14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

A GREEDY ALGORITHM FOR OPTIMIZING THE KERNEL ALIGNMENT AND THE PERFORMANCE OF KERNEL MACHINES

Jean-Baptiste Pothin, Cédric Richard

Institut des Sciences et Technologies de l'Information de Troyes (ISTIT-M2S, FRE CNRS 2732)
Université de Technologie de Troyes, 12 rue Marie Curie, BP 2060, 10010 Troyes cedex - France
jean_baptiste.pothin@utt.fr cedric.richard@utt.fr

ABSTRACT

Kernel-target alignment has recently been proposed as a criterion for measuring the degree of agreement between a reproducing kernel and a learning task. It makes possible to find a powerful kernel for a given classification problem without designing any classifier. In this paper, we present an alternating optimization strategy, based on a greedy algorithm for maximizing the alignment over linear combinations of kernels, and a gradient descent to adjust the free parameters of each kernel. Experimental results show an improvement in the classification performance of support vector machines, and a drastic reduction in the training time.

1. INTRODUCTION

The last ten years have seen an explosion of research in kernel methods; see [1] for a recent survey. These include support vector machines (SVM), which map data into a high dimensional space where the classes of data are more readily separable, and maximize the margin – or distance – between the separating hyperplane and the closest points of each class. However, despite the success of kernel machines, the selection of an appropriate kernel is still critical for achieving good generalization performance. A typical approach for kernel selection involves the following steps: choose some kernels before learning starts, estimate their performance from cross-validation experiments, and pick the best one. This strategy becomes intractable as the number of kernels increases.

Observing that all the information required by a kernel machine is contained in the so-called Gram matrix, a recent work has suggested to learn it from data [2]. Another interesting solution was developed through the concept of kernel-target alignment [3]. In this reference and in [4], the approaches for optimizing this criterion are limited to a transductive setting. The kernel matrices are of the form

$$\boldsymbol{K} = \sum_i \mu_i \boldsymbol{v}_i \boldsymbol{v}_i^t, \quad \mu_i \geq 0, \tag{1}$$

where the $\boldsymbol{v}_i$'s are the eigenvectors of the full kernel matrix constructed from training and test samples. An inductive procedure was also proposed in [4], based on the eigendecomposition of the training kernel matrix. Unfortunately, these subspace methods become extremely computationally expensive when dealing with large kernel matrices. A more efficient approximation strategy based on the Gram-Schmidt decomposition and a quadratic programming method (QP) was presented in [5]. A semi-definite programming approach (SDP) was also considered in [6]. The authors, however, concede that the SDP applied to the kernel matrix (1) boils down to the quadratic program described in [3]. In this paper, we propose an alternating optimization strategy, based on a greedy algorithm for maximizing the alignment over linear combinations of kernels, and combined with a gradient descent to adjust the free parameters of each kernel.

The rest of this paper is organized as follows. In Section 2, kernel-target alignment is introduced. Our fast algorithm for optimizing this criterion is presented in Sections 3 and 4. Its effectiveness is confirmed through simulations in Section 5. Finally, concluding remarks and suggestions follow.

2. KERNEL-TARGET ALIGNMENT

We start with a few basic definitions. Let $\mathcal{X}$ be a compact space. A symmetric function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ verifying

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$$

for all $n \in \mathbb{N}$, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathcal{X}$ and $a_1, \ldots, a_n \in \mathbb{R}$ is said to be a *Mercer kernel*. An explicit way to describe it is via a mapping ϕ from $\mathcal{X}$ to a reproducing kernel Hilbert space $\mathcal{H}$

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}.$$

The alignment criterion is a measure of similarity between two kernels, or between a kernel and a target function [3]. Given a n-sample data set $\mathcal{S}_n$, the alignment of kernels κ_1 and κ_2 is defined as follows

$$\mathcal{A}(\boldsymbol{K}_1, \boldsymbol{K}_2) = \frac{\langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F}{\sqrt{\langle \boldsymbol{K}_1, \boldsymbol{K}_1 \rangle_F \langle \boldsymbol{K}_2, \boldsymbol{K}_2 \rangle_F}}, \tag{2}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product, and $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ are the Gram matrices with respective entries $\kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\kappa_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for all $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}_n$.

For binary classification, the decision statistic should satisfy $\phi(\boldsymbol{x}_i) = y_i$, where y_i is the class label of $\boldsymbol{x}_i$. By setting $y_i = \pm 1$, the ideal Gram matrix would be given by

$$\boldsymbol{K}^*(i,j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle = \begin{cases} 1 & \text{if } y_i = y_j \\ -1 & \text{if } y_i \neq y_j. \end{cases} \quad (3)$$

In [3], Cristianini *et al.* propose to maximize the alignment with the target $\boldsymbol{K}^* = \boldsymbol{y}\boldsymbol{y}^t$ in order to determine the most relevant kernel for a given classification task.

$$\mathcal{A}(\boldsymbol{K}, \boldsymbol{K}^*) = \frac{\langle \boldsymbol{K}, \boldsymbol{y}\boldsymbol{y}^t \rangle_F}{\sqrt{\langle \boldsymbol{K}, \boldsymbol{K} \rangle_F \langle \boldsymbol{y}\boldsymbol{y}^t, \boldsymbol{y}\boldsymbol{y}^t \rangle_F}} = \frac{\boldsymbol{y}^t \boldsymbol{K} \boldsymbol{y}}{n \, \|\boldsymbol{K}\|_F}. \quad (4)$$

The ease with which this criterion can be estimated using only training data, prior to any computationally intensive training, makes it an interesting tool for kernel selection. It has been shown that the alignment is *concentrated*, ie the probability of the empirical estimator (4) deviating from its mean can be bounded by an exponentially decaying function of this deviation [3]. This means that if one optimizes the alignment on a training set, one can expect it to remain high on a validation set. It has also been demonstrated that $h(\boldsymbol{x}) = \text{sgn}(\mathbb{E}_{\boldsymbol{x}',y'}[y' \kappa(\boldsymbol{x}', \boldsymbol{x})])$ has good generalization performance when the alignment is high.

3. OPTIMIZING THE LINEAR COMBINATION OF KERNELS BY A GREEDY APPROACH

Any positive linear combination of Mercer kernels is a Mercer kernel [7]. Given a collection of m kernels, we then consider the kernel expansion

$$\kappa_{\boldsymbol{\mu}}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{m} \mu_i \, \kappa_i(\boldsymbol{x}, \boldsymbol{x}'), \quad \mu_i \geq 0, \quad (5)$$

and study the problem of determining the parameters μ_i that maximize the kernel-target alignment. For a classification task, this problem can be written as [4]

$$\begin{array}{ll} \max_{\boldsymbol{\mu}} & -\boldsymbol{\mu}^t (\boldsymbol{H} + \lambda \boldsymbol{I}) \boldsymbol{\mu} + \boldsymbol{f}^t \boldsymbol{\mu} \\ \text{subject to} & \mu_i \geq 0, \text{ for all } i = 1, \ldots, m \end{array} \quad (6)$$

with $\boldsymbol{I}$ denoting the identity matrix, $\boldsymbol{H}(i,j) = \langle \boldsymbol{K}_i, \boldsymbol{K}_j \rangle_F$ and $\boldsymbol{f}(i) = \langle \boldsymbol{K}_i, \boldsymbol{K}^* \rangle_F$. The parameter $\lambda \geq 0$ arises from a regularization constraint penalizing $\|\boldsymbol{\mu}\|^2$. As for SVM optimization, large values of m make the resolution of equation (6) time and memory consuming with standard QP methods. Our strategy to handle this problem is to divide it into subproblems. When applied to (6) with $m = 2$, this leads to the maximization of

$$\begin{aligned} &W(\mu_1, \mu_2) = \\ &-\mu_1^2 (\|\boldsymbol{K}_1\|_F^2 + \lambda) - \mu_2^2 (\|\boldsymbol{K}_2\|_F^2 + \lambda) \\ &-2 \, \mu_1 \, \mu_2 \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F + \mu_1 \, \langle \boldsymbol{K}_1, \boldsymbol{K}^* \rangle_F + \mu_2 \, \langle \boldsymbol{K}_2, \boldsymbol{K}^* \rangle_F \end{aligned}$$

subject to $\mu_1, \, \mu_2 \geq 0$. Optimality conditions $\partial W / \partial \mu_1 = 0$ and $\partial W / \partial \mu_2 = 0$ yields

$$\begin{aligned} (\|\boldsymbol{K}_1\|_F^2 + \lambda) \mu_1 + \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F \mu_2 &= \tfrac{\langle \boldsymbol{K}_1, \boldsymbol{K}^* \rangle_F}{2}, \\ \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F \mu_1 + (\|\boldsymbol{K}_2\|_F + \lambda) \mu_2 &= \tfrac{\langle \boldsymbol{K}_2, \boldsymbol{K}^* \rangle_F}{2}. \end{aligned}$$

Hence, the solution of the unconstrained problem is

$$\mu_1^* = \frac{\langle \boldsymbol{K}_1, \boldsymbol{K}^* \rangle_F - 2 \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F \, \mu_2^*}{2 \, [\|\boldsymbol{K}_1\|_F^2 + \lambda]}$$

with

$$\mu_2^* = \frac{(\|\boldsymbol{K}_1\|_F^2 + \lambda) \langle \boldsymbol{K}_2, \boldsymbol{K}^* \rangle_F - \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F \langle \boldsymbol{K}_1, \boldsymbol{K}^* \rangle_F}{2 \, [(\|\boldsymbol{K}_1\|_F^2 + \lambda)(\|\boldsymbol{K}_2\|_F^2 + \lambda) - \langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F^2]}.$$

If $\mu_1^*, \, \mu_2^* > 0$, the latter is the optimal solution of the constrained problem since it lies in the feasible region. Consider the case $\mu_1^*, \mu_2^* \leq 0$. Because $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ are positive (semi)-definite matrices, the alignment of $\mu_1^* \boldsymbol{K}_1 + \mu_2^* \boldsymbol{K}_2$ is negative. This result cannot be optimal since $(\mu_1, \mu_2) = (1, 0)$ leads to a positive alignment. We then conclude that this case cannot arise. Finally, suppose that $\mu_1^* > 0$ and $\mu_2^* < 0$. Starting from a feasible point, the feasible direction method [8] provides the constrained solution $\mu_1^+ > 0$ and $\mu_2^+ = 0$. Note that the alignment criterion is invariant under scale changes, that is, $\mathcal{A}(\mu_1^+ \boldsymbol{K}_1, \cdot) = \mathcal{A}(\boldsymbol{K}_1, \cdot)$. This implies that μ_1^+ can be arbitrarily fixed to 1. As a conclusion, the general solution to problem (6) in the $m = 2$ case is

$$(\mu_1^+, \mu_2^+) = \begin{cases} (\mu_1^*, \mu_2^*) & \text{if } \mu_1^*, \mu_2^* > 0 \\ (1, 0) & \text{if } \mu_2^* \leq 0 \\ (0, 1) & \text{if } \mu_1^* \leq 0. \end{cases} \quad (7)$$

Let $J(t) = \max_{\boldsymbol{\mu} \geq \boldsymbol{0}} \mathcal{A}\big(\sum_{i \in I_t} \mu_i \boldsymbol{K}_i, \boldsymbol{K}^*\big)$ with I_t a set of indexes. It can be shown that

$$I_1 \subset I_2 \subset \ldots \Rightarrow J(1) \leq J(2) \ldots$$

This property means that the alignment of the linear combination of a subset of kernels should not be better than any larger set containing the subset. This suggests the use of a greedy strategy to combine more than two kernels. It starts from the best available kernel in the sense of (4). Next it determines the coefficients $(\mu_{1j^*}^+, \mu_{2j^*}^+)$ that maximize the kernel-target alignment of

$$\hat{\boldsymbol{K}}(j) = \mu_{1j}^+ \left(\sum_{i \in I} \mu_i \boldsymbol{K}_i \right) + \mu_{2j}^+ \, \boldsymbol{K}_j, \; j \notin I, \quad (8)$$

where $I \subset \{1, \ldots, m\}$ contains the indexes of the kernels selected previously. This process is repeated until the alignment cannot be improved by more than $\epsilon \geq 0$, see Table 1. This strategy is obviously suboptimal. However, the update rule (8) is not subject to ill-conditioned Hessian since the solutions are calculated analytically. As can be seen in Section 5, it leads to solutions as good as, and sometimes better than those obtained by solving (6) with standard QP methods.

Choose $j^* = \arg\max_j \mathcal{A}(\boldsymbol{K}_j, \boldsymbol{K}^*)$

Set $\hat{\boldsymbol{K}} = \boldsymbol{0}$, $\boldsymbol{\mu} = (0\ 0\ ...\ 0)^t$, $\mu^+_{1j^*} = 0$ and $\mu^+_{2j^*} = 1$.

Do

Add j^* to the set of indexes I: $I = I \cup \{j^*\}$

Update $\hat{\boldsymbol{K}}$ and $\boldsymbol{\mu}$ as follows:

$$\hat{\boldsymbol{K}} = \mu^+_{1j^*}\hat{\boldsymbol{K}} + \mu^+_{2j^*}\boldsymbol{K}_{j^*}$$

$$\boldsymbol{\mu} = \mu^+_{1j^*}\boldsymbol{\mu}, \mu_{j^*} = \mu^+_{2j^*}$$

For all $j \notin I$

Maximize $\mathcal{A}(\mu_{1j}\hat{\boldsymbol{K}} + \mu_{2j}\boldsymbol{K}_j, \boldsymbol{K}^*)$ using (7)

Choose $j^* = \arg\max_{j\notin I} \mathcal{A}(\mu^+_{1j}\hat{\boldsymbol{K}} + \mu^+_{2j}\boldsymbol{K}_j, \boldsymbol{K}^*)$

While $\mathcal{A}(\mu^+_{1j^*}\hat{\boldsymbol{K}} + \mu^+_{2j}\boldsymbol{K}_j, \boldsymbol{K}^*) - \mathcal{A}(\hat{\boldsymbol{K}}, \boldsymbol{K}^*) > \epsilon$

Return $\hat{\boldsymbol{K}}$ and/or $\boldsymbol{\mu}$

Table 1. The greedy algorithm

4. OPTIMIZING THE KERNEL PARAMETERS BY A GRADIENT STEP

Previously, we have supposed that the parameters of the kernels were available from previous calculations. Here, we relax this assumption by iteratively adjusting them during the calculation of the solution (8). Let us rewrite the model (5) as

$$\kappa_{\boldsymbol{\mu}}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\Theta}) = \sum_{i=1}^{m} \mu_i\, \kappa_i(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}_i) \tag{9}$$

with $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m)$, and $\boldsymbol{\theta}_i$ the parameters of κ_i. Ideally, the model parameters should be obtained by maximizing the kernel-target alignment

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} \mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*) = \arg\max_{\boldsymbol{\Theta}} \frac{\langle \boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^* \rangle_F}{\|\boldsymbol{K}_{\boldsymbol{\Theta}}\|_F},$$

where $\boldsymbol{K}_{\boldsymbol{\Theta}}$ is the Gram matrix of the kernel (9). Let us restrict ourselves to the case where this kernel can be differentiated with respect to $\boldsymbol{\Theta}$. We have

$$\begin{aligned} \frac{\partial \langle \boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^* \rangle_F}{\partial \theta_k} &= \sum_{i,j} y_i\, y_j \frac{\partial \kappa_{\boldsymbol{\mu}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\Theta})}{\partial \theta_k} \\ &\triangleq \langle \partial_k \boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^* \rangle_F \end{aligned}$$

and

$$\begin{aligned} \frac{\partial \|\boldsymbol{K}_{\boldsymbol{\Theta}}\|_F}{\partial \theta_k} &= \left[\sum_{i,j} \frac{\partial \kappa_{\boldsymbol{\mu}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\Theta})}{\partial \theta_k}\, \kappa_{\boldsymbol{\mu}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\Theta})\right] \\ &\qquad \left[\sum_{i,j} \kappa_{\boldsymbol{\mu}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\Theta})^2\right]^{-\frac{1}{2}} \qquad (10) \\ &= \langle \partial_k \boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}_{\boldsymbol{\Theta}} \rangle_F / \|\boldsymbol{K}_{\boldsymbol{\Theta}}\|_F. \qquad (11) \end{aligned}$$

We can then express the derivative of the alignment with respect to θ_k as follows

$$\begin{aligned} \frac{\partial \mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*)}{\partial \theta_k} &= \frac{\langle \partial_k \boldsymbol{K}_{\Theta}, \boldsymbol{K}^* \rangle_F}{\|\boldsymbol{K}^*\|_F \|\boldsymbol{K}_{\boldsymbol{\Theta}}\|_F} \\ &\quad - \frac{\langle \boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^* \rangle_F \langle \boldsymbol{K}_{\boldsymbol{\Theta}}, \partial_k \boldsymbol{K}_{\Theta} \rangle_F}{\|\boldsymbol{K}^*\|_F \|\boldsymbol{K}_{\boldsymbol{\Theta}}\|_F^3}. \end{aligned} \tag{12}$$

In the spirit of [9], we propose an algorithm that alternates the optimisation of a linear combination of kernels, see Table 1, with a gradient step in the direction of the gradient of $\mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*)$ in the parameter $\boldsymbol{\Theta}$ space, see (12). This can be achieved by the following iterative procedure:

1. Initialize $\boldsymbol{\Theta}$ to some value;
2. Using our greedy algorithm, find $\boldsymbol{\mu}^*$ for $\boldsymbol{\Theta}$ fixed;
3. For all $\mu_i > 0$, update $\boldsymbol{\theta}_i$ such that $\mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*)$ is maximized. This can be achieved by a gradient ascent

   $$\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \eta \nabla_{\boldsymbol{\theta}_i} \mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*)$$

   until a given stopping criterion is met;
4. Go to step 2. or stop when the maximum of $\mathcal{A}(\boldsymbol{K}_{\boldsymbol{\Theta}}, \boldsymbol{K}^*)$ is reached.

5. EXPERIMENTS

To compare our greedy algorithm with a standard QP strategy, experiments were conducted on a Pentium 4, 3.4 GHz, 2GB RAM with the benchmark[1] *breast cancer Wisconsin*. It was randomly divided into a training set of 466 instances and a test set of 233 instances. Six kernels ($m = 6$) were selected from the family of polynomial kernels: $(1 + \gamma\langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^q$ with $q = 1, \ldots, 4$, gaussian kernels: $\exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/2\sigma^2\right)$ and exponential kernels: $\exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|/2\sigma^2\right)$. The free parameters γ and σ were chosen in the range $]0; 10]$ in such a manner that the composite kernel (5) obtained by the QP approach was composed of 3 different candidates. The threshold ϵ controlling the sparsity of the solution, see Table 1, was set to 10^{-3}. This resulted in solution with 2 nonzero μ_i's for our algorithm. In both cases, the gaussian kernel was combined with the 3rd degree polynomial kernel. The 4rd degree polynomial kernel was also involved in the QP solution, with a very small weight – about 10^{-14}. On Table 2, one can notice that the alignment obtained with our algorithm led to a substantial improvement compared to QP, which partly failed to converge due to ill-conditioned Hessian. In both cases, the alignment of the composite kernels remained high on the validation set. Figure 1 compares the computation time as a function of the number m of candidate kernels. We observe that our algorithm is much less time-consuming than the QP strategy for large m.

[1] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/

	training	test	# of kernels
alignment (QP)	0.5667	0.5012	3
alignment (greedy)	0.2198	0.2168	1
”	0.6375	0.6039	2

Table 2. Alignment of the composite kernel $\kappa_{\boldsymbol{\mu}}$. The performance of the kernel that was first selected by the greedy algorithm is also mentioned.

	alignment	error
ℓ_1-SVM using κ_{opt}	0.2313	5.14%
ℓ_1-SVM using $\kappa_{\boldsymbol{\mu}}$ (QP)	0.5667	3.58%
ℓ_1-SVM using $\kappa_{\boldsymbol{\mu}}$ (greedy)	0.6396	3.37%
ℓ_2-SVM using κ_{opt}	0.2411	3.58%
ℓ_2-SVM using $\kappa_{\boldsymbol{\mu}}$ (QP)	0.5667	2.93%
ℓ_2-SVM using $\kappa_{\boldsymbol{\mu}}$ (greedy)	0.6396	2.79%

Table 3. Alignment and error rate of SVM with κ_{opt} and $\kappa_{\boldsymbol{\mu}}$.

Our greedy approach was next coupled with a gradient descent to jointly adjust the free parameters of the 6 candidate kernels. The gradient step η was set to 0.5, and the gradient descent was stopped when the improvement of the alignment was less than 10^{-5}. The same stopping-criterion was used in step 4 of our alternating optimization scheme. The final values of the alignment were 0.6396 and 0.6093 on the training and test sets, respectively. Figure 2 shows the evolution of the alignment during the gradient descent. As suggested by the concentration property of the criterion, one can note that improving the alignment on the training set improved the alignment on the test set. Finally, the composite kernels were used with ℓ_1-SVM and ℓ_2-SVM classifiers. They were trained with C in $\{1, 10, 100\}$, and tested using 5-fold cross validation. Each kernel of the set of candidate kernels was also tested individually to determine the best one, denoted by κ_{opt}. The performances of the kernels $\kappa_{\boldsymbol{\mu}}$ and κ_{opt} are reported in Table 3. These results show that the composite kernels outperformed the kernel κ_{opt}, and give an advantage to our gradient-based greedy algorithm over the QP approach. We observed that the computation time required to exhibit κ_{opt}, including the training and the cross-validation stages, was about 6 times longer.

6. CONCLUSION

We presented a method for automatically optimizing the alignment of a linear combination of kernels while adjusting their free parameters in a data-dependent way. This led to an improvement of the performance of SVM in a binary classification context, and a drastic reduction of the time usually spent to pick a good kernel. Direct extensions of this work include multi-class and regression applications.

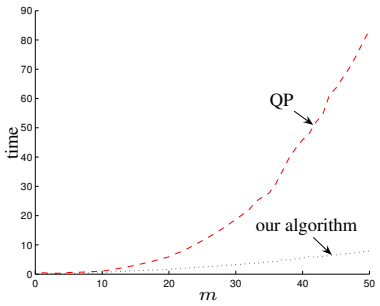

Fig. 1. Computation time, in seconds, as a function of the number m of candidate kernels.

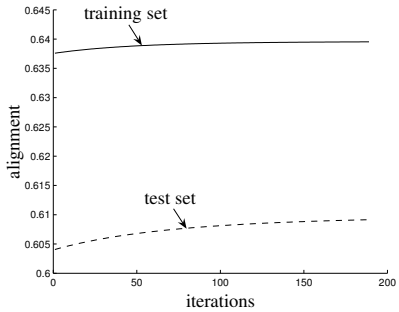

Fig. 2. Evolution of the alignment during gradient descent.

7. REFERENCES

[1] J. P. Vert, K. Tsuda, and B. Schölkopf, “A primer on kernel methods,” in *Kernel Methods in Computational Biology*, B. Schölkopf, K. Tsuda, and J. P. Vert, Eds. Cambridge, MA: The MIT Press, 2004, pp. 35–70.

[2] S. Vishwanathan, O. Guttman, K. Borgwardt, and A. Smola, “Kernel extrapolation,” National ICT Australia, Tech. Rep. 005027, 2005.

[3] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, “On kernel-target alignment,” *Advances in Neural Information Processing Systems*, vol. 14, pp. 367–373, 2002.

[4] J. Kandola, J. Shawe-Taylor, and N. Cristianini, “On the extensions of kernel alignment,” Department of Computer Science, University of London, Tech. Rep. 120, 2002.

[5] ——, “Optimizing kernel alignment over combinations of kernels,” Department of Computer Science, University of London, Tech. Rep. 121, 2002.

[6] G. R. G. Lanckriet, N. Cristianini, P. L. Bartlett, L. El Ghaoui, and M. I. Jordan, “Learning the kernel matrix with semi-definite programming,” *Proc. of the nineteenth International Conference on Machine Learning*, pp. 323–330, 2002.

[7] M. G. Genton, “Classes of kernels for machine learning: a statistics perspective,” *Journal of Machine Learning Research*, vol. 2, pp. 299–312, 2001.

[8] G. Zoutendijk, “Methods of feasible directions: a study in linear and non-linear programming,” in *Proc. of the 21st International Conference on Machines Learning*. Elsevier, 1970.

[9] O. Chapelle and V. Vapnik, “Choosing multiple parameters for support vector machines,” *Machine Learning*, vol. 46, no. 1-3, pp. 131–159, 2003.