we detail the weight of each kernel for the linear combination of kernels.

BLOOD		RVM-KB		SVM-MKL
Kernel	Feature	#RV	$\sum_i \lvert w_i \rvert$	β
1	Recency	22	0.6678	0.2484
2	Frequency	104	0.7228	0.1983
3	Monetary	104	0.7228	0.2500
4	Time	16	0.6656	0.2499

Talking about feature selection, this experiment is interesting because both RVM-KB and SVM-MKL made the same selection. As shown on table 6, all features are necessary to build the solution, and both give a similar weight to all features. It means that all features are relevant for this task.

This last experiment also point out the fact that one have to take care when building the kernel set. We have to pay attention to the fact that computing each kernel from a single feature can fail, since we loose dependancy between features. At the moment, we have to constitute the set of kernels for each feature combination. For example, if we have two features x_1 and x_2, the kernel set is composed of $K(x_1, .)$, $K(x_2, .)$ and $K(x_{12}, .)$. This point is a particular perspective, that we propose to face in a near future.

5 Conclusion

In this paper, we proposed to extend the Relevance Vector Machine to multiple kernel as a Kernel Basis. The interest is to build a solution which combines both local and global kernels, so that each vector uses a specific set of kernels, whereas a composite kernel can only deal with the same kernel for all vectors.

This approach is very useful for model selection tasks, like kernel parameter setting or feature selection. The results we obtained on such problem have shown that this approach is very promising since we obtained equivalent and sometimes best performance compared to SVM using a composite kernel, with a major advantage concerning the size of the solution. Actually, due to the sparsity of RVM, the solution is really smaller, which is very interesting by considering real time application, for example. The sparse aspect has also other interests that we propose to exploit in a near future concerning the explanation of the solution.

The main drawback of the RVM-KB, resides in the computational complexity, since we have to face a large matrix inversion, so that we can have a limitation. However, due to the parametrization of the SVM, learning time is comparable since we have to test different values for the SVM parameters. A last perspective is then linked with the large dataset, that we will have to face.

References

1. Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the smo algorithm. In: ICML 2004: Proceedings of the twenty-first international conference on Machine learning, p. 6. ACM Press, New York (2004)
2. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression, pp. 407–499 (January 2003)
3. Girolami, M., Rogers, S.: Hierarchic bayesian models for kernel learning. In: 22^{nd} International Conference on Machine Learning, pp. 241–248 (2005)
4. Guigue, V., Rakotomamonjy, A., Canu, S.: Kernel basis pursuit. In: CAP, pp. 93–106 (2005)
5. Gunn, S., Kandola, J.: Structural modelling with sparse kernels. Machine Learning 48, 137–163 (2002)
6. Mackay, D.J.: Probable networks and plausible predictions - a review of pratictal bayesian methods for supervised neural networks, vol. 6, pp. 469–505 (1995)
7. Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A.J., Mueller, K.-R.: Constructing descriptive and discriminative non-linear features: Rayleigh coefficients in kernel feature spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence (2004)
8. Rakotomamonjy, A., Bach, F., Canu, S., Grandvalet, Y.: Simple MKL. Journal of Machine Learning Research (2008)
9. Schölkopf, B., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge (2002)
10. Suard, F., Rakotomamonjy, A., Bensrhair, A.: Model selection in pedestrian detection using multiple kernel learning. In: Intelligent Vehicles Symposium 2007, Istanbul (June 2007)
11. Tipping, M.: The relevance vector machine. In: Solla, T.K.L.S.A., Müller, K.-R. (eds.) Advances in Neural Information Processing Systems, vol. 12. MIT Press, Cambridge (2000)
12. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, N.Y. (1995)
13. Vincent, P., Bengio, Y.: Kernel matching pursuit. Mach. Learn. 48(1-3), 165–187 (2002)
14. Wu, L., Schölkopf, B., Bakir, G.: A direct method for building sparse kernel learning algorithms. Journal of Machine Learning Research 7, 603–624 (2006)