OPTIMAL FEATURE REPRESENTATION FOR KERNEL MACHINES USING KERNEL-TARGET ALIGNMENT CRITERION

Jean-Baptiste Pothin, Cédric Richard

Institut Charles Delaunay (ICD-M2S, FRE CNRS 2848)
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 is commonly used to predict the behavior of any given reproducing kernel in a classification context, without training any kernel machine. In this paper, we present a gradient ascent algorithm for maximizing the alignment over linear transform of the input space. Our method is compared to the minimization of the radius-margin bound. Experimental results on multi-dimensional benchmarks show the effectiveness of our approach.

Index Terms— pattern classification, alignment, SVM

1. INTRODUCTION

Kernel-based methods map a set of data $\boldsymbol{x}$ from the input space $\mathcal{X}$ into some other (possibly infinite) feature space $\mathcal{F}$ via a nonlinear map ϕ, and then apply a linear procedure in $\mathcal{F}$. The embedding is performed by substituting kernel values for the inner products, i.e., $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{F}}$. This provides an elegant way of dealing with nonlinear algorithms by reducing them to linear ones in $\mathcal{F}$. A typical example is Support Vector Machines (SVMs) [1], 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.

Despite the success of kernel machines, the selection of an appropriate kernel is still critical for achieving good generalization performance. A recent development is on formulating the kernel design stage as the problem of optimizing distance metric. The basic idea is to determine a suitable linear transform $\boldsymbol{S}$ in $\mathcal{F}$ which minimizes the discrepancy between distances in the feature space and *idealized* distances based on weak label information [2]. To deal with the so-called curse of dimensionality, most of the existing techniques attempt to select $\boldsymbol{S}$ such that $\boldsymbol{S}\boldsymbol{S}^t$ remains close to an unweighted Euclidean metric under the constraints that similar points get closer and/or dissimilar points move away [3, 4, 5]. One advantage of this approach is that the solution can be expressed in terms of kind of support vectors only, as in SVMs. However, this leads to computationally expensive kernels when similarity information is abundant, making them impractical for real-time and large-scale applications.

In this paper, we consider a suitable linear transform in Euclidean input space $\mathcal{X}$ rather than in feature space $\mathcal{F}$. In [6], a similar strategy was employed to measure the relevance of input features specially for SVMs. However, the authors only consider adaptive scaling, that is, diagonal metric in input space. In [7], an extension to the full metric is proposed. In both references, the algorithms are based on alternating optimization of a standard nonlinear SVM and a gradient descent of the radius-margin bound (RMB). We suggest to replace the use of RMB criterion by kernel-target alignment (KTA). The latter measures the degree of agreement between a reproducing kernel and a learning task [8]. Previous works focused on its optimization by linear combination of kernels in a transductive or inductive settings, see [9] for a recent state of the art. The aim of this paper is a gradient step algorithm to optimize KTA over a linear transform in input space. Unlike [6, 7], our algorithm does not require the design of any classifier at each iteration.

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

2. KERNEL-TARGET ALIGNMENT

The alignment criterion is a measure of similarity between two kernels, or between a kernel and a target function [8]. Given a n-sample data set $\mathcal{D}_n$, the alignment of kernels κ_1 and κ_2 is defined as follows

$$\mathcal{A}(\boldsymbol{K}_1, \boldsymbol{K}_2) = \frac{\langle \boldsymbol{K}_1, \boldsymbol{K}_2 \rangle_F}{\sqrt{\langle \boldsymbol{K}_1, \boldsymbol{K}_1 \rangle_F \langle \boldsymbol{K}_2, \boldsymbol{K}_2 \rangle_F}}, \qquad (1)$$

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 $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D}_n$.

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

$$\boldsymbol{K}^*(i,j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle = \begin{cases} 1 & \text{if } y_i = y_j \\ -1 & \text{if } y_i \neq y_j. \end{cases} \quad (2)$$

In [8], Cristianini *et al.* propose to maximize the alignment with respect to $\boldsymbol{K}$ for the target $\boldsymbol{K}^* = \boldsymbol{y}\boldsymbol{y}^t$ in order to determine the most relevant kernel for a given classification task.

$$\mathcal{A}(\boldsymbol{K}, \boldsymbol{K}^*) = \frac{\langle \boldsymbol{K}, \boldsymbol{y}\boldsymbol{y}^t \rangle_F}{\sqrt{\langle \boldsymbol{K}, \boldsymbol{K} \rangle_F \langle \boldsymbol{y}\boldsymbol{y}^t, \boldsymbol{y}\boldsymbol{y}^t \rangle_F}} = \frac{\boldsymbol{y}^t \boldsymbol{K} \boldsymbol{y}}{n \|\boldsymbol{K}\|_F}. \quad (3)$$

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*, i.e., the probability of the empirical estimator (3) deviating from its mean can be bounded by an exponentially decaying function of this deviation [8]. This means that if one optimizes the alignment over 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. Finally, a close relationship between the well-known Fisher criterion for measuring the linear separability of classes and the kernel-target alignment was shown in [10].

3. OPTIMIZATION OF KTA BY LINEAR TRANSFORM OF THE INPUT SPACE

Let $\tilde{\boldsymbol{x}}$ be the linear transformation of input point $\boldsymbol{x} \in \mathbb{R}^p$ with the matrix $\boldsymbol{S} \in \mathbb{R}^{p \times p}$, namely:

$$\tilde{\boldsymbol{x}} = \boldsymbol{S}^t \boldsymbol{x}. \quad (4)$$

After transformation, distance in input space becomes:

$$d_{\boldsymbol{S}}(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^t \boldsymbol{S}\boldsymbol{S}^t (\boldsymbol{x} - \boldsymbol{x}')}. \quad (5)$$

The case of a diagonal matrix with positive scaling factors is also referred as *feature weighting* but, more generally, $\boldsymbol{S}$ can be any real-valued matrix. In that case, $\boldsymbol{S}\boldsymbol{S}^t$ is semi-positive definite and $d_{\boldsymbol{S}}(\boldsymbol{x}, \boldsymbol{x}')$ is a valid pseudo-metric, i.e., it satisfies non-negativity and the triangle inequality.

We propose in this section a gradient ascent algorithm to optimize the matrix transformation $\boldsymbol{S}$ so that the kernel-target alignment is maximized. In other word, we search for:

$$\boldsymbol{S}^* = \arg\max_{\boldsymbol{S}} \mathcal{A}(\tilde{\boldsymbol{K}}, \boldsymbol{K}^*), \quad (6)$$

where $\tilde{\boldsymbol{K}}$ is the Gram matrix with entries $\kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)$ for all $\boldsymbol{x}_i$, $\boldsymbol{x}_j \in \mathcal{D}_n$. Note that we attempt to optimize a criterion measured in a feature space by applying a linear transform over the input space. One can justify this approach, in particular, for the class of monotonic isotropic kernels such as the Gaussian or Laplacian kernels.

Let $\boldsymbol{\Delta} = \boldsymbol{x}_i - \boldsymbol{x}_j$, $\tilde{\boldsymbol{\Delta}} = \tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j$, $\boldsymbol{\Delta}_\phi = \phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)$ and $\tilde{\boldsymbol{\Delta}}_\phi = \phi(\tilde{\boldsymbol{x}}_i) - \phi(\tilde{\boldsymbol{x}}_j)$. For a monotonic isotropic kernel $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \psi(\|\boldsymbol{x} - \boldsymbol{x}'\|)$, we have

$$\|\tilde{\boldsymbol{\Delta}}\| < \|\boldsymbol{\Delta}\| \Rightarrow \psi(\|\tilde{\boldsymbol{\Delta}}\|) > \psi(\|\boldsymbol{\Delta}\|),$$

that is, the kernel value of close points is higher than the kernel value of distant points. Noting that

$$\begin{aligned} \|\phi(\boldsymbol{x}) - \phi(\boldsymbol{x}')\|^2 &= \kappa(\boldsymbol{x}, \boldsymbol{x}) + \kappa(\boldsymbol{x}', \boldsymbol{x}') - 2\kappa(\boldsymbol{x}, \boldsymbol{x}') \\ &= 2(\psi(0) - \psi(\|\boldsymbol{x} - \boldsymbol{x}'\|)), \end{aligned}$$

it follows that $\|\tilde{\boldsymbol{\Delta}}\| < \|\boldsymbol{\Delta}\|$ implies $\|\tilde{\boldsymbol{\Delta}}_\phi\| < \|\boldsymbol{\Delta}_\phi\|$. Thus, reducing the distance between points of the same class in input space leads to a reduction of their distance in induced feature space. This implies that, with these kernels, improving the *separability* of classes in input space improves the separability of the classes in the feature space as well.

3.1. Case of full matrix $\boldsymbol{S}$

Let us consider the case where the kernel κ can be differentiated with respect to the parameter $\boldsymbol{S}$. In this context, we have

$$\frac{\partial \langle \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F}{\partial \boldsymbol{S}} = \sum_{i,j} y_i\, y_j \frac{\partial \kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)}{\partial \boldsymbol{S}} \quad (7)$$

$$\triangleq \langle \partial \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F \quad (8)$$

and

$$\frac{\partial \|\tilde{\boldsymbol{K}}\|_F}{\partial \boldsymbol{S}} = \left[\sum_{i,j} \frac{\partial \kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)}{\partial \boldsymbol{S}} \kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) \right] \left[\sum_{i,j} \kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)^2 \right]^{-\frac{1}{2}} \quad (9)$$

$$= \langle \partial \tilde{\boldsymbol{K}}, \tilde{\boldsymbol{K}} \rangle_F / \|\tilde{\boldsymbol{K}}\|_F. \quad (10)$$

We can then express the derivative of the alignment with respect to $\boldsymbol{S}$ as follows

$$\frac{\partial \mathcal{A}(\tilde{\boldsymbol{K}}, \boldsymbol{K}^*)}{\partial \boldsymbol{S}} = \frac{\langle \partial \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F}{\|\boldsymbol{K}^*\|_F \|\tilde{\boldsymbol{K}}\|_F} - \frac{\langle \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F \langle \partial \tilde{\boldsymbol{K}}, \tilde{\boldsymbol{K}} \rangle_F}{\|\boldsymbol{K}^*\|_F \|\tilde{\boldsymbol{K}}\|_F^3}. \quad (11)$$

To pursue calculations, we restrict ourselves to the Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 / 2\sigma^2)$. Without loss of generality, we change notation as follows: $\boldsymbol{S} \leftarrow \boldsymbol{S}/2\sigma^2$. This leads to:

$$\kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) = e^{-(\boldsymbol{x}_i^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_i - 2\boldsymbol{x}_i^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_j + \boldsymbol{x}_j^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_j)}. \quad (12)$$

The derivative can be written as:

$$\frac{\partial \kappa(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j)}{\partial \boldsymbol{S}} = \frac{\partial \kappa(\boldsymbol{S}^t \boldsymbol{x}_i, \boldsymbol{S}^t \boldsymbol{x}_j)}{\partial \boldsymbol{S}} = f_{ij} \frac{\partial g_{ij}}{\partial \boldsymbol{S}}, \quad (13)$$

where f_{ij} and g_{ij} are defined as:

$$\begin{aligned} f_{ij} &= \kappa(\boldsymbol{S}^t \boldsymbol{x}_i, \boldsymbol{S}^t \boldsymbol{x}_j) \\ g_{ij} &= -(\boldsymbol{x}_i^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_i - 2\boldsymbol{x}_i^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_j + \boldsymbol{x}_j^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_j). \end{aligned} \quad (14)$$

The expression (13) is the derivative of a scalar function with respect to a matrix. As such, the derivative is a matrix where each element is the derivative with respect to each element of matrix $\boldsymbol{S}$. Since $\partial \boldsymbol{x}_i^t \boldsymbol{S}\boldsymbol{S}^t \boldsymbol{x}_j / \partial \boldsymbol{S} = (\boldsymbol{x}_i \boldsymbol{x}_j^t + \boldsymbol{x}_j \boldsymbol{x}_i^t)\boldsymbol{S}$, we have

$$\frac{\partial g_{ij}}{\partial \boldsymbol{S}} = -2(\boldsymbol{x}_i \boldsymbol{x}_i^t - \boldsymbol{x}_i \boldsymbol{x}_j^t - \boldsymbol{x}_j \boldsymbol{x}_i^t + \boldsymbol{x}_j \boldsymbol{x}_j^t)\boldsymbol{S}. \quad (15)$$

Thus:

$$\langle \partial \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F = \sum_{i,j} y_i y_j f_{ij} \boldsymbol{A}_{ij} \boldsymbol{S}, \quad (16)$$

where the matrix $\boldsymbol{A}_{ij} = -2(\boldsymbol{x}_i \boldsymbol{x}_i^t - \boldsymbol{x}_i \boldsymbol{x}_j^t - \boldsymbol{x}_j \boldsymbol{x}_i^t + \boldsymbol{x}_j \boldsymbol{x}_j^t)$. In the same way, we have:

$$\langle \partial \tilde{\boldsymbol{K}}, \boldsymbol{K} \rangle_F = \sum_{i,j} f_{ij}^2 \boldsymbol{A}_{ij} \boldsymbol{S}. \quad (17)$$

Reinjecting (16) and (17) into (11) leads to the following gradient update rule for $\boldsymbol{S}$:

$$\boldsymbol{S} \leftarrow \boldsymbol{S} + \eta \left(\sum_{i,j} [\alpha y_i y_j f_{ij} - \beta f_{ij}^2] \boldsymbol{A}_{ij} \right) \boldsymbol{S}, \quad (18)$$

where $\alpha = 1/\|\tilde{\boldsymbol{K}}\|_F$, $\beta = \langle \tilde{\boldsymbol{K}}, \boldsymbol{K}^* \rangle_F / \|\tilde{\boldsymbol{K}}\|_F^3$ and $\eta > 0$ is the gradient step. Finally, we compute $\boldsymbol{S}$ using the following iterative procedure:

1. Initialize randomly the matrix $\boldsymbol{S}$;
2. Update $\boldsymbol{S}$ with the rule in equation (18);
3. Return to step 2 or terminate if the alignment between $\tilde{\boldsymbol{K}}$ and $\boldsymbol{K}^*$ remains the same at the precision ϵ

3.2. Case of diagonal matrix $\boldsymbol{S}$

An efficient algorithm can be derived in the case where $\boldsymbol{S}$ is diagonal. The idea is to perform the gradient step onto the vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)^t$. From (15), one can show that:

$$\frac{\partial g_{ij}}{\partial \theta_k} = \frac{\partial g_{ij}}{\partial s_{kk}} = -2\left(x_{ik}^2 + x_{jk}^2 - 2x_{ik}x_{jk} \right)\theta_k, \quad (19)$$

where x_{lk} is the k^{th} attribute of sample $\boldsymbol{x}_l$. Using the element-wise product operator $*$, the gradient with respect to $\boldsymbol{\theta}$ can be written as:

$$\frac{\partial g}{\partial \boldsymbol{\theta}} = -2\left(\boldsymbol{x}_i * \boldsymbol{x}_i + \boldsymbol{x}_j * \boldsymbol{x}_j - 2\boldsymbol{x}_i * \boldsymbol{x}_j \right) * \boldsymbol{\theta}. \quad (20)$$

Data set	# variables	# training data	# test data
Ringnorm	20	400	7000
Thyroid	5	75	1500
Titanic	3	150	2051
Waveform	21	400	4600

Table 1. General information about the data sets. Other details can be found at http://www.ics.uci.edu/~mlearn/

Denoting by $\boldsymbol{a}_{ij}$ the column vector in parentheses, a similar calculation to (18) shows that the gradient update rule for $\boldsymbol{\theta}$ can be written as:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \left(\sum_{i,j} [\alpha y_i y_j f_{ij} - \beta f_{ij}^2] \boldsymbol{a}_{ij} \right) * \boldsymbol{\theta}. \quad (21)$$

Note here that we do not require $\theta_k \geq 0$ because the terms in the diagonal of the metric $\boldsymbol{S}\boldsymbol{S}^t$ are θ_k^2.

4. EXPERIMENTS

Experiments were conducted on a Pentium 4, 3.4Ghz, 2GB RAM. Four data sets were used to test the algorithm. General information about them is given in Table 1. For the above experiments, the linear transform $\boldsymbol{S}$ was initialized to $(1/2\sigma^2)\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. Our algorithm stopped when the improvement of the alignment was less than 10^{-6}.

Figure 1 shows the evolution of the alignment for the Ringnorm data set and different values of standard deviation σ. We note that this parameter has influence on the convergence rate only, which was found, on this particular data set, to remain roughly the same for $\sigma \geq 3$. Transformed data were used as inputs of a l_2-SVM. We fixed the regularization parameter C to 100 and σ to 1. Figure 2 shows the evolution of the alignment/test error across iterations for Ringnorm data set. Note the close connection between alignment maximization and generalization error minimization. For each data set, we compared our algorithm with an implementation of Fortuna *et al.*'s algorithm based on the minimization of the RMB [7]. To speed up the successive trainings of SVMs in this algorithm, we followed the standard practice [11] consisting of initializing them with the solutions of the preceding round. Table 2 describes the performance with and without the linear transform $\boldsymbol{S}$. We see that our linear transform increased the alignment as expected. It also reduced the number of support vectors and improved the generalization performance, except for Titanic where they remained unchanged. However, Fortuna *et al.*'s algorithm gave smaller test errors. This result confirms that RMB provides a good approximation of the leave-one-out error for l_2-SVMs, as shown in [12]. However, note that solutions involved more support vectors, in particular for Waveform data set. Finally, Table 3 gives the computation time for both algorithms, average over 100 iterations. Mainly due to the training of SVMs at each iteration,

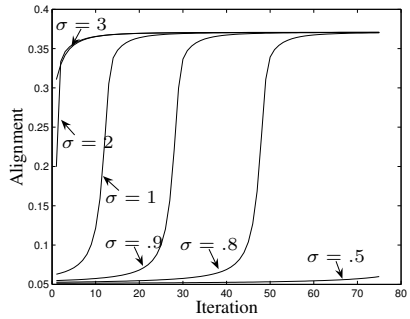

Fig. 1. Evolution of the alignment for different values of standard deviation σ and Ringnorm data set.

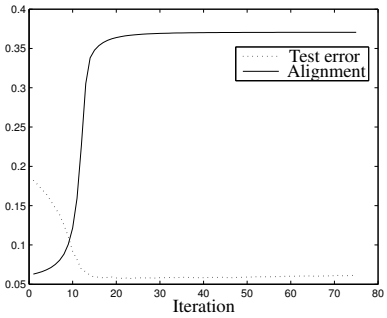

Fig. 2. Ringnorm data set iterations for $\sigma = 1$.

the RMB-based update rule was found to be up to 10 times slower than our KTA-based update rule.

5. CONCLUSION

We proposed a simple update rule for maximizing kernel-target alignment over linear transform in input space. Experimental results on multi-dimensional benchmarks showed its effectiveness. Compared to RMB-based update rule, our algorithm gave slightly worse performance. However, solutions were sparser and training was much less time consuming. Extension of this work includes multi-class and regression applications. It may also be interested to compare performance of our approach with distance metric learning in feature space [5].

6. REFERENCES

[1] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY: Springer, 1995.

[2] Z. Zhang, "Learning metrics via discriminant kernels and multidimensional scaling: toward expected euclidean representation," in *Proc. of the 20th International Conference on Machine Learning*, Washington DC, 2003.

[3] J. T. Kwok and I. W. Tsang, "Learning with idealized kernels," *Proc. of the Twentieth International Conference on Machine Learning*, pp. 400–407, 2003.

		without	*with*
Ringnorm	Alignment	.0629	.3704 (.3247)
	NSV	398	130 (141)
	Error	18.20%	5.89% (4.87%)
Thyroid	Alignment	.4757	.5805 (.5464)
	NSV	31	19 (37)
	Error	2.27%	2% (2.27%)
Titanic	Alignment	.1918	.3434 (.2293%)
	NSV	147	147 (147)
	Error	25.89%	25.89% (25.89%)
Waveform	Alignment	0.0519	.3937 (.2460)
	NSV	400	95 (237)
	Error	32.80%	15.52% (15.22%)

Table 2. Alignment, number of support vectors (NSV) and test error *with* and *without* linear transform $\boldsymbol{S}$. The values in parentheses are for Fortuna *et al.*'s algorithm.

Data set	Fortuna *et al.*'s algorithm	our algorithm
Ringnorm	28	3
Thyroid	0.35	0.05
Titanic	0.25	0.19
Waveform	26	3

Table 3. Computation time per iteration (in seconds).

[4] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in *Proc. of the Conference on Advance in Neural Information Processing Systems*, 2003.

[5] G. Wu, N. Panda, and E. Y. Change, "Formulating distance functions via the kernel trick," in *ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2005, pp. 703–709.

[6] Y. Grandvalet and S. Canu, "Adaptive scaling for feature selection in SVMs." Cambridge, MA: The MIT Press, 2003.

[7] J. Fortuna and D. Capson, "An optimal basis for feature extraction with support vector machine classification using the radius-margin bound," *IEEE International Conference on Acoustic, Speech and Signal Processing*, vol. 5, pp. 565–568, 2006.

[8] 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.

[9] J.-B. Pothin and C. Richard, "A greedy algorithm for optimizing the kernel alignment and the performance of kernel machines." *EUSIPCO'06*, 4-8 September 2006.

[10] H. Xiong and M. O. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Transactions on Neural Networks*, vol. 16, no. 2, pp. 460–474, March 2005.

[11] N. Cristianini and C. Campbell, "Dynamically adapting kernels in support vector machines," *Proc. of Neural Information Processing Workshop*, vol. 11, pp. 204–210, 1999.

[12] K. Duan, S. Keerthi, and A. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," *Neurocomputing*, vol. 51, pp. 41–59, 2002.