Classification on Proximity Data with LP–Machines

Thore Graepel*, Ralf Herbrich*, Bernhard Schölkopf†,‡, Alex Smola†,‡
Peter Bartlett†, Klaus–Robert Müller‡, Klaus Obermayer*, Robert Williamson†

† Australian National University, FEIT, Canberra ACT 0200, Australia
‡ GMD FIRST, Rudower Chaussee 5, 12489 Berlin, Germany
∗ Technische Universität Berlin, Franklinstr. 28/29, 10587 Berlin
{graepel2,ralfh}@cs.tu–berlin.de,{bs,smola}@first.gmd.de,
Peter.Bartlett@anu.edu.au, klaus@first.gmd.de,
oby@cs.tu-berlin.de, Bob.Williamson@anu.edu.au

Abstract

We provide a new linear program to deal with classification of data in the case of functions written in terms of pairwise proximities. This allows to avoid the problems inherent in using feature spaces with indefinite metric in Support Vector Machines, since the notion of a margin is purely needed in input space where the classification actually occurs. Moreover in our approach we can enforce sparsity in the proximity representation by sacrificing training error. This turns out to be favorable for proximity data. Similar to ν–SV methods, the only parameter needed in the algorithm is the (asymptotical) number of data points being classified with a margin. Finally, the algorithm is successfully compared with ν–SV learning in proximity space and K–nearest-neighbors on real world data from Neuroscience and molecular biology.

1 Introduction

Support Vector (SV) learning has proven to be an effective algorithm for data classification. However, it is inherently connected to using quadratic (or even general convex) programming and kernels k satisfying Mercer's condition [16]. Whilst this is not a restriction in general it may sometimes be desirable to overcome this limitation. Such a case occurs if the data is available only in terms of an *implicit* proximity measure [4], which may not satisfy the properties of a metric at all. Proximity data are frequently encountered in fields like psychology, Neuroscience, molecular biology, and economics [5]. Whereas previous approaches focused on constructing a proper Euclidean metric by furnishing the feature space with a positive signature (see also [12]), we avoid the problem completely by switching to a different regularization approach that does not require these properties at all.

Moreover it is sometimes important to obtain solutions that have a pre–specified level of accuracy. In [10] we showed how this could be taken into account automatically in the SV case. The current modification is geared towards providing the same versatility also for classification of proximity data via linear programming. We apply the following modifications to standard SV machines:

- The relation between proximity data and kernels (sec. 2) leads to methods to use also functions $k(\mathbf{x}, \mathbf{x}')$ that do not satisfy Mercer's condition.
- By replacing the standard SV regularization term corresponding to the flattest function in feature space by a regularizer enforcing sparseness in the proximity representation we reduce the optimization problem from quadratic to linear (sec. 3).
- Introduction of an adaptive margin leads to an algorithm that automatically selects the number of basis functions, e.g. reference points, such that asymptotically a specified fraction of patterns is classified correctly with a margin (cf. sec. 4).

Finally we give a theoretical (sec. 5) and experimental (sec. 6) analysis of the properties of the new algorithm.

2 Proximities and Kernels

Suppose we want to learn $\mathbf{w}$ in a linear decision function f, i.e. classification of a data point $\mathbf{x} \in \mathcal{X}$ is carried out by computing the sign of f with

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b. \tag{1}$$

Kernels are introduced in SV–type learning algorithms by making use of an implicit representation of a feature map $\Phi : \mathcal{X} \to \mathcal{F}$ via

$$k(\mathbf{x}, \mathbf{x}') := \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle. \tag{2}$$

Here $\langle \cdot, \cdot \rangle$ is the dot product in some feature space $\mathcal{F}$. Any symmetric kernel satisfying Mercer's condition

$$\int_{\mathcal{X}} k(\mathbf{x}, \mathbf{x}') g(\mathbf{x}) g(\mathbf{x}') \geq 0 \text{ for all } g \in L_2(\mathcal{X}), \tag{3}$$

where $\mathcal{X}$ is a compact set, can be written as in (2). In practice, these assumptions may not always hold.

Kernels on Data For instance, we may not want to (or be unable to) analyze a given kernel k analytically. However, we still may be able to compute for a finite amount of data $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$ a map Φ such that k corresponds to a dot product in the linear span of the $\Phi(\mathbf{x}_i)$, provided the original dot product matrix $\mathbf{K}_{ij} := k(\mathbf{x}_i, \mathbf{x}_j)$ is nonnegative [9]:

Proposition 1 *Suppose the data* $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$ *and the kernel* k *are such that the matrix* $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ *is positive. Then it is possible to construct a map* Φ *into a feature space* $\mathcal{F}$ *such that* $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. *Conversely, for a map* Φ *into some feature space* $\mathcal{F}$, *the matrix* $\mathbf{K}_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$ *is positive.*

In particular, this result implies that given data $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$, and a kernel k which gives rise to a positive matrix $\mathbf{K}$, it is always possible to construct a feature space $\mathcal{F}$ of dimensionality $\leq \ell$ that we are implicitly working in when using kernels.

Kernels via Proximity Measures In other cases we may be only given a proximity or "distance" measure $p(\mathbf{x}, \mathbf{x}')$. Under the assumption that p was derived from a quadratic form by $p(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}) - \Phi(x'), \Phi(\mathbf{x}) - \Phi(x') \rangle^{\frac{1}{2}}$ we may apply the polarization equality to reconstruct the bilinear form by

$$k(\mathbf{x}, \mathbf{x}') = \frac{1}{2} \left(p(\mathbf{x}, \mathbf{0})^2 + p(\mathbf{x}', \mathbf{0})^2 - p(\mathbf{x}, \mathbf{x}')^2 \right). \tag{4}$$

From a numerical point of view the kernel should be reconstructed by choosing the origin $\Phi(\mathbf{0})$ in $\mathcal{F}$ at the center of mass $1/\ell \sum_i \Phi(\mathbf{x}_i)$ (see [15]).[1]

If the signature of the underlying space $\mathcal{F}$ is indefinite (e.g. relativistic space–time), one may still use (4) to reconstruct the quadratic form. Projective methods in $\mathcal{F}$, like the classification given by (1) or kernel PCA, however, can still be carried out (see [4, 12]).

Metrics and the Proximity Space Another way to deal with a proximity measure p not generated by a quadratic form with positive signature is the following: The functions $p(\mathbf{x}_i, \cdot)$ can be used as basis functions of a feature space directly [4]. Thus, given $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$, we define a *data–dependent* mapping Ψ by

$$\Psi : \mathbf{x} \mapsto (p(\mathbf{x}_1, \mathbf{x}), \ldots, p(\mathbf{x}_\ell, \mathbf{x}))^\top \tag{5}$$

and our decision functions (1) become

$$f(\mathbf{x}) = \mathbf{w}^\top \Psi(\mathbf{x}) + b\,, \tag{6}$$

which will be the basis of all of subsequent considerations. If we use $\mathbf{P}_{ij} = p(\mathbf{x}_i, \mathbf{x}_j)$ to denote the given proximity matrix, we see that the kernel matrix is given by $\mathbf{K} = \mathbf{P}^2$ which is positive definite by construction and thus can be used in standard SV–learning.

3 Constructing LP–Machines

For the classification task our goal is to find a function f that minimizes the following risk functional

$$R[f] := \int c(f(\mathbf{x}), y)\, dP(\mathbf{x}, y)\,, \tag{7}$$

where $c(\cdot, \cdot)$ is a given loss function. It is well known [16] that this problem cannot be solved directly, since

[1] The freedom to choose the origin is obvious from the data given, since inter–pattern distances only provide information about their *relative* distances. The dot product, however, depends on the *absolute* distance from the origin, too.

$P(\mathbf{x}, y)$ is generally unknown. Instead, we are given a training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\} \subset \mathcal{X} \times \{-1, +1\}$ and try to find some suitable f based thereon. Minimization of the empirical risk

$$R_{\text{emp}}[f] := \frac{1}{\ell} \sum_{i=1}^{\ell} c(f(\mathbf{x}_i), y_i) \tag{8}$$

is an ill–posed problem. Moreover, the solution will have poor generalization performance, unless further restrictions are imposed on f. Both problems can be addressed by adding a convex regularizer $Q[f]$ which effectively restricts the choice of models to a compact set. Hence for some $\lambda > 0$ we minimize

$$R_{\text{reg}}[f] := R_{\text{emp}}[f] + \lambda Q[f]. \tag{9}$$

For SV–learning one uses $Q[f] = \frac{1}{2}\|\mathbf{w}\|_2^2$ which leads to flat linear functions in feature space. While this is an appealing property for vectorial data, it is unfavorable for proximity data represented in the data–dependent proximity space (5). Here, it is preferable to enforce sparseness in the components of the vector $\mathbf{w}$ itself, because each component i therein requires the measurement of the proximity to the respective training example $\mathbf{x}_i$. The attractive reduction of the training set achieved by SV–learning does not carry over in the case of data–dependent representation.

Following the reasoning of [1, 2] with regard to feature selection, i.e. sparseness in the expansion coefficients of $\mathbf{w}$, we employ the following regularizer

$$Q[f] = \|\mathbf{w}\|_1 = \sum_{i=1}^{\ell} |w_i|. \tag{10}$$

Finally, the convexity criterion is easily computable in the general case and ensures restriction of the weight vector to a (pre)compact domain, too. One can show [14] that efficient capacity control is possible as well in this case. For the soft margin loss function [3], i.e.

$$c(f(\mathbf{x}), y) = \max(1 - yf(\mathbf{x}), 0) \tag{11}$$

we obtain the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{\ell}(\alpha_i + \alpha_i^*) + C\sum_{i=1}^{\ell}\xi_i \\ \text{subject to} & y_i f(\mathbf{x}_i) \geq 1 - \xi_i \\ & \alpha_i, \alpha_i^*, \xi_i \geq 0 \end{array} \tag{12}$$

where by virtue of (5) and (6)

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \underbrace{(\alpha_i - \alpha_i^*)}_{w_i} p(\mathbf{x}_i, \mathbf{x}) + b,$$

and C is some trade off constant to be adjusted separately. To convert (9) into (12) we split up α_i into α_i and α_i^* in order to eliminate the $|\cdot|$ in the objective function (with positive constrained α_i, α_i^*). Moreover we modified (11) in a standard way [3] by introducing slack variables to eliminate $\max(\cdot, \cdot)$.

Similar settings for vectorial data have been proposed recently in [1, 7]. The disadvantage is that it is quite difficult to find a meaningful interpretation of C and to adjust it properly.

4 Adaptive Margins

Several uniform convergence bounds (e.g. [13]) use the size of the model class and the number of margin errors, i.e. the number of training patterns with $y_i f(\mathbf{x}_i) \leq 1$ as a fundamental criterion to determine the confidence rates of the estimate. Also for this reason a modification of the algorithm described in the previous section would be desirable.

Both goals can be achieved by a slight modification of the original classification problem following the lines of [10]: we make the width of the margin ρ, so far set to 1 (cf. (11)) a variable of the optimization problem.

$$R_{\text{reg}}[f] := R_{\text{emp}}[f] + \sum_{i=1}^{\ell} |\alpha_i| - \nu\rho \tag{13}$$

Note that we dropped the regularization constant λ — as in the Support Vector case [11], one can show that keeping λ adds nothing: the decision function is invariant with respect to multiplication of α, ξ, b by some positive constant. Hence we may fix a priori $\sum_{i=1}^{\ell} |\alpha_i| = 1$ and solve

$$\begin{array}{ll} \text{minimize} & \frac{1}{\ell}\sum\limits_{i=1}^{\ell}\xi_i - \nu\rho \\ \text{subject to} & \sum\limits_{i=1}^{\ell}\alpha_i + \alpha_i^* = 1 \\ & y_i f(\mathbf{x}_i) \geq \rho - \xi_i \\ & \alpha_i, \alpha_i^*, \xi_i, \rho \geq 0 \end{array} \tag{14}$$

Again we used the trick of splitting up α_i into two positive variables α_i and α_i^* in order to obtain a purely linear optimization problem. In matrix notation (which is convenient when dealing with a linear optimizer) (14) can be stated as follows:

$$
\begin{array}{ll}
\text{minimize} & \mathbf{c}^\top \mathbf{a} \\
\text{subject to} & \begin{array}{lcl} \mathbf{A}\mathbf{a} & \geq & \mathbf{0} \\ \mathbf{d}^\top \mathbf{a} & = & 1 \end{array} \\
\text{where} & \begin{array}{lcl} \mathbf{a} & := & (\boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\xi}, \rho, b) \in {\mathbb{R}_0^+}^{3\ell+1} \times \mathbb{R} \\ \mathbf{c} & := & (\mathbf{0}, \mathbf{0}, \frac{1}{\ell}\mathbf{1}, -\nu, 0) \in \mathbb{R}^{3\ell+2} \\ \mathbf{A} & := & (\mathbf{P}_y, -\mathbf{P}_y, \mathbf{I}, -\mathbf{1}, \mathbf{y}) \in \mathbb{R}^{(3\ell+2)\times\ell} \\ \mathbf{d} & := & (\mathbf{1}, \mathbf{1}, \mathbf{0}, 0, 0) \in \mathbb{R}^{3\ell+2} \end{array}
\end{array}
$$

Here $(\mathbf{P}_y)_{ij} := y_i p(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{y} := (y_1, \ldots, y_\ell)^\top$, $\mathbf{1}$ denotes the vector of ones, $\mathbf{0}$ the vector of zeros, and $\mathbf{I}$ the unit matrix.

5 Theoretical Analysis

We may use the reasoning of [11] which was developed for SV Classification directly to analyze the theoretical properties of the algorithm.

Proposition 2 *Suppose we run ν–LP classification with k on some data with the resulting $\rho > 0$. Then*

(i) ν upper–bounds the fraction of margin errors.

(ii) $1-\nu$ is an upper bound on the fraction of patterns with a margin larger than ρ.

(iii) Suppose the data were drawn iid from a distribution $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$ such that neither $P(\mathbf{x}, y = 1)$ nor $P(\mathbf{x}, y = -1)$ contains any discrete component. Suppose, moreover, that the kernel is analytic and non-constant. With probability 1, asymptotically, ν equals the fraction of margin errors and $1 - \nu$ the number of patterns classified with a margin larger than ρ.

Proof
Ad (i) + (ii): Assume that we found the optimal solution in terms of $\alpha_i, \alpha_i^*, \xi_i, b, \rho$. Moreover denote by $1 - \nu_{\text{up}}$ the fraction of points being classified with a margin larger than ρ, and by ν_{low} the fraction of points with a margin smaller than ρ. Now we apply a variational argument in terms of ρ.

Decreasing ρ by a small amount Δ will cause a fraction of ν_{low} slack variables ξ_i to decrease by Δ, hence the overall change in the objective function would be $\Delta(\nu - \nu_{\text{low}})$. Since we assumed that the objective function attains a minimum for ν, we conclude that $\nu - \nu_{\text{low}} > 0$ which proves (i). Likewise, increasing ρ yields the same conclusion for (ii).

Ad (iii): It follows from the condition on $P(\mathbf{x}, y)$ that apart from some set of measure zero (arising from possible singular components), the two class distributions are absolutely continuous and can be written as integrals over distribution functions. As the kernel is analytic and non-constant, it cannot be constant in any open set — otherwise it would be constant everywhere. Therefore, functions f constituting the argument of the sign in the decision function (i.e. $\text{sign}(f(\mathbf{x}))$) transform the distribution over $\mathbf{x}$ into distributions such that for all f, and all $t \in \mathbb{R}$, $\lim_{\gamma\to 0} P(|f(x) + t| < \gamma) = 0$. Moreover we know that the class of these functions has well-behaved covering numbers [14], hence we get uniform convergence: for all $\alpha > 0$, $\lim\limits_{\gamma\to 0} \lim\limits_{\ell\to\infty} P(\sup_f \hat{P}_\ell(|f(\mathbf{x}) + t| < \gamma) > \alpha) = 0$. Hence, $\sup_f \hat{P}_\ell(|f(\mathbf{x}) + t| = 0)$ converges to zero in probability. In particular ($t = \pm\rho$) almost surely the fraction of points exactly on the margin tends to zero. Combining (i) and (ii) then shows that both fractions converge almost surely to ν. ■

6 Experiments

Artifical Data In order to allow for proper visualization we performed experiments on 2–D vectorial data. We used the city–block metric (L_1–metric) $p(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_1$ for the calculation of proximities (not to be confused with the L_1 regularizer (10)). The binary classification problem (crosses vs. dots) — not linearly separable in $\mathbb{R}^2$ — is shown in Fig. 1. We varied the parameter ν in order to assess its effect on the shape of the decision functions and the sparseness of the *representation* (5). For small values of ν the empirical risk is reduced to zero at the cost of a high–dimensional representation (proximities to all circled training objects required). Increasing the value of ν leads to a higher fraction of margin errors (see Proposition 2). However, with increasing ν the number of expansion coefficients is considerably reduced. The lower right part of Fig. 1 illustratetes the effective embedding space resulting from a learning with $\nu = 0.7$. Note, how the algorithm not only learned a decision function but also an appropriate representation in term of a small number of proximities.

Real-World Data We also evaluated the algorithm on real–world data. The data set called "cat cortex" consists of a matrix of connection strengths between

	A	V	SS	FL
Size	10	19	17	19
LP (0.05)	9.2 (15)	6.2 (16)	4.6 (17)	3.1 (10)
LP (0.1)	7.7 (14)	6.2 (15)	4.6 (16)	3.1 (10)
LP (0.2)	6.1 (5)	7.7 (11)	6.1 (8)	3.1 (9)
LP (0.3)	15.4 (2)	7.7 (4)	7.7 (3)	3.1 (7)
QP (0.05)	3.1	4.6	3.1	1.5
QP (0.1)	3.1	4.6	6.1	1.5
QP (0.2)	1.5	6.1	3.1	3.1
QP (0.3)	3.1	6.1	3.1	3.1
B-NN	3.1	1.5	3.1	4.6
W-NN	6.1	4.6	10.8	9.2

Table 1: Classification results for "cat cortex". Shown is the classification error estimated by leave–one–out crossvalidation. For LP–Machines, the average number of non–zero coefficients (the effective dimensionality of the proximity space chosen) is shown. The number in braces indicate the chosen values of ν. The last two lines contain base–line result of the best (B-NN) and worst (W-NN) K-nearest-neighbor results with $1 \leq K \leq 5$.

	H-α	H-β	M	GH
Size	72	72	37	30
LP (0.05)	1.8 (36)	5.0 (60)	0.5 (12)	0.0 (34)
LP (0.1)	1.3 (34)	5.8 (55)	0.5(10)	0.9 (29)
LP (0.2)	3.1 (16)	5.8 (37)	0.5(4)	0.9 (20)
LP (0.3)	5.0 (13)	10.7 (19)	5.0(2)	13.4 (14)
QP (0.05)	1.3	4.0	0.5	0.5
QP (0.1)	1.8	4.5	0.5	0.9
QP (0.2)	2.2	8.9	0.5	0.9
QP (0.3)	2.2	10.3	0.5	11.6
B-NN	1.3	3.6	0.0	1.8
W-NN	2.2	6.7	0.0	4.5

Table 2: Classification results for Protein data. The experimental setup is identical to the one of table 1.

65 cortical areas of the cat. The data was collected by Scannell [8] from text and figures of the available anatomical literature and the connections are assigned proximity values p as follows: self-connection ($p = 0$), strong and dense connection ($p = 1$), intermediate connection ($p = 2$), weak connection ($p = 3$), and absent or unreported connection ($p = 4$). From functional considerations the areas can be assigned to four different regions: auditory (A), visual (V), somatosensory (SS), and frontolimbic (FL). The classification task is to discriminate between these four regions, each time one against the three others.

The second data set consists of a proximity matrix from the structural comparison of 224 protein sequences based upon the concept of evolutionary distance. The majority of these proteins can be assigned to one of four classes of globins: hemoglobin-α (H-α), hemoglobin-β (H-β), myoglobin (M), and heterogenous globins (GH). The classification task is to assign proteins to one of these classes, one against the rest.

We compared the LP–algorithm presented in Sec. 4 with the corresponding SV–algorithm (see [11, 4]) and K-nearest-neighbor ($K \in \{1, 2, 3, 4, 5\}$), the natural choice of classifier for proximity data. Crossvalidation was performed to estimate the misclassification probability (cf. tables 1 and 2). While the results in terms of accuracy are roughly equivalent, especially for small values of ν, for LP, QP, and B (est) K-nearest-neighbor it should be noted that only in the case of LP–learning the dataset is reduced by up to 98% ("proteins" myoglobin and $\nu = 0.2$). Considering that each proximity corresponds to a single measurement of evolutionary distance between two proteins, the striking advantage of sparse proximity representation should be obvious.

7 Discussion

In this paper we considered the problem of constructing decision functions for data given in terms of pairwise proximities. Whereas former approaches [4] regularized in proximity space via the flatness of the decision function (SV–approach) we proposed a linear programm which explicitly enforces sparsity in proximity space. Thus the presented algorithm not only finds *a decision function* with small error but at the same time a computationally efficient *representation* due to sparseness in the weight vector.

Apart from the sparseness considerations it should also be noted that the choice of the L_1–norm as regularizer corresponds to viewing the distance between data in proximity space as measured by the L_∞–norm which is dual to the L_1–norm [6]. This means that the difference between two examples is dominated by the maximum difference between proximities to the refer-

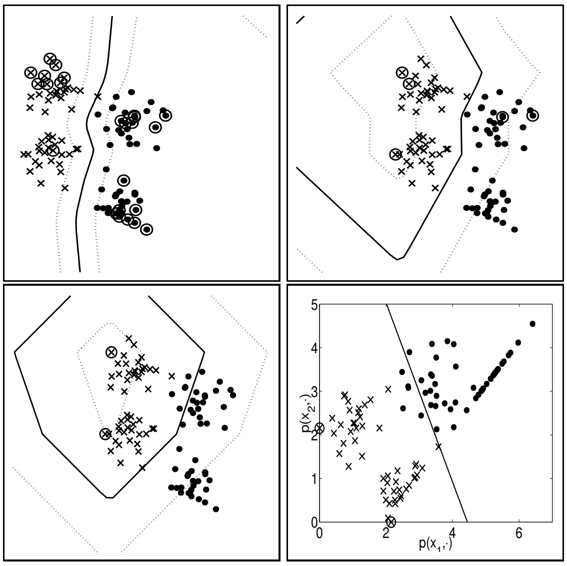

Figure 1: Decision functions (solid lines) in a simple two–class classification problem found by ν–LP Machines when using the city–block metric $p(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_1$ (upper left: $\nu = 0.1$, upper right: $\nu = 0.5$, lower left: $\nu = 0.7$). Circled are training examples $\mathbf{x}_i$ with $\mathbf{w}_i \neq 0$. The dashed lines indicate the resulting margin ρ. By increasing ν and thus sacrificing margin errors, the dimensionality of the proximity space is reduced. The lower right figure shows the data points embedded in the 2–D proximity space ($\nu = 0.7$). Note, that the linear decision function in proximity space (lower right) corresponds to the non–linear decision function in data space $\mathbb{R}^2$ (lower left).

ence objects (training set).

Acknowledgements The authors would like to thank Gunnar Rätsch, Peter Bollmann-Sdorra, and Ulrich Kockelkorn. This work was supported by a grant of the DFG Ja 379/51 and by the Technical University of Berlin via Forschungsinitiativprojekt FIP 13/41.

References

[1] K. Bennett. Combining support vector and mathematical programming methods for induction. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - SV Learning*, pages 307–326, Cambridge, MA, 1999. MIT Press.

[2] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Feature selection via mathematical programming. *INFORMS Journal on Computing*, 10:209–217, 1998.

[3] C. Cortes and V. Vapnik. Support vector networks. *M. Learning*, 20:273 – 297, 1995.

[4] T. Graepel, R. Herbrich, and K. Obermayer. Classification on pairwise proximity data. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *NIPS*, volume 11. MIT Press, Cambridge, MA, 1999. in press.

[5] T. Graepel and K. Obermayer. A stochastic self–organizing map for proximity data. *Neural Computation*, 11:139–155, 1999.

[6] O. Mangasarian. Arbitrary–norm separating plane. Technical report, University of Wisconsin, 1997. TR-97-07.

[7] O. L. Mangasarian. Generalized support vector machines. Technical Report 98-xx, University of Wisconsin, Computer Sciences Department, Madison, 1998.

[8] J. W. Scannell, C. Blakemore, and M. P. Young. Analysis of connectivity in the cat cerebral cortex. *The Journal of Neuroscience*, 15(2):1463–1483, 1995.

[9] B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, Munich, 1997.

[10] B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson. Support vector regression with automatic accuracy control. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of ICANN'98*, Perspectives in Neural Computing, pages 111 – 116, Berlin, 1998. Springer Verlag.

[11] B. Schölkopf, A. Smola, P. Bartlett, and R. Williamson. New support vector algorithms. *Neural Computation*, 1999. submitted, also NeuroCOLT TR–31–89.

[12] B. Schölkopf, A. J. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. Technical Report 44, MPI für biologische Kybernetik, December 1996.

[13] J. Shawe-Taylor, P. Bartlett, R. Williamson, and M. Anthony. A framework for structural risk minimization. In *COLT*, 1996.

[14] A. Smola, R. Williamson, and B. Schölkopf. Generalization bounds for convex combinations of kernel functions. Technical Report NC-TR-98-022, Royal Holloway College, University of London, UK, 1998.

[15] W. S. Torgerson. *Theory and Methods of Scaling*. Wiley, New York, 1958.

[16] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, N.Y., 1995.