Sparse Greedy Matrix Approximation for Machine Learning

Alex J. Smola ALEX.SMOLA@ANU.EDU.AU
Department of Engineering and RSISE, Australian National University, Canberra, ACT 0200, Australia

Bernhard Schölkopf BSC@MICROSOFT.COM
Microsoft Research, St. George House, 1 Guildhall Street, Cambridge CB2 3NH, UK

Abstract

In kernel based methods such as Regularization Networks large datasets pose significant problems since the number of basis functions required for an optimal solution equals the number of samples. We present a sparse greedy approximation technique to construct a compressed representation of the design matrix. Experimental results are given and connections to Kernel-PCA, Sparse Kernel Feature Analysis, and Matching Pursuit are pointed out.

1. Introduction

Many recent advances in machine learning such as Support Vector Machines [Vapnik, 1995], Regularization Networks [Girosi et al., 1995], or Gaussian Processes [Williams, 1998] are based on kernel methods. Given an m-sample $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ of patterns $x_i \in X$ and target values $y_i \in Y$ these algorithms minimize the regularized risk functional

$$\min_{f \in \mathcal{H}} R_{\text{reg}}[f] = \frac{1}{m} \sum_{i=1}^{m} c(x_i, y_i, f(x_i)) + \frac{\Lambda}{2} \|f\|_{\mathcal{H}}^2. \quad (1)$$

Here $\mathcal{H}$ denotes a reproducing kernel Hilbert space (RKHS) [Aronszajn, 1950], $\|\cdot\|_{\mathcal{H}}$ the corresponding norm, $\Lambda > 0$ a regularization constant, and $c : X \times Y \times Y \to \mathbb{R}$ a cost function penalizing the deviation between $f(x_i)$ and y_i at location x_i.

1.1 Representer Theorem

Kimeldorf and Wahba [1971] and later Cox and O'Sullivan [1990] proved the following which characterizes the optimal solution of (1).

Theorem 1 (Representer Theorem) *Denote by $k : X \times X \to Y$ the kernel of the corresponding RKHS $\mathcal{H}$. Then the minimizer of (1) can be described by (the x_i are the training patterns):*

$$f(x) = \sum_{i=1}^{m} \alpha_i k(x_i, x) \quad (2)$$

Thus rather than dealing with a (possibly) infinite dimensional Hilbert space $\mathcal{H}$ all one has to do is find a *finite* set of parameters α_i to obtain an optimal solution in $\mathcal{H}$. This finding led to successful algorithms such as the ones described above.

1.2 Sparsity

While (2) is easy to deal with if m is not too large (the memory requirements for storing the symmetric positive semidefinite matrix $K_{ij} := k(x_i, x_j)$ are quadratic in m), large datasets pose a serious problem. Hence one has to find means to limit the number of nonzero coefficients α_i occuring in expansion (2). Support Vector Machines address this problem by introducing a cost function $c(x, y, f(x))$ that eliminates the contribution of basis functions $k(x_i, x)$ corresponding to points which have a large margin (classification) [Schölkopf et al., 1995] or are close to their target values[Vapnik et al., 1997]. However, if the data is noisy the improvement can be negligible [Smola, 1998].

Another approach is to add (yet another) regularization term penalizing the ℓ_1 norm of the expansion coefficients [Mangasarian, 1965, Chen et al., 1999, Girosi, 1998]. However, this does not alleviate the problem that we have to compute (and invert) the matrix $K_{ij} := k(x_i, x_j)$ for $i, j \in [m]$ ($[m] := \{1, \ldots, m\}$). Since the latter scales with $O(m^3)$ (except for special matrices) this approach is not suitable for large datasets either.

1.3 Specific Assumptions

Unless specified otherwise we assume that

$$c(x, y, f(x)) = \frac{1}{2}(y - f(x))^2. \quad (3)$$

This is done for the simplicity of the presentation and Section 6 will show how under different conditions on c the proposed techniques could be applied, too. Using Theorem 1, (3), and the fact that in RKHS

$$\langle k(x_i, x), k(x_j, x)\rangle_{\mathcal{H}} = k(x_i, x_j) \tag{4}$$

the problem (1) becomes [Girosi et al., 1995]

$$\min_{\alpha \in \mathbb{R}^m} R_{\text{reg}}[\alpha] = \frac{1}{2m}\|y - K\alpha\|^2 + \frac{\Lambda}{2}\alpha^\top K \alpha. \tag{5}$$

The minimum of (5) is given by

$$\alpha = (K^\top K + m\Lambda K)^{-1} K^\top y = (K + m\Lambda \mathbf{1})^{-1} y \tag{6}$$

if $K = K^\top$ (which is the case for reproducing kernel Hilbert spaces) and K is of full rank. If, however, we pick only a subset of basis functions, say $I = \{i_1 \dots i_n\} \subset [m]$, the optimization problem can be stated as follows

$$\min_{\alpha \in \mathbb{R}^n} R_{\text{reg}}[\alpha] = \frac{1}{2m}\|y - K^{mn}\alpha\|^2 + \frac{\Lambda}{2}\alpha^\top K^{nn} \alpha.$$

Here $\alpha \in \mathbb{R}^n$, K^{nn} is the $n \times n$ submatrix of K given by $K^{nn}_{jj'} = K_{i_j i_{j'}}$ and $K^{mn}_{jj'} = K_{j i_{j'}}$. The minimum can be found by

$$\alpha = (K^{mn\top} K^{mn} + m\Lambda K^{nn})^{-1} K^{mn\top} y \tag{7}$$

which is only an $O(m \cdot n + n^3)$ rather than an $O(m^3)$ operation.

1.4 Matrix Approximations

This leads us to a third method how to solve this dilemma. The idea is to find a good subset I beforehand. One has to bear in mind that such a subset is not optimal in terms of Theorem 1 any more. However, if one manages to eliminate irrelevant basis functions the solution on the subset (of basis functions) may be *close* to optimal. This is the case in particular, if the spectrum of of the design matrix K decays rapidly: hence there exists a subspace S such that $\tilde{K} = KP_S$ is similar to K, namely the space corresponding to the first m' largest singular values of K (P_S denotes the projector on the subspace S).

This paper studies several ways of finding matrices $\tilde{K}$ of lower rank (and other numerical desirable properties like sparsity) such that the norm of the residual matrix $K - \tilde{K}$ is minimized. Conventionally this is the Frobenius norm

$$\|\tilde{K} - K\|^2_{\text{Frob}} := \sum_{i,j=1}^{m} (\tilde{K} - K)^2_{ij}. \tag{8}$$

It is easy to check that the Frobenius norm is the 2-norm of the the singular values, since for any matrix X with singular value decomposition $X = L\Sigma R$ we have

$$\|X\|^2_{\text{Frob}} = \operatorname{tr}(X^\top X) = \operatorname{tr}(R^\top \Sigma L^\top L \Sigma R) = \operatorname{tr}(\Sigma^2). \tag{9}$$

Therefore minimization of (8) also minimizes the conventional operator norm (the largest singular value)

$$\|\tilde{K} - K\|_2 := \max_{\|x\|_2 \leq 1} \|(\tilde{K} - K)x\|_2. \tag{10}$$

Likewise, in some cases when we deal with function spaces, rather than the entries of K explicitly, we will try to minimize the trace of a symmetric positive semidefinite matrix of residuals $K - \tilde{K}$. By definition, this is then equivalent to minimizing the 1-norm of the eigenvalues of that matrix.

See Section 6.4 for further motivation why good approximations $\tilde{K}$ exist if the spectrum of K is rapidly decaying. In particular we will look for ways of finding such subsets I with computational cost not larger than $O(m^2)$, and, if possible $O(mn)$. Moreover, we seek an algorithm that has smaller memory requirements than $O(m^2)$ since storage of the complete design matrix K_{ij} may be out of question on large datasets.

1.5 Outline

The rest of the paper is organized as follows. Section 2 describes the basic approximation algorithm. Section 4 deals with ways of finding a good basis function to add to the subset of functions under consideration, and Section 3 describes orthogonalization procedures once a particular set of functions has been selected. Experimental results are given in Section 5. Finally, Section 6 points out applications of the basic theory to other machine learning problems. Relations to other algorithms (in particular to Sparse Kernel Feature Analysis [Smola et al., 1999]) are given in Section 7.

2. Greedy Approximation Algorithms

In the following denote by K_i the *columns* of the matrix K, and by $k_i \in \mathcal{H}$ the basis functions $k(x_i, \cdot)$ generating the columns by $K_{ij} = k_i(x_j)$. Moreover denote by T an $n \times m$ matrix containing expansion coefficients for an approximation of the columns of K, i.e.

$$\tilde{K}_i = \sum_{j=1}^{n} K_{i_j} T_{ji}. \tag{11}$$

Thus $K_i - \tilde{K}_i$ are the residuals of the approximation. A sensible requirement for any such expansion is that

$T_{ii_j} = \delta_{ij}$, or, in other words, that any K_l in the subset of selected basis functions is described by itself. Moreover we would like to find the subset I and the remaining coefficients T_{ij} such that the approximation (11) is good.[1]

2.1 A Matching Pursuit Variant

Finding an optimal subset I is a combinatorial problem since there exist $\binom{n}{m}$ possibilities. A possible strategy to tackle this problem is to use a greedy iterative algorithm much in the spirit of [Mallat and Zhang, 1993, Natarajan, 1995, Schölkopf et al., 1999], however with the difference that we are not approximating one single target function but a matrix K i.e. m columns (or alternatively m basis functions).

Hence we proceed as follows: (1) find the index $\hat{\imath}$ from the list of possible candidates $[m]\backslash I$ of the basis functions / columns that approximates the columns K_i / basis functions k_i best. (2) find a decomposition of K_i into $\tilde{K}_i$ involving $K_{\hat{\imath}}$ and $K_{i_1}, \ldots, K_{i_m}$, i.e. find suitable matrix elements T_{ij}. (3) use the residuals $K_i - \tilde{K}_i$ as the new set of basis functions and repeat.

2.2 A Probabilistic Speedup

Unfortunately, step (1) may be too expensive to carry out explicitly. Even if the computational cost for assessing the improvement by using column K_i was only $O(m)$ (which is very likely, since all other basis functions should be considered), the total cost would already be $O(m^2)$. This is more than what we want to accept in most cases.

The trick is to consider only a random subset M of fixed size, say κ, and pick the best basis function $\hat{\imath}$ from this set rather than performing an exhaustive search over all possible indices $i \in [m]\backslash I$. This is a feasible strategy due to the following lemma:

Lemma 2 (Maximum of Random Variables) *Denote by $\xi_1, \ldots, \xi_m$ identically distributed independent random variables with the common cumulative distribution function $F(\xi_i)$. Then the cumulative distribution function of $\xi := \max_{i\in[m]} \xi_i$ is $(F(\xi))^m$.*

In particular, for the uniform distribution on $[0, 1]$, $\max_{i\in[\kappa]} \xi_i$ is distributed according to ξ^κ. Thus, in order to obtain an estimate that is with probability 0.95 among the best 0.05 of all estimates, a random subsample of size $\lceil \log 0.05/\log 0.95\rceil = 59$ will guarantee nearly as good performance as if we considered the whole set of basis functions.

2.3 Prototype of an Algorithm

This allows us to state a prototype of an algorithm for matrix approximation (Algorithm 1).

Algorithm 1 Matrix Approximation Prototype

```
input: K or functions k_i, bound on residuals ϵ
input: sub  PickBestColumn, ProjectOutColumn,
  and BoundResiduals
  n = 0, I = {}, T = 0
  repeat
    n++
    Draw random subset M from [m]\I
    i_n = PickBestColumn(M, K, T)
    T^new = ProjectOutColumn(K, T, I)
    ε = BoundResiduals(K, T)
  until ε < ϵ
output: n, T, I, ε
```

We investigated whether this assumption is true on two real world datasets (`Abalone` and `Boston Housing` from the UCI repository, and the USPS database of handwritten characters). In all cases we used Gaussian RBF kernels with $2\sigma^2 = 0.5d$ (d is the dimensionality of the data; see Section 5 for further details). Figure 1 shows that a subset of size 59 is large enough to yield near optimal performance. The same results were obtained on the USPS postal and the `Boston Housing` dataset. Note, however, that in later stages of the algorithm the distribution of improvements may become very long-tailed, which e.g. led to the edge in Figure 3. Such effects, however, can easily be taken into account by a rearrangement of the chosen patterns afterwards.

2.4 Estimating a bound on the residuals

$$\varepsilon = \|K - \tilde{K}\|^2_{\text{Frob}} = \sum_{i,j=1}^{m} (K - \tilde{K})^2_{ij}, \qquad (12)$$

i.e. computing the error, is of cost $O(m^2)$. This is more expensive than what we often are willing to accept per iteration (if we strive for an $O(mn)$ algorithm).

If we treat (12) as a sum of random variables, say $\xi_i := \sum_{j=1}^m (K-\tilde{K})^2_{ij}$ and the total residuals as $\sum_{i=1}^m \xi_i$ we can apply uniform convergence theory to estimate (12) by considering only a subset of the ξ_i. This is useful since in the case of the Frobenius norm we will only consider a subset M of the columns of $K - \tilde{K}$ for the selection of a suitable column anyway (cf. Section

[1] Optimality with respect to some norms may too expensive to achieve numerically since it can be of order $O(m^2)$ or worse, thus one will have to choose a suitable norm as well. See section 3 for possible choices of such norms.

2.2). Moreover the values of ξ_i for $i \in M$ will be readily available at no additional cost (cf. Section 4). The following bound is a simple corollary of a theorem of Hoeffding [1963].

Corollary 3 (Smola et al. [1999])
Let $\xi_1, \ldots, \xi_m$ be independent identically distributed bounded random variables, falling into the interval $[a, a+b]$ with probability one. Denote their average by $S_m = \frac{1}{m}\sum_i \xi_i$. Moreover denote by $\xi_{s(1)}, \ldots, \xi_{s(m')}$ with $m' < m$ a subset of the same random variables (with $s : \{1, \ldots, m'\} \to \{1, \ldots, m\}$ being an injective map), and $S_{m'} = \frac{1}{m'}\sum_i \xi_{s(i)}$. Then for any $\epsilon > 0$ one has

$$\left.\begin{array}{l}\Pr\{S_m - S_{m'} \geq \epsilon\} \\ \Pr\{S_{m'} - S_m \geq \epsilon\}\end{array}\right\} \leq e^{-\frac{2mm'\epsilon^2}{(m-m')b^2}} \tag{13}$$

Hence, if we are satisfied with the confidence level of (13) we may simply reuse the values of the 2-norm of the columns to be considered for basis function selection. Otherwise, we might as well include more basis function in the selection process by considering a larger subset M such that the latter would benefit from more computations at the same time.

Finally, note that a similar technique can also be applied when computing the trace of the residual matrix as needed in the approximation of function spaces. There, however, this is not crucial for the numerical feasibility of the algorithm since an exact computation can be carried out in $O(mn)$ steps, too (see (22)).

3. Orthogonalization Procedures

Let us assume for the moment, that by some procedure, we picked $\hat{\imath}$ as the next column / basis function to be included in our list of basis functions. Now we have to remove the contribution of the corresponding column / basis function from K.

3.1 Column Space

The immediate choice when dealing with approximations in the Frobenius norm is to find coefficients T_{ij} such that $\tilde{K}$ minimizes the distance to K in $\|\cdot\|_{\text{Frob}}$. For simplicity let us start with the first iteration ($n = 1$). Here we have to solve the problem ($\tilde{K}_i = T_{i1}K_i$)

$$\operatorname*{argmin}_{T_{11},\ldots T_{m1}} \sum_{i=1}^{m} \|K_i - T_{i1}K_{\hat{\imath}}\|^2. \tag{14}$$

Solving for T_{i1} yields

$$T_{i1} = \|K_{\hat{\imath}}\|^{-2}\langle K_{\hat{\imath}}, K_i\rangle \tag{15}$$

and consequently $\langle K_i - \tilde{K}_i, K_{\hat{\imath}}\rangle = 0$ for all i. In other words, $\operatorname{span}\{(K_i - \tilde{K}_i)|i \in [m]\} \perp K_{\hat{\imath}}$.

It is easy to check that for larger than one-dimensional spaces this statement still holds (orthogonal projections are optimal in Hilbert spaces). Hence, in subsequent iterations, say, at step j, it is sufficient to apply (14) with the residuals $K_i - \tilde{K}_i^{\text{old}}$ rather than K_i in place since this will lead to a new set of residuals with

$$(K_i - \tilde{K}_i^{\text{new}}) \perp \operatorname{span}\{K_{i_j}|j \in [n]\} \tag{16}$$

as required for optimality. The computational cost per iteration is $O(m^2)$ (since each orthogonalization costs $2m$ multiply-add operations). We asumed that the residuals are cached (causing an $O(m^2)$ memory requirement). The algorithm would be $O(n \cdot m^2)$ otherwise, since one has to recompute the residuals at every step again. This may be more expensive than what we are willing to accept in practice.

3.2 Function Space

An alternative is to orthogonalize in function space rather than column space. The advantage is that the dot product $\langle k(x_i, \cdot), k(x_j, \cdot)\rangle = k(x_i, x_j)$ is $O(1)$, at least if dependencies on the dimensionality of the inputs are ignored. Moreover, we obtain an approximation that is close to the basis functions in the RKHS under consideration and not only at the training patterns x_i where k is evaluated. Finally, as we shall see, both the approximating matrix $\tilde{K}$ and the matrix of residuals $K - \tilde{K}$ are symmetric and positive semidefinite.

The disadvantage is, that by such a decomposition we will not achieve an optimal decrease in the Frobenius norm of the residuals. But it will allow an exact measurement of the 1-norm at $O(mn)$ cost. The strategy works as follows, again, starting from the first iteration. Denote by

$$\tilde{k}_i := \sum_{j=1}^{n} k_{i_j} T_{ij} \tag{17}$$

and thus for $n = 1$ we have $\tilde{k}_i = T_{i1}k_{\hat{\imath}}$. If we want to minimize

$$\sum_{i=1}^{m} \|k_i - \tilde{k}_i\|^2_{\mathcal{H}} = \sum_{i=1}^{m} \|k_i - T_{i1}k_{\hat{\imath}}\|^2_{\mathcal{H}} \tag{18}$$

we obtain in analogy to Section 3.1

$$T_{i1} = \|k_{\hat{\imath}}\|^{-2}_{\mathcal{H}}\langle k_i, k_{\hat{\imath}}\rangle = K_{\hat{\imath}\hat{\imath}}^{-2} K_{i\hat{\imath}} \tag{19}$$

Again, the same considerations regarding the orthogonality of the residuals $k_i - \tilde{k}_i$ wrt. the chosen basis functions $k_{i_1}, \dots k_{i_j}$ apply:

$$\operatorname{span}\{(k_i - \tilde{k}_i) | i \in [m]\} \perp \operatorname{span}\{k_{i_j} | j \in [n]\} \qquad (20)$$

Caching $\tilde{k}_i$ as in column space is not an option since the basis functions are abstract quantities. Therefore, at iteration n computing the dot product between all residuals $(k_i - \tilde{k}_i)$ and $(k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}})$ is of order $O(mn)$: if $\tilde{k}_i$ was chosen optimally according to the orthogonalization procedure described above it follows from (20)

$$\langle k_i - \tilde{k}_i, k_j - \tilde{k}_j \rangle = \langle k_i, k_j - \tilde{k}_j \rangle. \qquad (21)$$

This implies that, as pointed out before, the matrix of residuals $K - \tilde{K}$ is *symmetric* and positive semidefinite. Moreover we can compute the trace of $K - \tilde{K}$ (which is also the sum of the eigenvalues) by

$$\operatorname{tr}(K - \tilde{K}) = \sum_{i=1}^{m} K_{ii} - \langle k_i, \tilde{k}_i \rangle \qquad (22)$$

which is an $O(mn)$ operation, and in particular, involves no additional cost over the orthogonalization.

3.3 Kronecker Metric

As one can see from (15) and (19) the metric on the space of columns / kernel functions plays the crucial role in defining how projections are carried out. Rather than using a data-dependent metric we could think of a much more drastic simplification: to define the dot product between two columns / basis functions to be the Kronecker delta, i.e.

$$\langle k_i, k_j \rangle_K := \delta_{ij}. \qquad (23)$$

This is a very crude simplification, however, it still leads to a practical projection rule: to remove the basis function $\hat{\imath}$ and leave to the rest of the basis functions / columns untouched. One can check that this is equivalent to finding the best approximation of the columns K_i if we choose K^{-2} as the metric tensor, since then $K_i^\top K^{-2} K_j = \delta_{ij}$.

Unfortunately this is quite meaningless so we leave it at the observation that other metrics could be used, too.

4. Selection Rules

The problem which basis function should be selected naturally involves a tradeoff between the quality of the selection and the amount of computational resources needed to compute the latter. Therefore we present three possible choices for selecting a suitable column or basis function.

4.1 Frobenius Norm

One possibility is to seek the column $K_{\hat{\imath}}$ that yields the largest improvement in minimizing $\|K - \tilde{K}\|^2_{\text{Frob}}$, where during the previous i iterations the columns of $\tilde{K}$ were chosen such that (16) holds. Since optimal projections have to be orthogonal to the previous ones we can write (for fixed $\hat{\imath}$)

$$\begin{aligned} & \|K - \tilde{K}^{\text{new}}\|^2_{\text{Frob}} && (24)\\ = & \sum_{i=1}^{m} \|K_i - \tilde{K}_i^{\text{old}} - t_i(K_{\hat{\imath}} - \tilde{K}_{\hat{\imath}}^{\text{old}}))\|^2 && (25) \end{aligned}$$

where

$$t_i = \|K_{\hat{\imath}} - \tilde{K}_{\hat{\imath}}^{\text{old}}\|^{-2} \langle K_i - \tilde{K}_i^{\text{old}}, K_{\hat{\imath}} - \tilde{K}_{\hat{\imath}}^{\text{old}} \rangle. \qquad (26)$$

Thus the reduction in the residuals is given by

$$\begin{aligned} & \|K - \tilde{K}^{\text{old}}\|^2_{\text{frob}} - \|K - \tilde{K}^{\text{new}}\|^2_{\text{frob}} && (27)\\ = & \|K_{\hat{\imath}} - \tilde{K}_{\hat{\imath}}^{\text{old}}\|^{-2} \sum_{i=1}^{m} \langle K_i - \tilde{K}_i^{\text{old}}, K_{\hat{\imath}} - \tilde{K}_{\hat{\imath}}^{\text{old}} \rangle^2 \end{aligned}$$

Therefore the selection criterion is to evaluate (27) for all possible $\hat{\imath} \in M$ (the subset of size κ to be analyzed). The complexity per evaluation of (27) is $O(m^2)$ if the residuals are cached, and $O(nm^2)$ otherwise.

4.2 Function Space

If we choose the function space perspective, a very similar selection rule can be developed, however with lower computational complexity. In analogy to (24) we minimize

$$\sum_{i=1}^{m} \|k_i - \tilde{k}_i^{\text{old}} - t_i(k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}}))\|^2_{\mathcal{H}} \qquad (28)$$

where

$$t_i = \|k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}}\|^{-2}_{\mathcal{H}} \langle k_i - \tilde{k}_i^{\text{old}}, k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}} \rangle_{\mathcal{H}}. \qquad (29)$$

Again, the reduction in the residuals is given by

$$\begin{aligned} & \sum_{i=1}^{m} \|k_i - \tilde{k}_i^{\text{old}}\|^2_{\mathcal{H}} - \|k_i - \tilde{k}_i^{\text{new}}\|^2_{\mathcal{H}} && (30)\\ = & \|k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}}\|^{-2}_{\mathcal{H}} \sum_{i=1}^{m} \langle k_i - \tilde{k}_i^{\text{old}}, k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}} \rangle^2 \\ = & \langle k_{\hat{\imath}}, k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}} \rangle^{-1} \sum_{i=1}^{m} \langle k_i, k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}^{\text{old}} \rangle^2 && (31) \end{aligned}$$

where (31) follows from the orthogonality between the residuals $k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}$ and the approximations $\tilde{k}_i$. Computing (31) is an $O(mn)$ operation and thus cheaper than selection criteria involving the Frobenius norm.

4.3 Columns

A third criterion can be obtained from a related algorithm - Sparse Kernel Feature Analysis [Smola et al., 1999]. There, the selection criterion is to choose the basis function accounting for the largest variance on the training data.

In the present context this translates into choosing the column / basis function with the largest 2-norm of the column, i.e.

$$\hat{\imath} = \operatorname*{argmax}_{i \in M} \|K_i - \tilde{K}_i\|^2. \tag{32}$$

This criterion can be even cheaper to compute than the previous ones. If the residuals are cached (which is an $O(m^2)$ by itself, though) the rhs of (32) is of order $O(m)$. Without caching the cost is comparable to the previous section, i.e. it is $O(mn)$.

It is impossible to claim superiority of one of the three presented approaches (or at least not for Section 4.1 and Section 4.2). They all represent design choices regarding which target function is minimized and how much computational effort one is willing to spend in order to obtain a satisfactory solution.

While the Frobenius norm perspective may seem the most appealing at first glance, the function space view offers decompositions that leave symmetric, positive semidefinite matrices both in $\tilde{K}$ and $K - \tilde{K}$ which is a big advantage, e.g. in Interior Point codes for mathematical programming [Vanderbei, 1994].

5. Experiments

5.1 Datasets

In order to compare the algorithm with a convenvional regularization network we looked at regression problems of moderate size (up to 5000 samples) in order to be able to invert matrices and compute the spectrum of the design matrix. Our algorithm would be amenable to much larger datasets, however, no comparison would have been possible in the latter case.

We chose the `Boston Housing` and the `Abalone` dataset from the UCI Repository [Blake et al., 1998] and the USPS database of handwritten digits. The first data is of size 506 (350 training, 156 testing), the `Abalone` dataset of size 4177 (3000 training and 1177 testing). In the first two cases the data was rescaled to zero mean and unit variance, coordinate-wise, while the USPS dataset remained unchanged. Finally, the gender encoding in `Abalone` (male/female/infant) was mapped into $\{(1,0,0),(0,1,0),(0,0,1)\}$. We used Gaussian RBF-kernels where the kernel width σ and the regularization constant Λ were chosen to yield optimal performance on the test set.[2]

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} \text{ where } \sigma > 0 \tag{33}$$

We obtain $\sigma/d = 1, \Lambda m = 0.1$ for the `Abalone` dataset, and $\sigma/d = 10, \Lambda m = 0.01$ on the `Boston Housing` database. The parameter range was analyzed in logarithmic scale, i.e. $\ldots, 0.1, 0.2, 0.5, 1, \ldots$.

5.2 Algorithms

For convenience we give an explicit description of a projection algorithm working in function space (Algorithm 2). Since it is $O(nm)$ per iteration (rather than $O(m^2)$ for the Frobenius norm), all experiments were carried out using Algorithm 2.

Algorithm 2 Approximation in Function Space

input: basis functions k_i, bound on residuals ϵ
  $n = 0, I = \{\}, T = 0, K^{\text{cache}} = 0$
  $\{K^{\text{c}}$ stores the matrix $K_{ii_j}\}$
  repeat
    $n{+}{+}$
    Draw random subset M from $[m]\backslash I$
    {**Select best basis function**}
    Compute $\langle k_i, k_j\rangle = K_{ij}$ where $i \in [m]$ and $j \in M$
    Compute $\langle k_i, \tilde{k}_j\rangle = K^{\text{c}} T^M$ where T^M is an $n \times |M|$ submatrix of T.
    for all $i \in M$ **do**
      $\texttt{Improvement}_i = \frac{\sum_{j=1}^m (K_{ji} - (K^{\text{c}}T^M)_{ji})^2}{K_{ii} - (K^{\text{c}}T^M)_{ii}}$
    end for
    $i_n = \hat{\imath} = \operatorname{argmax}_{i \in M} \texttt{Improvement}_i$
    {**Project out** $k_{\hat{\imath}}$}
    $t = (K_{\hat{\imath}\hat{\imath}} - (K^{\text{c}}T^M)_{\hat{\imath}\hat{\imath}})^{-1}(-T_{\hat{\imath}1}, \ldots, -T_{\hat{\imath}n}, 1)$
    $\texttt{dot} = \langle k_1, k_{\hat{\imath}} - \tilde{k}_{\hat{\imath}}\rangle$ {use K^{c} and $K^{\text{c}}T^M$}
    $T = T + \texttt{dot}^\top t$ {Rank 1 update}
    $K^{\text{c}} = (K^{\text{c}}, K_{\hat{\imath}})$
    {**Bound the Residuals**}
    $\epsilon_n = \sum_{i=1}^m K_{ii} - \sum_{i=1}^m \sum_{j=1}^n K^{\text{c}}_{ij} T_{ij}$
  until $\epsilon_n < \epsilon$
output: n, T, I, ϵ_n

5.3 Random Subset Selection

The first thing to check is whether the reasoning of Section 2.2 applies to real world datasets. We used the USPS and the `Abalone` dataset to investigate this issue. In both cases we used $2\sigma^2 = 0.5d$, where d is the dimensionality of the data.

[2]This is a very conservative setting: the chosen parameters (σ, Λ) may not be optimal for the other algorithms presented.

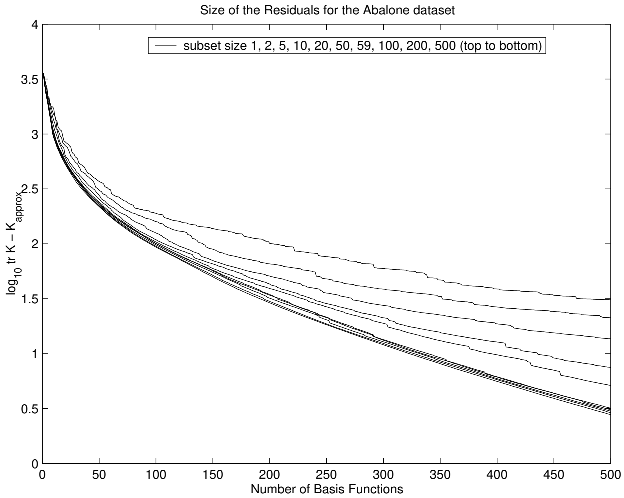

Figure 1. Size of the Residuals $\mathrm{tr}(K - \tilde{K})$ for the **Abalone** dataset. From top to bottom: subsets of size 1, 2, 5, 10, 20, 50, 59, 100, 200. Note that for subsets of size 50 and more, no noticeable difference in performance can be observed.

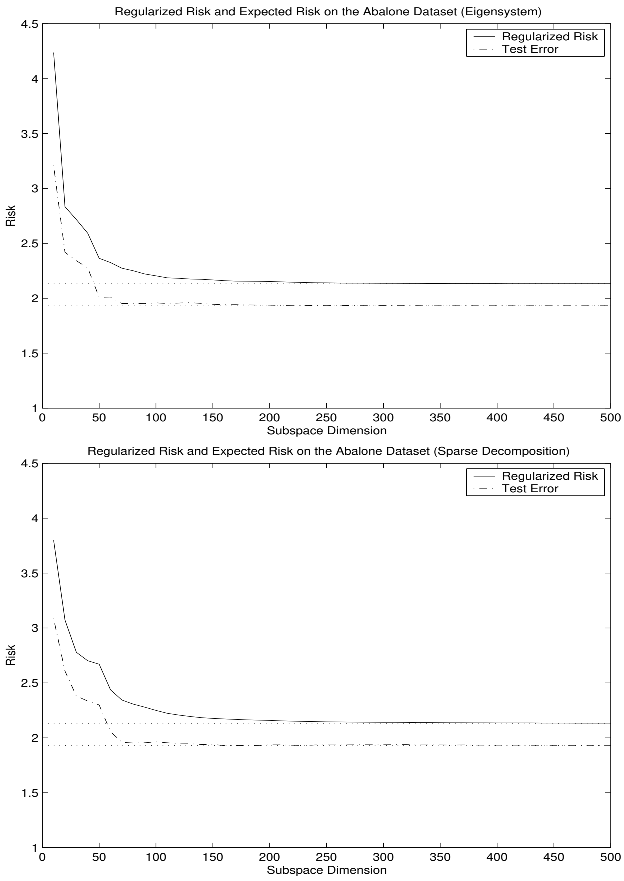

Figure 2. Top: optimal reduced system generated by principal component analysis (the bottom line is the value obtained by solving the full regularized risk functional); reduced system generated by sparse decomposition (again, baseline is the full matrix K).

Figure 1 shows that a subset of size 59 is large enough to yield near optimal performance. The same results were obtained on the USPS postal dataset.

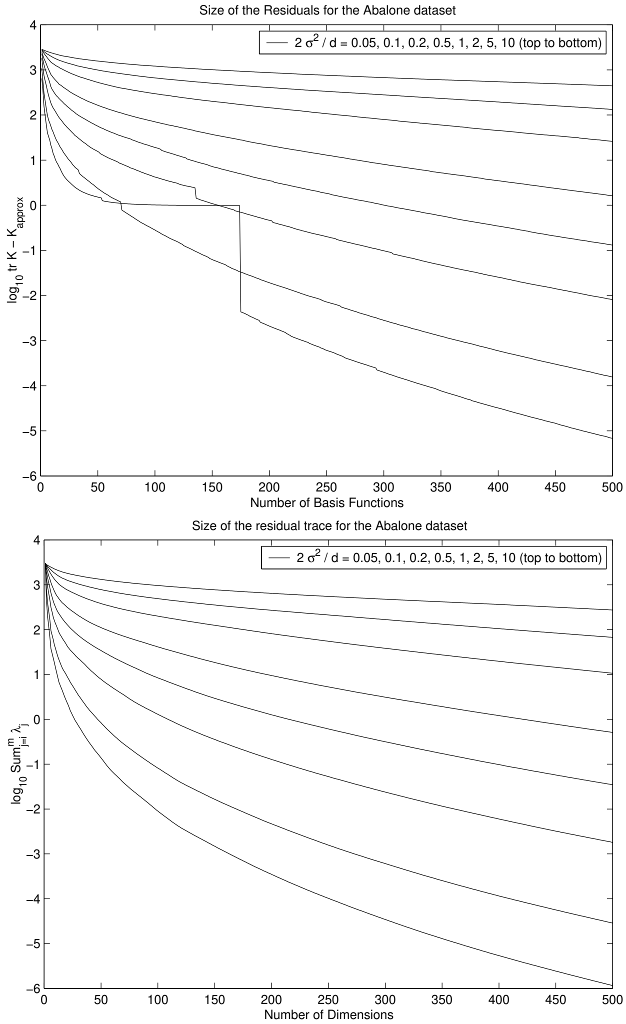

Figure 3. Size of the Residuals $\mathrm{tr}(K - \tilde{K})$ for the **Abalone** datset. Top: projection on basis functions given by the greedy approximation scheme. Bottom: projection on subspaces given by principal component analysis.

5.4 Regularized Risk and Generalization

We compare the following three algorithms: a plain standard regularization network where α is obtained according to (6), solutions of reduced dimensionality where the reduction is obtained by PCA, and a sparse expansion obtained by minimizing the residuals in function space. Since Λ and σ were chosen to be optimal for regularization the key point is to check how many basis functions are needed to obtain similar performance.

Figures 2 and 5 compare the performance obtained by

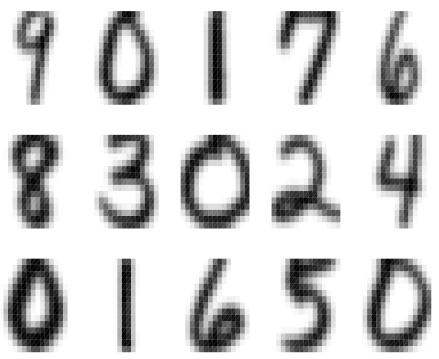

Figure 4. Left to Right, Top to Bottom. Patterns corresponding to the first basis functions. Note that 9 out of 10 digits are chosen among the first 10 patterns and that all patterns are sufficiently different (or shifted).

using lower dimensional subspaces generated by PCA or sparse greedy approximation. One can see that their difference in terms of performance is only marginal. The crucial advantage of sparse greedy approximation, however, is, that it is much faster ($O(nm)$ rather than $O(m^2)$ per iteration), leads to sparse decompositions (PCA may speed up training but has no effect on testing speed since it yields *dense* decompositions), and requires $O(nm)$ rather than $O(m^2)$ memory.

Even more striking — the qualitative behaviour of the generalization performance is very similar (see Figure 5). This can be understood by considering Figure 3: the subspaces selected by the two methods are probably very similar since the trace of the residual matrix also exhibits almost identical behaviour.

Hence, sparse decompositions can easily account for a speedup of between 4 and 10 with no penalty in terms of generalization performance (and possibly even more dramatic improvements on larger datasets) but much reduced classification time. The size of the residuals indicates when to stop (three oders of magnitude in trace reduction proved to be sufficient in all cases).

6. Applications

While the approximation scheme in Hilbert spaces certainly could be applied to solving linear systems of equations $Ax = y$ directly, it would be unreasonable to do so, since in the latter case we do not want to approximate a *matrix* but rather just a *single vector*. Good approximations of the matrix guarantee good approximations of a vector, however, *better* approximations with less iterations can be obtained by an application of a matching pursuit technique directly in Hilbert space [Schölkopf et al., 1999]. Furthermore, subspace methods such as [Skilling, 1988] also provide

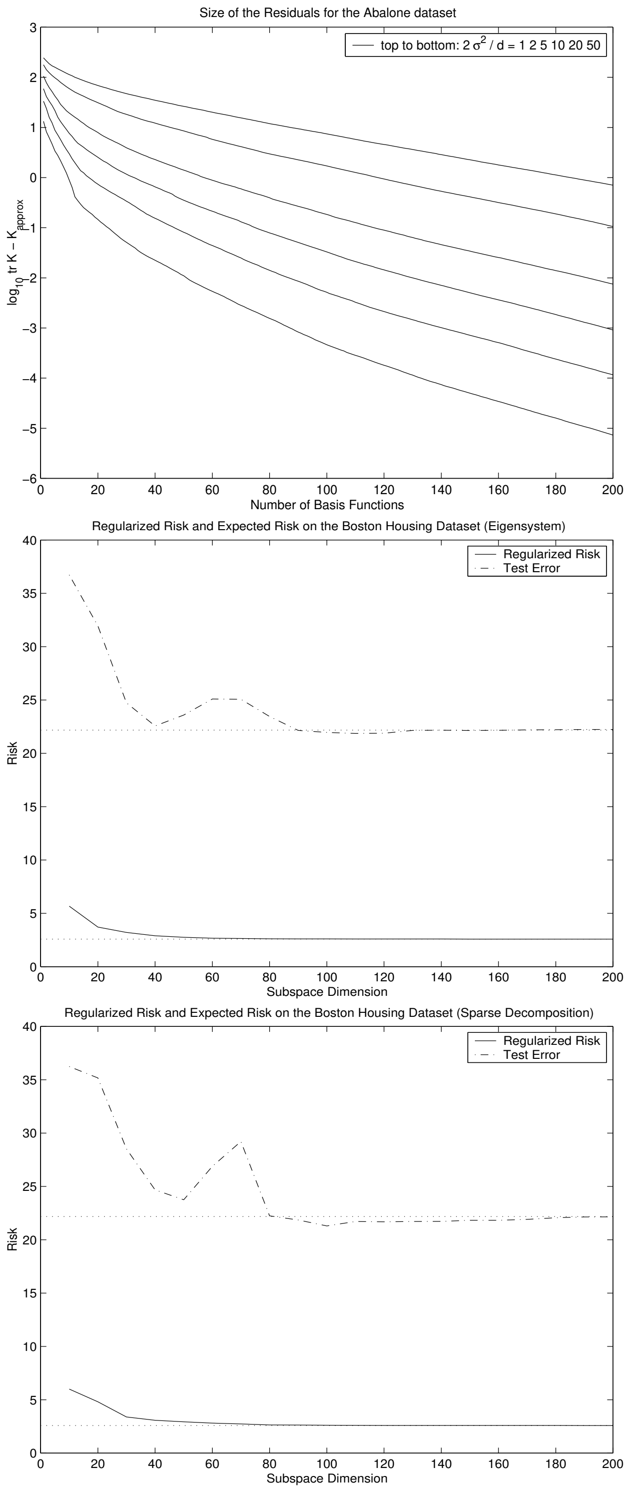

Figure 5. Top to bottom: Size of the Residuals $\mathrm{tr}(K - \tilde{K})$ for the `Boston Housing` datset; optimal reduced system generated by principal component analysis (the bottom line is the value obtained by solving the full regularized risk functional); reduced system generated by sparse decomposition. Note the similar increase in training error for both algorithms (again, baseline is the full matrix K).

a fast means of approximate solutions of linear systems. But there exist cases where the knowledge of an approximation of the design matrix may be useful.

6.1 Support Vector Machines

Interior Point Codes are used frequently to solve the quadratic and linear programs of Support Vector Machines. While being very precise, they do not scale well to large problems since they require that the complete design matrix be stored in memory and inverted. Unfortunately, it is not exactly K but $K_{\mathrm{KKT}} := K + \mathrm{diag}\{\sigma_1, \dots \sigma_m\}$ that has to be inverted (the so-called reduced KKT system Vanderbei [1994]). This happens several times during optimization.

While K itself in general has rapidly decaying eigenvalues [Williamson et al., 1998], the actual matrix K_{KKT} does not exhibit this property. However, if we consider the approximation $\tilde{K}_{\mathrm{KKT}} := \tilde{K} + \mathrm{diag}\{\sigma_1, \dots \sigma_m\}$, the latter can be inverted by $O(nm^2)$ operations. This is so, since the inversion of the diagonal part is $O(m)$, and $\tilde{K}$ can be seen as a rank n matrix. Furthermore, rank n updates of the inverse of an $m \times m$ matrix are $O(nm^2)$.[3] This will be the topic of further research.

6.2 Linear Programming Machines

A modification of the regularization term in the regularized risk functional leads to linear programming machines [Mangasarian, 1965]. While these have a sparsity term already included by penalizing the 1-norm of the expansion coefficients, it can be numerically very demanding to solve the optimization problem exactly.

Again, by choosing a subset of basis functions to start with (which has approximately the same expressive power as the overall set of basis functions) we can find a solution that is almost as good as the general solution but much cheaper to compute.

Modern mathematical programming codes such as CPLEX have a so-valled *Hotstart* feature which enables them to solve larger problems efficiently, given a solution on a subset of variables. This suggests the following algorithm: find a subset of basis functions approximating K, solve the reduced subproblem; if the approximation quality is sufficiently high in comparison to the size of the residuals $K - \tilde{K}$ (e.g. large margin in classification) stop, otherwise generate more basis functions and continue.

[3] Thanks to Chris Williams for suggesting this approach.

6.3 Random Projections

Random Projection techniques can be used to create (local) hashing functions for neighbourhood and search queries in high dimensional spaces (cf. e.g. [Gionis et al., 1999]). They work by randomly projecting on a lower dimensional space and replacing the high dimensional neighbourhood query by a search operation that can be computed more easily. In general these methods operate directly in the *Euclidean* space given by data.

Defining a proximity measure on discrete data (e.g. webpages and other text documents or gene sequences) by a Euclidean metric will have problems in reflecting the specific structure of the data. Metrics based on generative models (see [Jaakkola and Haussler, 1999] and subsequent work) can provide a much more adequate representation. Yet, a full neighbourhood search in these spaces is out of question due to the computational cost.

Sparse matrix approximations, however, are able to recover most of the structure of the space by projecting on just a few basis functions (e.g. a 99% approximation on the `Abalone` dataset involves just 200 out of 3000 kernels). We expect the effects to be even more dramatic as the sample size increases. Consequently sparse matrix approximation methods are a good candidate for fast and cheap search criteria in high-dimensional spaces.

6.4 Further Applications

Results such as [Makovoz, 1996] show that under certain conditions sparse aproximations of matrices exist. Morever, optimality results for sparse greedy approximation algorithms [Natarajan, 1995] give good bounds on the number of terms required by such non-optimal approximation schemes. Combining both statements may show that the algorithms presented in this paper are actually close to optimal. This is the subject of further research.

Another application of our methods are preconditioners, used in numerical mathematics, to transform matrices before e.g. inverting them. The core part is to find an easy to compute approximation to the matrix under consideration. Since the present techniques can be applied to any positive semidefinite matrix regardless whether it was generated by a kernel function or not (such an equivalent representation always exists), we hope that our approximation schemes will be useful in this area, too.

7. Relations to other Algorithms

7.1 Sparse Kernel Feature Analysis

This algorithm [Smola et al., 1999] is very similar in its structure to the techniques presented in the current paper. Its main differences are the additional normalization step in the ℓ_1 norm, carried out on the basis functions after projection and the fact that either the column selection criterion (Section 4.3) or a Projection Pursuit-type criterion is chosen in order to pick the best basis functions.

7.2 Clustering

A popular method for finding a subset of functions k_i in the special case of RBF-kernels is clustering (see e.g. [Schölkopf et al., 1997]): the cluster centers obtained from a dataset are used to generate the basis functions. Since $e^{-\|x-x'\|^2}$ is monotonic in $\|x - x'\|$, approximations of the latter (i.e. $\|\hat{x} - x'\|$ rather than $\|x-x'\|$ for all x in the Voronoi region of $\hat{x}$) one can see that such an algorithm has the side effect of generating an approximating matrix $\tilde{K}$. However, this is not the immediate objective of the algorithm, convergence on large datasets may be problematic, and the number of cluster centers has to be specified beforehand. Finally, for all kernels other than RBFs, the technique is not applicable at all.

8. Discussion

We presented a general scheme to approximate symmetric positive semidefinite matrices by subsets of columns or basis functions. The function space algorithm, in particular, has the advantage of being $O(nm)$ per iteration which is significantly lower than any algorithm operating in column space directly (they all require $O(m^2)$ complexity per iteration).

First experiments show that the algorithm is both fast and has very good approximation characteristics which will help overcome some of the limitations of current kernel methods. We believe that it is the starting point for an exciting new class of optimization procedures and approximation methods, some of which were already pointed out in Section 6.

Acknowledgements This research was supported by a grant of the Deutsche Forschungsgesellschaft (Sm 62/1-1) and the Australian Research Council. The authors thank Peter Bartlett, Markus Hegland, Olvi Mangasarian, and Bob Williamson for helpful discussions.

References

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

C. Blake, E. Keogh, and C. J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *Siam Journal of Scientific Computing*, 20(1):33–61, 1999.

D. Cox and F. O'Sullivan. Asymptotic analysis of penalized likelihood and related estimators. *The Annals of Statistics*, 18:1676–1695, 1990.

A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 1999.

F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1455–1480, 1998.

F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269, 1995.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, Cambridge, MA, 1999. MIT Press.

G. S. Kimeldorf and G. Wahba. A correspondence between Bayesan estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 2: 495–502, 1971.

Y. Makovoz. Random approximants and neural networks. *Journal of Approximation Theory*, 85:98–109, 1996.

S. Mallat and Z. Zhang. Matching Pursuit in a time-frequency dictionary. *IEEE Transactions on Signal Processing*, 41:3397–3415, 1993.

O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

B.K̃. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal of Computing*, 24(2): 227–234, 1995.

B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings, First International Conference on Knowledge Discovery & Data Mining*, Menlo Park, 1995. AAAI Press.

B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, September 1999.

B. Schölkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Trans. Sign. Processing*, 45:2758 – 2765, 1997.

J. Skilling. *Maximum Entropy and Bayesian Methods.* Cambridge University Press, 1988.

A. J. Smola. *Learning with Kernels.* PhD thesis, Technische Universität Berlin, 1998.

A. J. Smola, O. L. Mangasarian, and B. Schölkopf. Sparse kernel feature analysis. Technical Report 99-03, University of Wisconsin, Data Mining Institute, Madison, 1999.

R. J. Vanderbei. LOQO: An interior point code for quadratic programming. TR SOR-94-15, Statistics and Operations Research, Princeton Univ., NJ, 1994.

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, N.Y., 1995. ISBN 0-387-94559-8.

V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 281–287, Cambridge, MA, 1997. MIT Press.

C. K. I. Williams. Prediction with gaussian processes: From linear regression to linear prediction and beyond. *Learning and Inference in Graphical Models*, 1998.

R. C. Williamson, A. J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. NeuroCOLT NC-TR-98-019, Royal Holloway College, 1998.