From Samples to Objects in Kernel Methods

Alexei Pozdnoukhov [a] Samy Bengio [a]

IDIAP–RR 03-29

June 2003

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 − 27 − 721 77 11
fax +41 − 27 − 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

[a] IDIAP, P.O. Box 592, CH-1920 Martigny, Switzerland

IDIAP Research Report 03-29

From Samples to Objects in Kernel Methods

Alexei Pozdnoukhov Samy Bengio

June 2003

Abstract. This paper presents a general method for incorporating prior knowledge into kernel methods. It applies when the prior knowledge can be formalized by the description of an object around each sample of the training set, assuming that all points in the given object share the same desired class. Two implementation techniques of this method, based on analytical kernel jittering and the vicinal risk minimization principle, are considered. Empirical results on one artificial dataset and one real dataset based on EEG signals demonstrate the performance of the proposed method.

1 Introduction

Prior knowledge is often used in machine learning algorithms to constrain models towards reasonable solutions. One such class of prior knowledge relates to invariances. These are transformations of the inputs that leave the outputs unchanged. The general setting of including invariances into kernel methods was considered in [1]. One of the widely used practical methods for incorporating invariances into SVMs is the Virtual Support Vector method, based on generating artificial samples from the current Support Vectors of the problem [2]. The method performed particularly well on an optical digit recognition task. Another general way is to modify the cost function of the algorithm in order to penalize solutions not following the invariance properties. One such method (though not really suitable for large-scale datasets) was developed in [3]. Finally, a method called "kernel jittering" [4] combines the generation of artificial examples with kernel modification.

In this paper, we present yet another approach to the problem, which assumes that the prior knowledge can be formalized as a mapping of a special kind. This mapping transforms each sample into an object in such a way that it includes the prior knowledge, similar to that done in the Tangent Distance approach [5]. The method does not lead either to enlarged training sets or to modification of the cost function. It simply exploits standard SVM optimization algorithms.

The rest of the paper is organized as follows. The general idea is presented in Section 2, as well as two implementations based on "analytical jittering" and the Vicinal Risk Minimization principle. Section 3 presents some experiments on artificial data where we illustrate the performance of the proposed method, and on real data, where the task is to classify EEG signals for a Brain-Computer Interface system. Section 4 completes the paper with discussion and conclusion.

2 From Samples To Objects

Suppose we have some understanding of our data that can be formalized as a transformation of the inputs that leaves the outputs unchanged. For example, in a 2D image classification task we are often given the evident knowledge that small rotations and translations of the raw images do not affect the desired output class. Suppose the representation of the data (the set of features) allows us to describe the desired transformation as a mapping that leaves the outputs unchanged. The mapping applied to every sample produces a set of corresponding objects, which becomes a point of our consideration. In other words, we assume that given some understanding of the data we are able to generalize each sample into the equivalence class - the object in the input space.

In general, this can also be thought of as follows. Suppose we have an (output-independent) *model* for the samples we are dealing with. The model represents our knowledge of the sample's properties and can be of a very general kind. Given a sample from the dataset, the model transforms it into an *object*:

$$object_i = model|sample_i \tag{1}$$

If this formalization is successfully performed, it is possible to deal with *objects* instead of samples when solving our particular learning problem. The approach is quite general since it gives a way to include some prior knowledge (on the invariances, for instance) into the learning method.

In the following subsections, we describe some particular examples of implementation of the described approach.

2.1 Analytical "Jittering"

The method of kernel jittering was proposed in [4]. It combines artificial sample generation and kernel function modification as follows. Consider two samples, x_i and x_j and the corresponding non-jittered kernel function K_{ij}. Assume sample x_j could have been any of a set of values around x_j according to a "jittering" function. Consider the transformed ("jittered") forms of the sample x_j, including itself, and select one (x_{q^*}) closest to x_i in the feature space according to the Euclidean distance in the feature space:

$$q^* = \arg\min_q \sqrt{K_{ii} - 2K_{iq} + K_{qq}}. \qquad (2)$$

The new "jittered" kernel for the examples x_i and x_j is simply K_{iq^*}. This idea can be interpreted as follows. Believing that transformed examples belong to the same class, kernel jittering corresponds to a kernel based on the distance between the sets generated from the examples by the allowed transformations.

The main drawback of the jittering approach is the need to do a lot of kernel calculations while selecting the minimal distance (2). The approach also requires that we do these calculations during the testing phase. Moreover, the kernel matrix is no longer symmetric and can, in fact, become non positive definite.

We argue in this paper that it is sometimes possible to implicitly define the desired transformation, hence carrying out the "jittering" and distance minimization analytically.

Suppose we use the isotropic Gaussian RBF kernel as the "base" one:

$$K(x_i, x_j) = e^{\frac{-\rho^2}{2\sigma^2}}, \qquad (3)$$

where the distance in the input space is defined by $\rho^2 = \|x_i - x_j\|^2$ and σ^2 is the variance of the kernel. Then, distance minimization in the feature space (2) is equivalent to distance minimization in the input space. The jittered kernel is then the RBF kernel with the minimized distance.

Let us now consider the analytical jittering process. We restrict ourselves to jitterings that can be described analytically. This assumption restricts the set of possible invariances, since a lot of real-life invariant transformations can not, unfortunately, be formalized.

Having proposed to define a *geometrical object* in the input space that corresponds to the set generated by jittering the sample, we can now consider distances between the samples and objects or between two geometrical objects. In the following subsections we propose some examples of such analytical jittering.

2.1.1 Linear Transformation

Consider the linear transformation

$$x \mapsto \alpha x + \beta e, \qquad (4)$$

where $\alpha \in R^1$ or $\alpha \in [a, b]$, $a, b \in R^1$, $\beta \in [c, d]$, $c, d \in R^1$, and e is a unit vector.

In this case and considering $\beta = 0$, the desired distance is given by

$$\rho^2(x, L(x_i)) = (x - \alpha^* x_i)^2, \;\; \alpha^* = \frac{(x, x_i)}{x_i^2}, \qquad (5)$$

which is simply the distance between x and the line $L(x_i)$ corresponding to the directing vector x_i. If the possible scalings of x_i are bounded to the segment $[ax_i, bx_i]$, the distance is given by

$$\rho^2(x, L(x_i)) = (x - |\alpha^*|_{[a,b]} x_i)^2, \;\; \alpha^* = \frac{(x, x_i)}{x_i^2} \qquad (6)$$

where $|g|_{[a,b]}$ is defined as a if $g < a$, b if $g > b$, and g otherwise.

When $\beta \neq 0$ it becomes slightly more complicated. The distance between x and a two-dimensional manifold given by

$$x \mapsto G_x = \{xt + bp; t \in [a, b], p \in [c, d]\} \qquad (7)$$

has to be computed. In 3D it is a flat parallelogram with the vertexes formed by vectors $ax_i - ce$, $ax_i + ce$, $bx_i - ce$, $bx_i + ce$. This problem can be solved analytically.

2.1.2 Translations

The second example is a particular case of translation invariance, i.e. the desired transformation is

$$x \mapsto P_x = \{\vec{e}_i t_i + x; t_i \in [-t_i^{lim}, t_i^{lim}]\}, \tag{8}$$

where $\vec{e}_i$ are the basis vectors of the input space R^N and t_i^{lim} is the maximum allowed translation in dimension i, with $i = 1, 2, ...N$. It corresponds to the mapping into a parallelepiped whose "center" is vector x and all the edges are parallel to the axes.

The distance between a vector x and a parallelepiped $P_{x'}$ is given by minimization of

$$\rho^2(x, P_{x'}) = \min_{\vec{t}} (x - P_{x'}(\vec{t}))^2, \tag{9}$$

over the set of parameters $\vec{t} = \{t_1, t_2, ...t_N\}$ and can be calculated as follows:

$$\rho^2(x, P_{x'}) = \sum_{j=1}^{N} ((x^j - x'^j) - |x^j - x'^j|_{[-t_i^{lim}, t_i^{lim}]})^2, \tag{10}$$

where x^j, x'^j are the components of the corresponding vectors.

2.1.3 Object to Object Distances

Using the sample-to-object distances, we take into account some prior knowledge but still use a non symmetric kernel matrix. For the considered examples, the segment-to-segment and box-to-box distances are derived analytically. However, in general, the object-to-object distance calculation is quite a difficult task and often can not be easily performed. In general, the computation of the Euclidean distance between the objects leads to a constrained optimization problem. However, if the unbounded parameterization is used, one can apply the gradient methods for distance calculation while taking care of the local minima if the object is not convex. If the object is a differentiable manifold, the approximation with the tangent plane can also be used [5].

In some cases it is possible to provide a part of the procedure analytically, but the rest of the procedure can still be computationally expensive for high-dimensional input spaces. The evident way of applying a Monte-Carlo method in these cases can appear to be equivalent to the original jittering.

2.2 Vicinal Risk Minimization

Consider the standard setting of the learning problem. Given the problem of learning the function $f(x, \lambda)$ from the examples $((x_1, y_1), (x_2, y_2), ...(x_\ell, y_\ell))$, one has to take an appropriate loss function $L(.)$ and minimize the following risk functional

$$R(\lambda) = \int L(y - f(x, \lambda)) \hat{p}(x, y) dx dy \tag{11}$$

using some estimation of the probability density $\hat{p}(x, y)$. The well-known Empirical Risk Minimization principle [6] can be easily derived from (11), assuming the trivial empirical distribution.

Making more sophisticated assumptions on the density $\hat{p}(x, y)$, it is possible to derive different learning algorithms. It was shown that this framework integrates several existing algorithms, including Ridge Regression, Constrained Logistic Classifiers and Support Vector Machines [7].

Vicinal Risk Minimization (VRM) is a new learning principle proposed by Vapnik [8]. Defining the *vicinities* of the training samples (the support of the distribution functions) and estimating the unknown input density therein, we obtain the following Vicinal Risk functional:

$$R_{vic}(\alpha) = \frac{1}{\ell} \sum_{i=1}^{\ell} L\left(y - \int f(x, \alpha) p(x | x_i, r_i) dx\right), \tag{12}$$

where x_i is a training sample and r_i is its vicinity parameter. Minimizing (12) instead of (11) is called the Vicinal Risk Minimization (VRM) principle.

Vapnik mentions how to use the VRM principle to incorporate an invariance into the learning algorithm. Using the density functions $p(x|x_i, r_i)$ defined on the non-symmetrical support that describes the invariance to the desired transformation, one enforces the learning algorithm to obey the invariance's properties.

The idea can be related to the one described in section 2. As we will see, in the VRM-based SVM algorithm the vicinity definition appears to be equivalent to specifying the corresponding objects and kernels between them.

The following Vicinal Support Vector algorithm is obtained in [8]:

$$f(x) = \sum_{i=1}^{\ell} \beta_i^* D(x, x_i) + b, \tag{13}$$

where the β_i^* coefficients are such that

$$\beta^* = \arg\max_{\beta} \sum_{i=1}^{\ell} \beta_i - \frac{1}{2} \sum_{i,j=1}^{\ell} y_i y_j \beta_i \beta_j M(x_i, x_j), \tag{14}$$

subject to the constraints:

$$\begin{array}{l} \sum\limits_{i=1}^{\ell} y_i \beta_i = 0, \\ 0 \le \beta_i \le C. \end{array} \tag{15}$$

Functions $D(x, x_i)$ and $M(x_i, x_j)$ are one- and two-vicinal kernels correspondingly:

$$D(x, x_i) = \int K(x, x') p(x'|x_i, r_i) dx', \tag{16}$$

$$M(x_i, x_j) = \iint K(x, x') p(x|x_i, r_i) p(x'|x_j, r_j) dx dx'. \tag{17}$$

2.2.1 Scaling Invariance

Let us present here a simple example. To obtain the invariance described by (4) with $\beta = 0$, consider the following vicinity density function:

$$p(x|x_i, \gamma) = \frac{1}{\sqrt{2\pi}\gamma} \int \delta(x - (1-\alpha)x_i) e^{-\frac{\alpha^2}{2\gamma^2}} d\alpha, \tag{18}$$

where δ is the delta function, and the γ parameter defines the width of the vicinity and, hence, the influence of scaling invariance.

Substituting (18) in both (16) and (17) using the standard isotropic RBF kernel function given in (3) with the bandwidth parameter σ gives:

$$D(x, x_i) = \frac{\sigma}{\kappa} e^{-\frac{(x-x_i)^2}{2\kappa^2}} e^{-\frac{\gamma^2}{2\sigma^2\kappa^2}(x^2 x_i^2 - (x, x_i)^2)} \tag{19}$$

and

$$M(x_i, x_j) = \frac{\sigma^2}{\eta} e^{-\frac{\sigma^2 (x_i - x_j)^2}{2\eta^2}} e^{-\frac{\gamma^2}{\eta^2}(x_i^2 x_j^2 - (x_i, x_j)^2)}, \tag{20}$$

where the following definitions were used:

$$\kappa^2 = \gamma^2 x_i^2 + \sigma^2, \tag{21}$$

$$\eta^2 = \gamma^2\sigma^2(x_i^2 + x_j^2) + \gamma^4(x_i^2 x_j^2 - (x_i, x_j)^2) + \sigma^4. \tag{22}$$

The resulting kernels are still RBF-based. One can note that the "effective" kernel bandwidth depends both on the σ and γ parameters and on the samples x_i and x_j. This is an important property of the algorithm: *the kernel function varies in the input space.*

The one-vicinal VRM-based kernel for the scalings bounded to $[a, b]$ ("hard" vicinity with constant density is used) looks like

$$D(x, x_i) = \sqrt{\frac{\pi}{2}} \frac{\sigma}{\|x_i\|(b-a)} e^{-\frac{x^2 x_i^2 - (x, x_i)^2}{2\sigma^2 x_i^2}} \left[erf\left(\frac{t x_i^2 - (x, x_i)}{\sqrt{2}\sigma\|x_i\|}\right)\Bigg|_{t=a}^{t=b} \right]. \tag{23}$$

In the limit case of unbounded scalings, this kernel mainly coincides with the RBF kernel (3) with distance (6), obtained by "analytical jittering".

In general, the kernels (19) and (20) correspond to the kernels between objects defined by (18). The difference between the approaches described in sections 2.1 and 2.2 lies in several aspects. First of all, VRM allows using "soft" objects, while in the analytical jittering approach we considered only "hard" geometrical objects. Another difference is that in VRM we use the kernel averaged over the object, while in analytical jittering the kernel is modified by taking the distance between objects: the minimal distance between samples the object consists of. The VRM approach also provides a way to deal with any kernel and not only with distance-based kernels.

3 Applications and Experiments

Real-life problems can rarely be formulated in a way suitable for object definition in the input space. For example, it is hardly possible to define objects that correspond to the invariances of interest in image processing such as 3D rotations with changing lighting conditions. This is the main drawback of the described approaches. However, in some cases preprocessing and feature selection can be applied to obtain the suitable data representation.

We present in this section two series of experiments illustrating the proposed approaches. The first one is an artificial two-class classification task, while the second one is a problem of EEG signals classification - a real task from the field of Brain-Computer Interface system design.

3.1 Artificial Data

To illustrate the action of the considered methods, we used an artificial dataset generated to be invariant to (4) with $\beta = 0$. The goal is thus to illustrate the influence of the modified kernels on the decision boundary.

Figure 1 illustrates the training data of both classes and the decision boundaries obtained with the following algorithms: the left image shows the original SVM with RBF kernel ($\sigma = 0.2$); the center one shows an SVM with RBF kernel ($\sigma = 0.2$) and distance defined by (6) with $a = 0.5$, $b = 2$; finally, on the right we see an SVM with RBF kernel ($\sigma = 0.2$) and distance defined by (6) with $a = 0.01$, $b = 10$. The substantial difference between the presented solutions lies in the number of SV's, which is 20 for the standard solution (left figure) and 8 for the modified one (right figure). Applying the VRM-based algorithm results in similar solutions.

3.2 EEG Signals

The next series of experiments used EEG signals taken from the competition devoted to Brain-Computer Interface system design. The details of data collecting and problem settings can be found at [http://crick.bme.columbia.edu/competition.htm]. The task is to classify the signals that correspond to imaginary movements of the left or right hand. The original data consists of signals taken from the

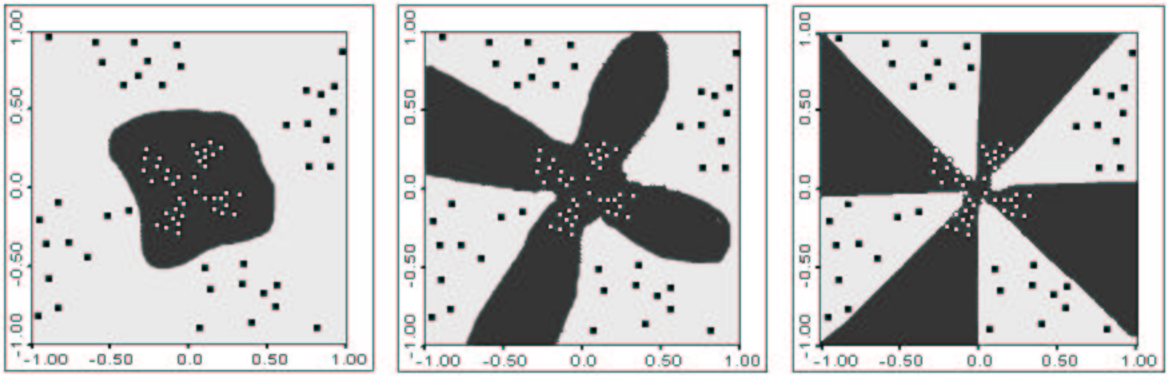

Figure 1: Artificial two-class classification problem. Black training points have to be discriminated against white training points. Left: Original decision function of SVM with RBF kernel ($\sigma = 0.2$), Center: decision function using slightly jittered kernel, Right: decision function facing full invariance.

Table 1: Experimental results on the EEG dataset

	Algorithm	Testing Error, %
EEG	SVM	9
	AJ SVM	6
	VRM-based	6

different electrodes located on a human's head. The difference between two particular signals[1] was taken as input for the algorithm. The input dimension (the signal length) is 150. The dataset consists of 413 training and 100 testing samples.

Raw data usage may appear to not be the best way of carrying out the classification. However, it was found to work well for SVMs. For example, the classification performance based on auto-regressive coefficients was significantly worse. The evident properties of these data are the invariances to the signal amplitude and the selection of the reference point of the "zero" level of the signal. These findings are also justified by the physical conditions of the EEG signal measuring process.

The results of the baseline SVM classifier based on Gaussian RBF kernel and SVMs with modified kernels are presented in Table 1. The hyper-parameters of all the algorithms were tuned according to cross-validation on the training set. The obtained values are $C = 25$, $\sigma = 1500$. The invariance-defining parameters are $\gamma = 0.55$ for VRM-based kernel, and for the analytically jittered (AJ) kernel the scaling range is $[0.5, 1.5]$.

Both methods provided an improvement to the classification performance according to the testing error. However, this improvement is hardly statistically significant (79% confidence only) since the size of the test set is 100 samples. This is a basic disadvantage of the competition setting caused by difficulties in data collection.

[1] Signals from the C3 and C4 electrodes, according to the standard labeling.

4 Discussion and Conclusion

The method of prior knowledge incorporation considered in this paper results in a kernel modification and exploits the standard SVM algorithm[2]. Kernel calculation can appear to be a computationally expensive part of the algorithm; although in the considered examples it was not the case. Tangent Distance [5] is a similar distance-based approach (although proposed in fact for neural networks). It is based on the tangent planes of the manifolds and appears to be a good practical solution when the object is a differentiable manifold. Here, we propose to define an *arbitrary object* as long as the distance calculation is reasonable.

The proposed approach has also close links with the regularization framework. Loosely speaking, regularization is used to enforce smoothness of the function in the vicinity of the training points. For a learning algorithm based on the squared loss function it is shown that, under certain assumptions, the approaches of adding virtual samples to the training set and adding a regularization term to the cost function are equivalent [9]. Our approach is based on the objects "built" on the samples and hence is a kind of limit case of the virtual sample approach. So, obviously, it has regularizing properties.

Since we propose an object definition based on combining the sample and some prior knowledge, the presented method naturally establishes a link between kernel methods and generative models. The general approach in this field is given in [10], where the Fisher kernel based on a metric defined on a parametric generative probability model is presented. However, in the described approach we arrive at the notion of a *model* from the aspect of prior knowledge incorporation, while the Fisher Kernel was proposed to deal with distorted sequences or sequences of variable length.

In conclusion, in this paper we present a general method to incorporate prior knowledge into kernel methods. It is based on modifying the setting of the problem by a transition from samples to objects, which are generated from them using some prior knowledge. Two implementations of the method were considered: analytical jittering and a new insight at the VRM-based classification algorithm. Particular examples were presented, and experiments on both artificial and real datasets showed useful results.

Acknowledgments

This research has been partially carried out in the framework of the European project LAVA, funded by the Swiss OFES project number 01.0412. It was also partially funded by the Swiss NCCR project (IM)2.

References

[1] C.J.C. Burges. Geometry and invariance in kernel-based methods. In B.Scholkopf, C.J.C. Burges, and A.J. Smola (eds.), *Advances in Kernel Methods - Support Vector Learning*, MIT Press, 1999.

[2] B. Scholkopf, C. Burges, and V. Vapnik. Incorporating invariances in support vector learning machines. In C. von der Malsburg, W. von Seelen, J. C. Vorbrĺuggen, and B. Sendhoff, (eds.), Artificial Neural Networks ICANN'96, pp. 47-52, Berlin, 1996. Springer Lecture Notes in Computer Science, Vol. 1112.

[3] O. Chapelle and B. Scholkopf. Incorporating invariances in nonlinear SVMs. In T.G. Dietterich, S. Becker and Z. Ghahramani, (eds.),*Advances in Neural Information Processing Systems*, vol. 14, pp. 609-616. MIT Press, Cambridge, MA, USA (2002)

[4] D. DeCoste, M.C. Burl. Distortion-invariant recognition via jittered queries. In *Computer Vision and Pattern Recognition, CVPR-2000*, June, 2000.

[2] However, if we restrict ourselves to using the sample-to-object distances, the asymmetry of the kernel matrix should be taken into account in the implementation.

[5] P. Simard, Y. LeCun, J. Denker, B. Victorri. Transformation invariance in pattern recognition, tangent distance and tangent propagation. In G. Orr and K. Muller, (eds.), *Neural Networks: Tricks of the trade.* Springer, 1998.

[6] V. Vapnik. Statistical Learning Theory. J.Wiley, NY, 1998.

[7] O. Chapelle, J. Weston, L. Bottou, and V. Vapnik. Vicinal risk minimization. In T.K. Leen, T.G. Dietterich, and V. Tresp, (eds.), *Advances in Neural Information Processing Systems*, vol. 13, pp. 416-422, 2001.

[8] V. Vapnik. The Nature of Statistical Learning Theory. Second edition, Springer-Verlag, NY, 2000.

[9] T.K. Leen. From data distributions to regularization in invariant learning. Neural Computation, vol. 7, no. 5, pp. 974-981, 1995.

[10] T. Jaakkola, and D. Haussler. Exploiting generative models in discriminative classifiers. In M.S.Kearns, S.A.Solla, D.A.Cohn (eds.) *Advances in Neural Information Processing Systems*, vol. 11, pp. 487-493, MIT Press, 1999.