An Introduction to Kernel Methods*

Colin Campbell,
Department of Engineering Mathematics,
Bristol University, Bristol BS8 1TR,
United Kingdom

Abstract

Kernel methods give a systematic and principled approach to training learning machines and the good generalisation performance achieved can be readily justified using statistical learning theory or Bayesian arguments. We describe how to use kernel methods for classification, regression and novelty detection and in each case we find that training can be reduced to optimisation of a convex cost function. We describe algorithmic approaches for training these systems including model selection strategies and techniques for handling unlabelled data. Finally we present some recent applications. The emphasis will be on using RBF kernels which generate RBF networks but the approach is general since other types of learning machines (e.g. feed-forward neural networks or polynomial classifiers) can be readily generated with different choices of kernel.

1 Introduction.

Radial Basis Function (RBF) networks have been widely studied because they exhibit good generalisation and universal approximation through use of RBF nodes in the hidden layer. In this Chapter we will outline a new approach to designing RBF networks based on *kernel methods*. These techniques have a number of advantages. As we shall see, the approach is systematic and properly motivated theoretically. The learning machine is also explicitly constructed using the most informative patterns in the data. Because the dependence on the data is clear it is much easier to explain and interpret the model and data cleaning [16] could be implemented to improve performance. The learning process involves optimisation of a cost function which is provably convex. This contrasts with neural network approaches where the exist of false local minima in the error function

*To appear in *Radial Basis Function Networks: Design and Applications*. R.J. Howlett and L.C. Jain (eds). Springer Verlag, Berlin, 2000

can complicate the learning process. For kernel methods there are comparatively few parameters required for tuning the system. Indeed, recently proposed model selection schemes could lead to the elimination of these parameters altogether. Unlike neural network approaches the architecture is determined by the algorithm and not found by experimentation. It is also possible to give confidence estimates on the classification accuracy on new test examples. Finally these learning machines typically exhibit good generalisation and perform well in practice.

In this introduction to the subject we will focus on Support Vector Machines (SVMs) which are the most well known learning systems based on kernel methods. The emphasis will be on classification, regression and novelty detection and we will not cover other interesting topics, for example, kernel methods for unsupervised learning [43, 52]. We will begin by introducing SVMs for binary classification and the idea of kernel substitution. The kernel representation of data amounts to a nonlinear projection of data into a high-dimensional space where it is easier to separate the two classes of data. We then develop this approach to handle noisy datasets, multiclass classification, regression and novelty detection. We also consider strategies for finding the kernel parameter and techniques for handling unlabelled data. In section 3 we then describe algorithms for training these systems and in section 4 we describe some current applications. In the conclusion we will briefly discuss other types of learning machines based on kernel methods.

2 Kernel Methods for Classification, Regression and Novelty Detection.

2.1 Binary Classification.

From the perspective of statistical learning theory the motivation for considering binary classifier SVMs comes from theoretical bounds on the generalisation error [58, 59, 10]. For ease of explanation we give the theorem in the Appendix 1 (Theorem 1) and simply note here that it has two important features. Firstly, the error bound is minimised by maximising the *margin,* γ, i.e. the minimal distance between the hyperplane separating the two classes and the closest datapoints to the hyperplane (Figure 1). Secondly, the upper bound on the generalization error do not depend on the dimension of the space.

The Learning Task. Let us consider a binary classification task with datapoints $\mathbf{x}_i$ $(i = 1, \ldots, m)$ having corresponding labels $y_i = \pm 1$ and let the decision function be:

$$f(\mathbf{x}) = \text{sign}\left(\mathbf{w} \cdot \mathbf{x} + b\right) \tag{1}$$

If the dataset is separable then the data will be correctly classified if $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) > 0$ $\forall i$. Clearly this relation is invariant under a positive rescaling of the argument inside the *sign*-function, hence we can define a *canonical hyperplane* such that $\mathbf{w} \cdot \mathbf{x} + b = 1$

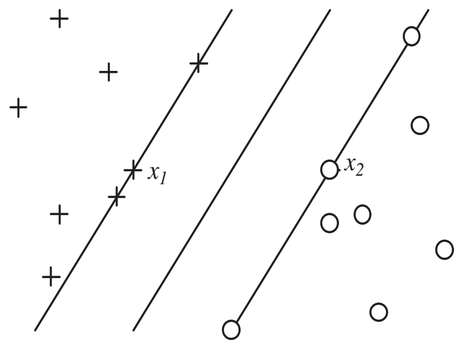

Figure 1: The *margin* is the perpendicular distance between the separating hyperplane and a hyperplane through the closest points (these are *support vectors*). The region between the hyperplanes on each side is called the *margin band.* $\mathbf{x}_1$ and $\mathbf{x}_2$ are examples of support vectors of opposite sign.

for the closest points on one side and $\mathbf{w} \cdot \mathbf{x} + b = -1$ for the closest on the other. For the separating hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ the normal vector is clearly $\mathbf{w} / ||\mathbf{w}||$. Hence the margin is given by the projection of $\mathbf{x}_1 - \mathbf{x}_2$ onto this vector where $\mathbf{x}_1$ and $\mathbf{x}_2$ are closest points on opposite sides of the separating hyperplane (see Figure 1). Since $\mathbf{w} \cdot \mathbf{x}_1 + b = 1$ and $\mathbf{w} \cdot \mathbf{x}_2 + b = -1$ this means the margin is $\gamma = 1/ ||\mathbf{w}||$. To maximise the margin the task is therefore:

$$\min \left[\frac{1}{2} ||\mathbf{w}||^2 \right] \tag{2}$$

subject to the constraints:

$$y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b \right) \geq 1 \qquad \forall i \tag{3}$$

and the learning task reduces to minimisation of the primal objective function :

$$L = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) - \sum_{i=1}^{m} \alpha_i \left(y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \right) \tag{4}$$

where α_i are Lagrange multipliers (hence $\alpha_i \geq 0$). Taking the derivatives with respect to b and $\mathbf{w}$ gives:

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{5}$$

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \tag{6}$$

and resubstituting these expressions back in the primal gives the Wolfe dual:

$$W(\alpha)=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j\left(\mathbf{x}_i\cdot\mathbf{x}_j\right) \tag{7}$$

which must be maximised with respect to the α_i subject to the constraint:

$$\alpha_i\geq 0 \qquad \sum_{i=1}^{m}\alpha_i y_i=0 \tag{8}$$

Kernel substitution. This constrained quadratic programming (QP) problem will give an optimal separating hyperplane with a maximal margin if the data is separable. However, we have still not exploited the second observation from theorem 1: the error bound does not depend on the dimension of the space. This feature enables us to give an alternative kernel representation of the data which is equivalent to a mapping into a high dimensional space where the two classes of data are more readily separable. This space is called *feature space* and must be a pre-Hilbert or inner product space. For the dual objective function in (7) we notice that the datapoints, $\mathbf{x}_i$, only appear inside an inner product. Thus the mapping is achieved through a replacement of the inner product:

$$\mathbf{x}_i\cdot\mathbf{x}_j\rightarrow\phi\left(\mathbf{x}_i\right)\cdot\phi(\mathbf{x}_j) \tag{9}$$

The functional form of the mapping $\phi(\mathbf{x}_i)$ does not need to be known since it is implicitly defined by the choice of *kernel:*

$$K(\mathbf{x}_i,\mathbf{x}_j)=\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j) \tag{10}$$

which is the inner product in the higher dimensional Hilbert space. With a suitable choice of kernel the data can become separable in feature space despite being non-separable in the original input space. Thus, for example, whereas data for n-parity is non-separable by a hyperplane in input space it can be separated in the feature space defined by RBF kernels:

$$K(\mathbf{x}_i,\mathbf{x}_j)=e^{-||\mathbf{x}_i-\mathbf{x}_j||^2/2\sigma^2} \tag{11}$$

Other choices of kernel are possible e.g.:

$$K(\mathbf{x}_i,\mathbf{x}_j)=(\mathbf{x}_i\cdot\mathbf{x}_j+1)^d \qquad\qquad K(\mathbf{x}_i,\mathbf{x}_j)=\tanh(\beta\mathbf{x}_i\cdot\mathbf{x}_j+b) \tag{12}$$

which would define polynomial and feedforward neural network classifiers. Each choice of kernel will define a different type of feature space and the resulting classifiers will perform differently on test data, though good generalisation should be assured from Theorem 1. For an SVM with RBF kernels the resulting architecture is an RBF network. However, the method for determining the number of nodes and their centers is quite different from

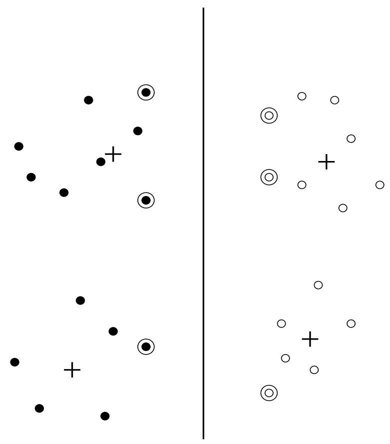

Figure 2: A classical RBF network finds the centers of RBF nodes by k-means clustering (marked by crosses). In contrast an SVM with RBF kernels uses RBF nodes centered on the support vectors (circled) i.e. the datapoints closest to the separating hyperplane (the vertical line illustrated).

standard RBF networks with the number of nodes equal to the number of support vectors and the centers of the RBF nodes identified with the support vectors themselves (Figure 2).

Feasible kernels implicitly describing this mapping must satisfy *Mercer's conditions* described in more detail in Appendix 2. The class of mathematical objects which can be used as kernels is very general and includes, for example, scores produced by dynamic alignment algorithms [18, 63] and a wide range of functions.

For the given choice of kernel the learning task therefore involves maximisation of the objective function :

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{13}$$

subject to the constraints (8). The associated *Karush-Kuhn-Tucker* (KKT) conditions are:

$$\begin{aligned} y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b \right) - 1 &\geq 0 \qquad \forall i \\ \alpha_i &\geq 0 \qquad \forall i \\ \alpha_i \left(y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \right) &= 0 \qquad \forall i \end{aligned} \tag{14}$$

which are always satisfied when a solution is found. Test examples are evaluated using a decision function given by the sign of:

$$f(\mathbf{z}) = \sum_{i=1}^{m} y_i \alpha_i K(\mathbf{x}_i, \mathbf{z}) + b \tag{15}$$

Since the bias, b, does not feature in the above dual formulation it is found from the primal constraints:

$$b = -\frac{1}{2}\left[\max_{\{i|y_i=-1\}}\left(\sum_{j\in\{\mathrm{SV}\}}^{m} y_j\alpha_j K(\mathbf{x}_i,\mathbf{x}_j)\right) + \min_{\{i|y_i=+1\}}\left(\sum_{j\in\{\mathrm{SV}\}}^{m} y_j\alpha_j K(\mathbf{x}_i,\mathbf{x}_j)\right)\right] \tag{16}$$

using the optimal values of α_j. When the maximal margin hyperplane is found in feature space, only those points which lie closest to the hyperplane have $\alpha_i > 0$ and these points are the *support vectors* (all other points have $\alpha_i = 0$). This means that the representation of the hypothesis is given solely by those points which are closest to the hyperplane and *they are the most informative patterns in the data.* Patterns which are not support vectors do not influence the position and orientation of the separating hyperplane and so do not contribute to the hypothesis (Figure 1).

We have motivated SVMs using statistical learning theory but they can also be understood from a Bayesian perspective [51, 25, 26]. Bayesian [53] and statistical learning theory can also be used to define confidence measures for classification. From the latter we find that the confidence of a classification is directly related to the magnitude of $f(\mathbf{z})$ on a test example [46].

2.2 Multiclass Classification.

Many real-life datasets involve multiclass classification and various schemes have been proposed to handle this [28]. One approach is to generalise the binary classifier to an $n-$class classifier with weights and biases $(\mathbf{w}_k, b_k), k = 1, \ldots, n$ for each class and a decision function [64]:

$$f(\mathbf{z}) = \arg\max_{1\leq k\leq n} (\mathbf{w}_k \cdot \mathbf{z} + b_k) \tag{17}$$

However, this type of classifier has a similar level of performance to the simpler scheme of n binary classifiers each of which performs one-against-all classification. Binary classifiers also can be incorporated into a directed acyclic graph (Figure 3) so that multiclass classification is decomposed to binary classification at each node in the tree [34].

2.3 Allowing for Training Errors: Soft Margin Techniques.

Most real life datasets contain noise and an SVM can fit to this noise leading to poor generalisation. The effect of outliers and noise can be reduced by introducing a *soft margin* [8] and two schemes are currently used. In the first (L_1 error norm) the learning task is the same as in (13,8) except for the introduction of the box constraint:

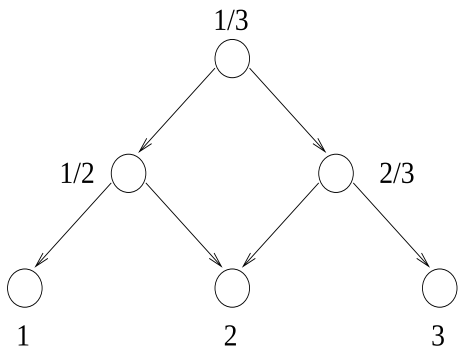

Figure 3: A multi-class classification problem can be reduced to a series of binary classification tasks using a tree structure with a binary decision at each node.

$$0 \leq \alpha_i \leq C \tag{18}$$

while in the second (L_2 error norm) the learning task is (13,8) except for addition of a small positive constant to the leading diagonal of the kernel matrix [8, 48]:

$$K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \lambda \tag{19}$$

C and λ control the trade-off between training error and generalisation ability and are chosen by means of a validation set. The effect of these soft margins is illustrated in Figure 4 for the ionosphere dataset from the UCI Repository [57].

The justification for these approaches comes from statistical learning theory (cf. Theorems 2 and 3 in Appendix 1). Thus for the L_1 error norm (and prior to introducing kernels) condition (3) is relaxed by introducing a positive slack variable ξ_i:

$$y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1 - \xi_i \tag{20}$$

and the task is now to minimise the sum of errors $\sum_{i=1}^{m} \xi_i$ in addition to $||\mathbf{w}||^2$:

$$\min \left[\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{m} \xi_i \right] \tag{21}$$

This is readily formulated as a primal objective function :

$$\begin{aligned} L(\mathbf{w}, b, \alpha, \xi) &= \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{m} \xi_i & (22) \\ &\quad - \sum_{i=1}^{m} \alpha_i \left[y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{m} r_i \xi_i & (23) \end{aligned}$$

with Lagrange multipliers $\alpha_i \geq 0$ and $r_i \geq 0$. The derivatives with respect to $\mathbf{w}$, b and ξ give:

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i = 0 & (24) \\
\frac{\partial L}{\partial b} &= \sum_{i=1}^{m} \alpha_i y_i = 0 & (25) \\
\frac{\partial L}{\partial \xi_i} &= C - \alpha_i - r_i = 0 & (26)
\end{aligned}$$

Resubstituting these back in the primal objective function we obtain the same dual objective function, (13), as before. However, $r_i > 0$ and $C - \alpha_i - r_i = 0$, hence $\alpha_i \leq C$ and the constraint $0 \leq \alpha_i$ is replaced by $0 \leq \alpha_i \leq C$. Patterns with values $0 < \alpha_i < C$ will be referred to later as *non-bound* and those with $\alpha_i = 0$ or $\alpha_i = C$ will be said to be *at bound*. For an L_1 error norm we find the bias in the decision function (15) by using the final KKT condition in (14). Thus if i is a *non-bound* pattern it follows that $b = y_i - \sum_j \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j)$ assuming $y_i = \pm 1$.
The optimal value of C must be found by experimentation using a validation set (Figure 4) and it cannot be readily related to the characteristics of the dataset or model. In an alternative approach (ν-SVM [44]) the soft margin parameter, ν, can be interpreted as an upper bound on the fraction of training errors and a lower bound on the fraction of patterns which are support vectors.
For the L_2 error norm the primal objective function is:

$$\begin{aligned}
L(\mathbf{w}, b, \alpha, \xi) &= \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{m} \xi_i^2 \\
&\quad - \sum_{i=1}^{m} \alpha_i \left[y_i \left(\mathbf{w}\cdot\mathbf{x}_i + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{m} r_i \xi_i & (27)
\end{aligned}$$

with $\alpha_i \geq 0$ and $r_i \geq 0$. After obtaining the derivatives with respect to $\mathbf{w}$, b and ξ, substituting for $\mathbf{w}$ and ξ in the primal objective function and noting that the dual objective function is maximal when $r_i = 0$, we obtain the following dual objective function after kernel substitution:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{4C}\sum_{i=1}^{m} \alpha_i^2 \qquad (28)$$

With $\lambda = 1/2C$ this gives the same dual objective function as before (13) but with the substitution (19). For many real-life datasets there is an imbalance between the amount

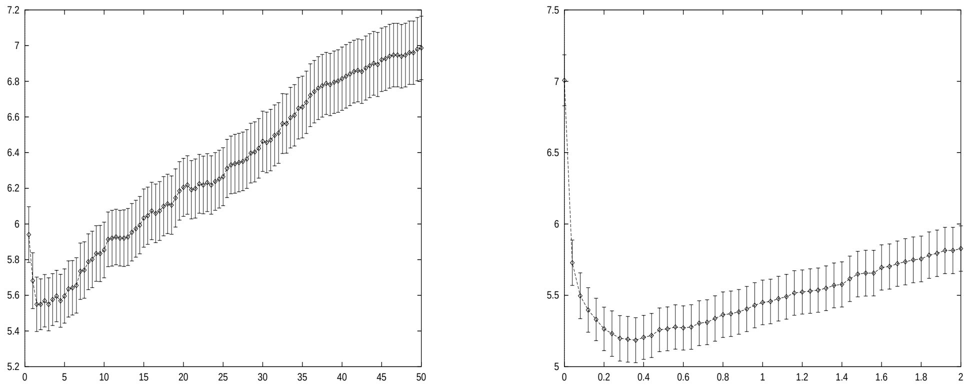

Figure 4: *Left:* Generalisation error as a percentage (y-axis) versus C (x-axis) and *right:* generalisation error as a percentage (y-axis) versus λ (x-axis) for soft margin classifiers based on L_1 and L_2 error norms respectively. The UCI ionosphere dataset was used with RBF kernels ($\sigma = 1.5$) and 100 samplings of the data.

of data in different classes, or the significance of the data in the two classes can be quite different. For example, for the detection of tumours on MRI scans it may be best to allow a higher number of false positives if this improved the true positive detection rate. The relative balance between the detection rate for different classes can be easily shifted [61] by introducing asymmetric soft margin parameters. Thus for binary classification with an L_1 error norm $0 \leq \alpha_i \leq C_+$ ($y_i = +1$), and $0 \leq \alpha_i \leq C_-$ ($y_i = -1$), while $K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \lambda_+$ (if $y_i = +1$) and $K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \lambda_-$ (if $y_i = -1$) for the L_2 error norm.

2.4 Novelty Detection.

For many real-world problems the task is not to classify but to detect novel or abnormal instances. For the above example involving classification of tumours it could be the case that the classification system does not correctly detect a tumour with a rare shape which is distinct from all members of the training set. On the other hand, a novelty detector would still potentially highlight the object as abnormal. Novelty detection has potential applications in many problem domains such as condition monitoring or medical diagnosis. Novelty detection can be viewed as modelling the *support* of a data distribution (rather than having to find a real-valued function for estimating the density of the data itself). Thus, at its simplest level, the objective is to create a binary-valued function which is positive in those regions of input space where the data predominantly lies and negative elsewhere.
One approach [54] is to find a hypersphere with a minimal radius R and centre $\mathbf{a}$ which contains most of the data: novel test points lie outside the boundary of this hypersphere.

The technique we now outline was originally suggested by Vapnik [58, 5], intepreted as a novelty detector by Tax and Duin [54] and used by the latter authors for real life applications [55]. The effect of outliers is reduced by using slack variables ξ_i to allow for datapoints outside the sphere and the task is to minimise the volume of the sphere and number of datapoints outside i.e.

$$\min \left[R^2 + \frac{1}{m\nu} \sum_i \xi_i \right]$$

subject to the constraints:

$$(\mathbf{x}_i - \mathbf{a})^T (\mathbf{x}_i - \mathbf{a}) \leq R^2 + \xi_i$$

and $\xi_i \geq 0$, and where ν controls the tradeoff between the two terms. The primal objective function is then:

$$L(R, \mathbf{a}, \alpha_i, \xi_i) = R^2 + \frac{1}{m\nu} \sum_{i=1}^{m} \xi_i - \sum_{i=1}^{m} \alpha_i \left(R^2 + \xi_i - (\mathbf{x}_i \cdot \mathbf{x}_i - 2\mathbf{a} \cdot \mathbf{x}_i + \mathbf{a} \cdot \mathbf{a}) \right) - \sum_{i=1}^{m} \gamma_i \xi_i$$

with $\alpha_i \geq 0$ and $\gamma_i \geq 0$. After kernel substitution the dual formulation amounts to maximisation of:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{29}$$

with respect to α_i and subject to $\sum_{i=1}^{m} \alpha_i = 1$ and $0 \leq \alpha_i \leq 1/m\nu$. If $m\nu > 1$ then *at bound* examples will occur with $\alpha_i = 1/m\nu$ and these correspond to outliers in the training process. Having completed the training process a test point $\mathbf{z}$ is declared novel if:

$$K(\mathbf{z}, \mathbf{z}) - 2 \sum_{i=1}^{m} \alpha_i K(\mathbf{z}, \mathbf{x}_i) + \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - R^2 \geq 0 \tag{30}$$

where R^2 is first computed by finding an example which is *non-bound* and setting this inequality to an equality.
An alternative approach has been developed by Schölkopf et al. [41]. Suppose we restricted our attention to RBF kernels: in this case the data lie in a region on the surface of a hypersphere in feature space since $\phi(\mathbf{x}) \cdot \phi(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}) = 1$ from (11). The objective is therefore to separate off this region from the surface region containing no data. This is achieved by constructing a hyperplane which is maximally distant from the origin with all datapoints lying on the opposite side from the origin and such that $\mathbf{w} \cdot \mathbf{x}_i + b \geq 0$.

This construction can be extended to allow for outliers by introducing a slack variable ξ_i giving rise to the following criterion:

$$\min \left[\frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{m\nu} \sum_{i=1}^{m} \xi_i + b \right] \tag{31}$$

subject to:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq -\xi_i \tag{32}$$

with $\xi_i \geq 0$. The primal objective function is therefore:

$$\begin{aligned} L(\mathbf{w}, \xi, b, \alpha, \beta) &= \frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{m\nu} \sum_{i=1}^{m} \xi_i + b \\ &\quad - \sum_{i=1}^{m} \alpha_i (\mathbf{w} \cdot \mathbf{x}_i + b + \xi_i) - \sum_{i=1}^{m} \beta_i \xi_i \end{aligned} \tag{33}$$

and the derivatives:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{m} \alpha_i \mathbf{x}_i = 0 \tag{34}$$

$$\frac{\partial L}{\partial \xi} = -\alpha_i - \beta_i + \frac{1}{m\nu} = 0 \tag{35}$$

$$\frac{\partial L}{\partial b} = 1 - \sum_{i=1}^{m} \alpha_i = 0 \tag{36}$$

Since $\alpha_i, \beta_i \geq 0$ the derivative $\partial L / \partial \xi = 0$ implies $0 \leq \alpha_i \leq 1/m\nu$. After kernel substitution the dual formulation involves minimisation of:

$$W(\alpha) = \frac{1}{2} \sum_{i,k=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{37}$$

subject to:

$$0 \leq \alpha_i \leq \frac{1}{m\nu} \qquad \sum_{i=1}^{m} \alpha_i = 1 \tag{38}$$

To determine the bias we find an example, k say, which is non-bound (α_i and β_i are nonzero and $0 < \alpha_i < 1/m\nu$) and determine b from:

$$b = -\sum_{j=1}^{m} \alpha_j K(\mathbf{x}_j, \mathbf{x}_k) \tag{39}$$

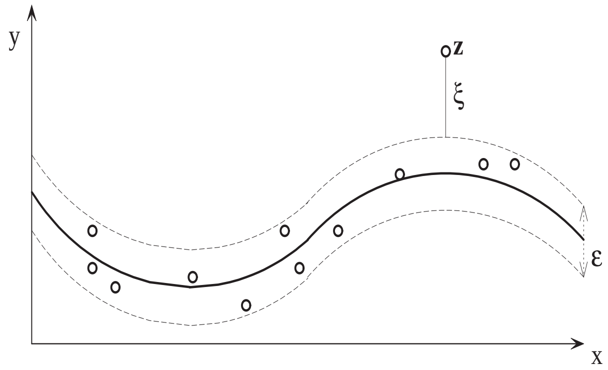

Figure 5: The ϵ-insensitive band around a nonlinear regression function. The variables ξ measure the cost of training errors corresponding to points outside the band e.g. $\mathbf{z}$.

The support of the distribution is then modelled by the decision function:

$$f(\mathbf{z}) = \text{sign}\left(\sum_{j=1}^{m} \alpha_j K(\mathbf{x}_j, \mathbf{z}) + b\right) \tag{40}$$

In the above models the parameter ν has a neat interpretation as an upper bound on the fraction of outliers and a lower bound of the fraction of patterns which are support vectors [41]. Schölkopf et al.[41] provide good experimental evidence in favour of this approach including the highlighting of abnormal digits in the USPS handwritten character dataset. The method also works well for other types of kernel. This and the earlier scheme for novelty detection can also be used with an L_2 error norm in which case the constraint $0 \leq \alpha_i \leq 1/m\nu$ is removed and an addition to the kernel diagonal (19) used instead.

2.5 Regression.

For real-valued outputs the learning task can also be theoretically motivated from statistical learning theory. Theorem 4 in Appendix 1 gives a bound on the generalisation error to within a margin tolerance θ. We can visualise this as a band or tube of size $\pm(\theta - \gamma)$ around the hypothesis function $f(\mathbf{x})$ and any points outside this tube can be viewed as training errors (Figure 5).
Thus instead of (3) we now use constraints $y_i - \mathbf{w} \cdot \mathbf{x}_i - b \leq \epsilon$ and $\mathbf{w} \cdot \mathbf{x}_i + b - y_i \leq \epsilon$ to allow for a deviation ϵ between the eventual targets y_i and the function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$, modelling the data. As before we would also minimise $||\mathbf{w}||^2$ to increase flatness or penalise overcomplexity. To account for training errors we introduce slack variables $\xi_i, \widehat{\xi}_i$ for the two types of training error and an $\epsilon-$*insensitive loss function* (Figure 6). These slack variables are zero for points inside the tube and progressively increase for points

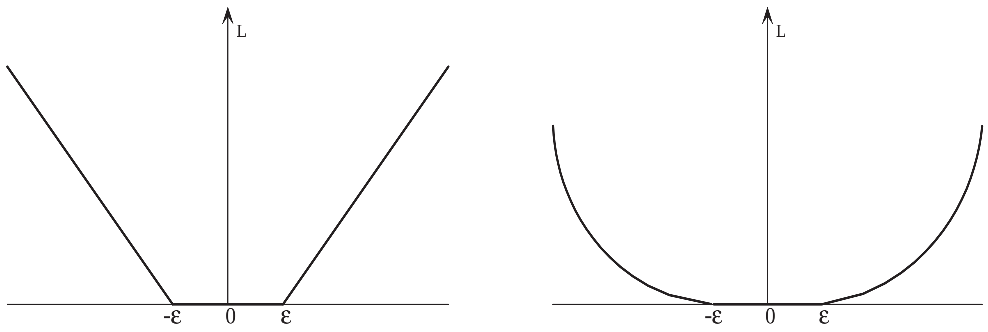

Figure 6: Left figure: a linear ϵ-insensitive loss function versus $y_i - \mathbf{w} \cdot \mathbf{x}_i - b$. Right figure: a quadratic ϵ-insensitive loss function.

outside the tube according to the loss function used. This general approach is called ϵ-SV regression [58] and is the most common approach to SV regression, though not the only one [9, 59]. For a *linear* $\epsilon-$*insensitive loss function* the task is therefore to minimise:

$$\min \left[||\mathbf{w}||^2 + C \sum_{i=1}^{m} \left(\xi_i + \widehat{\xi}_i \right) \right] \tag{41}$$

subject to

$$\begin{aligned} y_i - \mathbf{w} \cdot \mathbf{x}_i - b &\leq \epsilon + \xi_i \\ (\mathbf{w} \cdot \mathbf{x}_i + b) - y_i &\leq \epsilon + \widehat{\xi}_i \end{aligned} \tag{42}$$

where the slack variables are both positive $\xi_i, \widehat{\xi}_i \geq 0$. After kernel substitution the dual objective function is:

$$W(\alpha, \widehat{\alpha}) = \sum_{i=1}^{m} y_i (\alpha_i - \widehat{\alpha}_i) - \epsilon \sum_{i=1}^{m} (\alpha_i + \widehat{\alpha}_i) - \frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i - \widehat{\alpha}_i)(\alpha_j - \widehat{\alpha}_j) K(x_i, x_j) \tag{43}$$

which is maximised subject to

$$\sum_{i=1}^{m} \widehat{\alpha}_i = \sum_{i=1}^{m} \alpha_i \tag{44}$$

and:

$$0 \leq \alpha_i \leq C \qquad 0 \leq \widehat{\alpha}_i \leq C \tag{45}$$

Similarly a *quadratic* $\epsilon-$*insensitive loss function* gives rise to:

$$\min \left[||\mathbf{w}||^2 + C \sum_{i=1}^{m} \left(\xi_i^2 + \widehat{\xi}_i^2 \right) \right] \tag{46}$$

subject to (42), giving a dual objective function :

$$W(\alpha, \widehat{\alpha}) = \sum_{i=1}^{m} y_i(\alpha_i - \widehat{\alpha}_i) - \epsilon \sum_{i=1}^{m} (\alpha_i + \widehat{\alpha}_i) - \frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i - \widehat{\alpha}_i)(\alpha_j - \widehat{\alpha}_j) \left(K(\mathbf{x}_i, \mathbf{x}_j) + \delta_{ij}/C \right) \tag{47}$$

which is maximised subject to (44). The decision function is then:

$$f(\mathbf{z}) = \sum_{i=1}^{m} y_i(\alpha_i - \widehat{\alpha}_i) K(\mathbf{x}_i, \mathbf{z}) + b \tag{48}$$

We still have to compute the bias, b, and we do so by considering the KKT conditions for regression. For a linear loss function prior to kernel substitution these are:

$$\begin{aligned} \alpha_i \left(\epsilon + \xi_i - y_i + \mathbf{w} \cdot \mathbf{x}_i + b \right) &= 0 \\ \widehat{\alpha}_i \left(\epsilon + \widehat{\xi}_i + y_i - \mathbf{w} \cdot \mathbf{x}_i - b \right) &= 0 \end{aligned} \tag{49}$$

where $\mathbf{w} = \sum_{j=1}^{m} y_j(\alpha_j - \widehat{\alpha}_j)\mathbf{x}_j$, and:

$$\begin{aligned} (C - \alpha_i)\, \xi_i &= 0 \\ (C - \widehat{\alpha}_i)\, \widehat{\xi}_i &= 0 \end{aligned} \tag{50}$$

From the latter conditions we see that only when $\alpha_i = C$ or $\widehat{\alpha}_i = C$ are the slack variables non-zero: these examples correspond to points outside the ϵ-insensitive tube. Hence from (49) we can find the bias from a non-bound example with $0 < \alpha_i < C$ using $b = y_i - \mathbf{w} \cdot \mathbf{x}_i - \epsilon$ and similarly for $0 < \widehat{\alpha}_i < C$ we can obtain it from $b = y_i - \mathbf{w} \cdot \mathbf{x}_i + \epsilon$. Though the bias can be obtained from one such example it is best to compute it using an average over all points on the margin.

From the KKT conditions we also deduce that $\alpha_i \widehat{\alpha}_i = 0$ since α_i and $\widehat{\alpha}_i$ cannot be simultaneously non-zero because we would have non-zero slack variables on both sides of the band. Thus given that α_i is zero if $\widehat{\alpha}_i > 0$ and *vice versa* we can use a more convenient formulation for the actually optimisation task, e.g. maximise:

$$W(\gamma) = -\epsilon \sum_{i=1}^{m} |\gamma_i| + \sum_{i=1}^{m} y_i \gamma_i - \frac{1}{2} \sum_{i,j=1}^{m} \gamma_i \gamma_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{51}$$

subject to $\sum_{i=1}^{m} \gamma_i = 0$ for a linear ϵ-insensitive loss function.

Apart from the formulations given here it is possible to define other loss functions giving rise to different dual objective functions. In addition, rather than specifying ϵ *a priori* it is possible to specify an upper bound ν $(0 \leq \nu \leq 1)$ on the fraction of points lying outside the band and then find ϵ by optimising over the primal objective function:

$$\frac{1}{2}||\mathbf{w}||^2 + C\left(\nu m\epsilon + \sum_{i=1}^{m}|y_i - f(\mathbf{x}_i)|\right) \tag{52}$$

with ϵ acting as an additional parameter to minimise over [38].

2.6 Enhanced Learning Strategies.

Determining the kernel parameters. During the training process the kernel parameter (e.g. σ in (11)) needs to be specified. If it is too small, for example, then generalisation performance will suffer from overfitting (Figure 7). The kernel parameter is best found using cross-validation if sufficient data is available. However, recent model selection strategies can give a reasonable estimate for the kernel parameter based on theoretical arguments without use of validation data. As a first attempt, for the hard margin case, the generalisation error bound (which we denote here as E) can be approximated by $E \simeq R^2/m\gamma^2$ [47] where R is the radius of the smallest ball containing the training data. Let α_i^0 be the values of the Lagrange multipliers at the optimum of $W(\alpha)$. From $\gamma = 1/||\mathbf{w}||$ we can deduce that $\gamma^2 = 1/\sum_{i\in\{SV\}}\alpha_i^0$ since if i is a support vector then $y_i(\sum_{j\in SV}\alpha_j^o y_j(\mathbf{x}_i \cdot \mathbf{x}_j) + b) = 1$, thus:

$$\begin{aligned} \mathbf{w}\cdot\mathbf{w} &= \sum_{i,j\in\{SV\}}\alpha_i^o\alpha_j^o y_i y_j(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i\in\{SV\}}\alpha_i^o(1 - y_i b) \\ &= \sum_{i\in\{SV\}}\alpha_i^o \end{aligned} \tag{53}$$

since $\sum_{i\in\{SV\}}\alpha_i^o y_i = 0$. After kernel substitution RBF kernels give $R \simeq 1$ since the data lie on the surface of hypersphere. Hence an estimate for σ could be found by sequentially training SVMs on a dataset at successively larger values of σ, evaluating E from the α_i^0 for each case and choosing that value of σ for which E is minimised. This method [9] will give a reasonable estimate if the data is spread evenly over the surface of the hypersphere but it is poor if the data lie in a flat ellipsoid, for example, since the radius R would be influenced by the largest deviations.

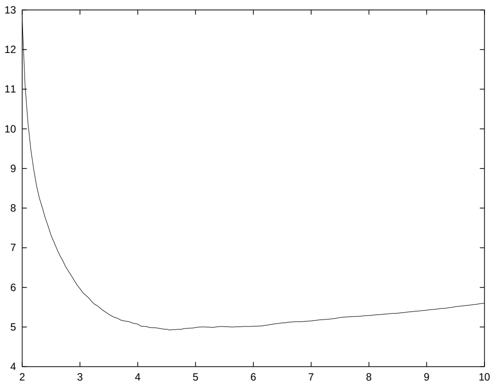

Figure 7: Generalisation error as a percentage (y-axis) versus σ (x-axis) from an experiment using artificial data from the mirror symmetry problem and a SVM with an RBF kernel (the curve has been smoothed using 100,000 generated datapoints, allowing repeats). After first overfitting the data the generalisation error passes through a minimum as σ increases.

More refined estimates therefore take into account the distribution of the data. One approach [7] to finding the error bound is to notionally rescale data in kernel space to compensate for uneven distributions. This rescaling is achieved using the eigenvalues and eigenvectors of the matrix $K(\mathbf{x}_i, \mathbf{x}_j)$. A more complex strategy along these lines has also been proposed by Schölkopf et al [42] which leads to an algorithm which has performed well in practice for a small number of datasets. A very efficient bound has also been derived recently by Herbrich et al [20].
The most economical way to use the training data is to use a *leave-one-out* cross-validation procedure. In this procedure, single elements from the data set are sequentially removed, the SVM is trained on the remaining $m - 1$ elements and then tested on the removed datapoint. Using the approximation that the set of support vectors does not change for removal of single patterns, it is possible to derive tight bounds on the generalisation error. Two examples of these model selection rules are the *span-rule* of Chapelle and Vapnik [7] and a rule proposed by Jaakkola and Haussler [22]. Based on recent studies with a limited number of datasets, these model selection strategies appear to work well. However, a comparative study of these different techniques and their application to a wider range of real-life datasets needs to be undertaken to establish if they are fully practical approaches.

Handling Unlabelled Data. For some real-life datasets the datapoints are initially unlabelled. Since the labels of points corresponding to *non-support* vectors are not actually required for determining an optimal separating hyperplane these points do not need to be labelled. This issue is particularly important for practical situations in which labelling data is expensive or the dataset is large and unlabelled. Since SVMs construct the hypothesis using a subset of the data containing the most informative patterns they are

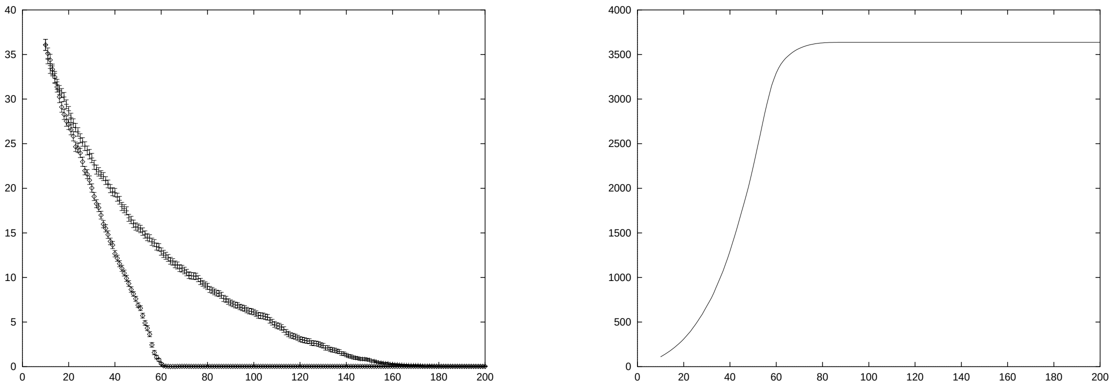

Figure 8: *Left figure*: Generalisation error (y-axis) as a percentage versus number of patterns (x-axis) for random selection (top curve) and selective sampling (bottom curve). *Right figure*: monitoring the value of the dual objective function provides a good stopping criterion for noiseless datasets. It this case the the majority rule for random bit strings was used with 100 samplings of the data each split into 200 training and 200 test examples.

good candidates for *active* or *selective sampling* techniques which would predominantly request the labels for those patterns which will become support vectors
During the process of active selection the information gained from an example depends both on the position (available information) and on its label (unavailable information before querying). Thus we must follow a heuristic strategy to maximise information gain at each step. Firstly we note that querying a point within the margin band (Figure 1) *always* guarantees a gain whatever the label of the point: we do not gain by querying a point outside the band unless the current hypothesis predicts the label incorrectly. The best points to query are indeed those points which are closest to the current hyperplane [6]. Intuitively this makes sense since these are most likely to be maximally ambiguous with respect to the current hypothesis and hence the best candidates for ensuring that the information received is maximised. Hence a good strategy [6] is to start by requesting the labels for a small initial set of data and then successively querying the labels of points closest to the current hyperplane. For noiseless datasets plateauing of the dual objective function provides a good stopping criterion (since learning non-support vectors would not change the value of $W(\alpha)$ - see Figure 8(right)), whereas for noisy datasets emptying of the margin band and a validation phase provide the best stopping criterion [6]. Active selection works best if the hypothesis modelling the data is *sparse* (i.e. there are comparatively few support vectors to be found by the query learning strategy) in which case good generalisation is achieved despite requesting only a subset of labels in the dataset (Figure 8).

3 Algorithmic Approaches to Training SVMs

For classification, regression or novelty detection we see that the learning task involves optimisation of a quadratic cost function and thus techniques from quadratic programming are most applicable including quasi-Newton, conjugate gradient and primal-dual interior point methods. Certain QP packages are readily applicable such as MINOS and LOQO. These methods can be used to train an SVM rapidly but they have the disadvantage that the kernel matrix is stored in memory. For small datasets this is practical and QP routines are the best choice, but for larger datasets alternative techniques have to be used. These split into two categories: techniques in which kernel components are evaluated and discarded during learning and *working set* methods in which an evolving subset of data is used. For the first category the most obvious approach is to sequentially update the α_i and this is the approach used by the Kernel Adatron (KA) algorithm [15]. For binary classification (with no soft margin or bias) this is a simple gradient ascent procedure on (13) in which $\alpha_i > 0$ initially and the α_i are subsequently sequentially updated using:

$$\alpha_i \leftarrow \beta_i \theta(\beta_i) \qquad \text{where} \qquad \beta_i = \alpha_i + \eta \left(1 - y_i \sum_{j=1}^{m} \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j) \right) \tag{54}$$

and $\theta(\beta)$ is the Heaviside step function. The optimal learning rate η can be readily evaluated: $\eta = 1/K(\mathbf{x}_i, \mathbf{x}_i)$ and a sufficient condition for convergence is $0 < \eta K(\mathbf{x}_i, \mathbf{x}_i) < 2$. With the given decision function (15) this method is very easy to implement and can give a quick impression of the performance of SVMs on classification tasks. It is equivalent to Hildreth's method in Optimisation theory and can be generalised to the case of soft margins and inclusion of a bias [27]. However, it is not as fast as most QP routines, especially on small datasets.

3.1 Chunking and Decomposition.

Rather than sequentially updating the α_i the alternative is to update the α_i in parallel but using only a subset or *chunk* of data at each stage. Thus a QP routine is used to optimise the objective function on an initial arbitrary subset of data. The support vectors found are retained and all other datapoints (with $\alpha_i = 0$) discarded. A new working set of data is then derived from these support vectors and additional datapoints which maximally violate the storage constraints. This *chunking* process is then iterated until the margin is maximised. Of course, this procedure may still fail because the dataset is too large or the hypothesis modelling the data is not sparse (most of the α_i are non-zero, say). In this case *decomposition* [31] methods provide a better approach: these algorithms only use a fixed size subset of data with the α_i for the remainder kept fixed.

3.2 Decomposition and Sequential Minimal Optimisation.

The limiting case of decomposition is the Sequential Minimial Optimisation (SMO) algorithm of Platt [33] in which only two α_i are optimised at each iteration. The smallest set of parameters which can be optimised with each iteration is plainly two if the constraint $\sum_{i=1}^{m} \alpha_i y_i = 0$ is to hold. Remarkably, if only two parameters are optimised and the rest kept fixed then it is possible to derive an analytical solution which can be executed using few numerical operations. The algorithm therefore selects two Lagrange multipliers to optimise at every step and separate heuristics are used to find the two members of the pair. Due to its decomposition of the learning task and speed it is probably the method of choice for training SVMs and hence we will describe it in detail here for the case of binary classification.

The outer loop. The heuristic for the first member of the pair provides the outer loop of the SMO algorithm. This loop iterates through the entire training set to determining if an example violates the KKT conditions and, if it does, to find if it is a candidate for optimisation. After an initial pass through the training set the outer loop does not subsequently iterate through the entire training set. Instead it iterates through those examples with Lagrange multipliers corresponding to non-bound examples (neither 0 nor C). Examples violating the KKT conditions are candidates for immediate optimisation and update. The outer loop makes repeated passes over the non-bound examples until all of the non-bound examples obey the KKT conditions. The outer loop then iterates over the entire training set again. The outer loop keeps alternating between single passes over the entire training set and multiple passes over the non-bound subset until the entire training set obeys the KKT conditions at which point the algorithm terminates.

The inner loop. During the pass through the outer loop let us suppose the algorithm finds an example which violates the KKT conditions (with an associated Lagrange multiplier we shall denote α_1 for convenience). To find the second member of the pair, α_2, we proceed to the inner loop. SMO selects the latter example to maximise the step-length taken during the joint 2-variable optimisation process outlined below. To achieve this SMO keeps a record of each value of $E_i = f(\mathbf{x}_i) - y_i$ (where $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$) for every non-bound example in the training set and then approximates the step-length by the absolute value of the numerator in equation (56) below i.e. $|E_1 - E_2|$. Since we want to maximise the step-length, this means we choose the minimum value of E_2 if E_1 is positive and the maximum value of E_2 if E_1 is negative. As we point out below, this step may not make an improvement. If so, SMO iterates through the non-bound examples searching for a second example that can make an improvement. If none of the non-bound examples gives an improvement, then SMO iterates through the entire training set until an example is found that makes an improvement. Both the iteration through the non-bound examples and the iteration through the entire training set are started at random locations to avoid an implicit bias towards examples at the beginning of the training set.

The update rules. Having described the outer and inner loops for SMO we now describe the rules used to update the chosen pair of Lagrange multipliers (α_1, α_2). The constraint $\sum_{i=1}^{N} \alpha_i y_i = 0$ gives:

$$\alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i \neq 1,2} \alpha_i y_i \tag{55}$$

Since $y_i = \pm 1$ we have two possibilities. Firstly $y_1 = y_2$ in which case $\alpha_1 + \alpha_2$ is equal to some constant: $\alpha_1 + \alpha_2 = \lambda$, say, or $y_1 \neq y_2$ in which case $\alpha_1 - \alpha_2 = \lambda$. The next step is to find the maximum of the dual objective function with only two Lagrange multipliers permitted to change. Usually this leads to a maximum along the direction of the linear equality constraint though this not always the case as we discuss shortly. We first determine the candidate value for second Lagrange multiplier α_2 and then the ends of the diagonal line segment in terms of α_2:

$$\alpha_2^{new} = \alpha_2^{old} - \frac{y_2(E_1 - E_2)}{\eta}, \tag{56}$$

where $E_i = f^{old}(\mathbf{x}_i) - y_i$ and:

$$\eta = 2K(\mathbf{x}_1, \mathbf{x}_2) - K(\mathbf{x}_1, \mathbf{x}_1) - K(\mathbf{x}_2, \mathbf{x}_2). \tag{57}$$

If noise is present and we use a L_1 soft margin then the next step is to determine the two ends of the diagonal line segment. Thus if $y_1 \neq y_2$ the following bounds apply :

$$L = \max(0, \alpha_2^{old} - \alpha_1^{old}), \qquad H = \min(C, C + \alpha_2^{old} - \alpha_1^{old}). \tag{58}$$

and if $y_1 = y_2$ then:

$$L = \max(0, \alpha_1^{old} + \alpha_2^{old} - C), \qquad H = \min(C, \alpha_2^{old} + \alpha_1^{old}). \tag{59}$$

The constrained maximum is then found by clipping the unconstrained maximum to the ends of the line segment:

$$\alpha_2^{new,clipped} = \begin{cases} H, & if\ \alpha_2^{new} \geq H; \\ \alpha_2^{new}, & if\ L < \alpha_2^{new} < H; \\ L, & if\ \alpha_2^{new} \leq L. \end{cases} \tag{60}$$

Next the value of $\alpha_1$is determined from the clipped α_2:

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new,clipped}). \tag{61}$$

This operation moves α_1 and α_2 to the end point with the highest value of $W(\alpha)$. Only when $W(\alpha)$ is the same at both ends will no improvement be made. After each step, the bias b is recomputed so that the KKT conditions are fulfilled for both examples. If the new α_1 is a non-bound variable then b_1 is determined from:

$$b_1 = b_1^{old} - E_1 - y_1(\alpha_1^{new} - \alpha_1^{old})K(\mathbf{x}_1, \mathbf{x}_1) - y_2(\alpha_2^{new,clipped} - \alpha_2^{old})K(\mathbf{x}_1, \mathbf{x}_2). \qquad (62)$$

Similarly if the new α_2 is non-bound then b_2 is determined from:

$$b_2 = b_2^{old} - E_2 - y_1(\alpha_1^{new} - \alpha_1^{old})K(\mathbf{x}_1, \mathbf{x}_2) - y_2(\alpha_2^{new,clipped} - \alpha_2^{old})K(\mathbf{x}_2, \mathbf{x}_2). \qquad (63)$$

If b_1 and b_2 are valid they should be equal. When both new Lagrange multipliers are at bound and if L is not equal to H, then all thresholds on the interval between b_1 and b_2 are consistent with the KKT conditions and we choose the threshold to be halfway in between b_1 and b_2.

The SMO algorithm has been refined to improve speed [24] and generalised to cover the above three tasks of classification [33], regression [49] and novelty detection [41].

4 Applications.

SVMs have been successfully applied to a number of applications ranging from particle identification [2], face detection, [32] and text categorisation [23, 13, 11] to engine knock detection [37], bioinformatics [4, 65] and database marketing [3]. In this section we discuss three successful application areas as illustrations: machine vision, handwritten character recognition and bioinformatics. This is a rapidly changing area so more contemporary accounts are best obtained from relevant websites (e.g. [17]).

Machine Vision. SVMs are very suited to the binary or multiclass classsification tasks which commonly arise in machine vision. As an example we consider a multiclass classification task involving face identification [14]. This experiment used the standard ORL dataset [30] consisting of 10 images per person from 40 different persons. Three methods were tried: a direct SVM classifier which learnt the original images directly (apart from some local rescaling), a classifier which used more extensive pre-processing involving rescaling, local sampling and local principal component analysis, and an invariant SVM classifier which learnt the original images plus a set of images which have been translated and zoomed. For the invariant SVM classifier the training set of 200 images (5 per person) was increased to 1400 translated and zoomed examples and an RBF kernel was used. On the test set these three methods gave generalisation errors of 5.5%, 3.7% and 1.5% respectively. This was compared with a number of alternative techniques [14] with the best result among the latter being 2.7%. Face and gender detection have also been successfully achieved. 3D object recognition [36] is another successful area of application including 3D face recognition, pedestrian recognition, etc.

Handwritten digit recognition. The United States Postal Service (USPS) dataset consists of 9298 handwritten digits each consisting of a 16×16 vector with entries between -1 and 1. An RBF network and a SVM were compared on this dataset. The RBF network had spherical Gaussian RBF nodes with the same number of Gaussian basis functions as there were support vectors for the SVM. The centroids and variances for the Gaussians were found using classical k-means clustering. For the SVM Gaussian kernels were used and the system was trained with a soft margin (with $C = 10.0$). A set of one-against-all classifiers were used since this is a multi-class problem. With a training set of 7291 the number of errors on the test set of 2007 was:

Digit	0	1	2	3	4	5	6	7	8	9
Classical RBF	20	16	43	38	46	31	15	18	37	26
SVM	16	8	25	19	29	23	14	12	25	16

and the SVM therefore outperformed the RBF network on all digits. SVMs have also been applied to the much larger NIST dataset of handwritten characters consisting of 60,000 training and 10,000 test images each with 400 pixels. SVMs with polynomial kernels perform at a comparable level to the best alternative techniques [59] with an 0.8% error on the test set.

Bioinformatics. Large-scale DNA sequencing projects are producing large volumes of data and there is a considerable demand for sophisticated methods for analysing biosequences. Bioinformatics presents a large number of important classification tasks such as prediction of protein secondary structure, classification of gene expression data, recognizing splice junctions i.e. the boundaries between exons and introns, etc. SVMs have been found to be very effective on these tasks. For example, SVMs outperformed four standard machine learning classifiers when applied to the functional classification of genes using gene expression data from DNA microarray hybridization experiments [4]. Several different similarity metrics and kernels were used and the best performance was achieved using an RBF kernel (the dataset was very imbalanced so asymmetric soft margin parameters were used). A second successful application has been protein homology detection to determine the structural and functional properties of new protein sequences [21]. Determination of these properties is achieved by relating new sequences to proteins with known structural features. In this application the SVM outperformed a number of established systems for homology detection for relating the test sequence to the correct families. As a third application we also mention the detection of translation initiation sites [65] (the points on nucleotide sequences where regions encoding proteins start). SVMs performed very well on this task using a kernel function specifically designed to include prior biological information.

5 Conclusion

Kernel methods have many appealing features. We have seen that they can be applied to a wide range of classification, regression and novelty detection tasks but they can also be applied to other areas we have not covered such as operator inversion and unsupervised learning. They can be used to generate many possible learning machine architectures (RBF networks, feedforward neural networks) through an appropriate choice of kernel. In particular the approach is properly motivated theoretically and systematic in execution. Our focus has been on SVMs but the concept of kernel substitution of the inner product is a powerful idea separate from margin maximisation and it can be used to define many other types of learning machines which can exhibit superior generalisation [19, 29] or which use few patterns to construct the hypothesis [56]. We have not been able to discuss these here but they also perform well and appear very promising. The excellent potential of this approach certainly suggests it will remain and develop as an important set of tools for machine learning.

Acknowledgements: The author would like to thank Nello Cristianini and Bernhard Schölkopf for comments on an earlier draft.

6 Appendices

6.1 Appendix 1: Generalisation Bounds

The generalisation bounds mentioned in section 2 are derived within the framework of probably approximately correct or *pac* learning. The principal assumption governing this approach is that the training and test data are independently and identically (iid) generated from a fixed distribution denoted $\mathcal{D}$. The distribution over input-output mappings will be denoted $(\mathbf{x}, y) \in X \times \{-1, 1\}$ and we will further assume that X is an inner product space. With these assumptions *pac*-learnability can be described as follows. Consider a class of possible target concepts C and a learner L using a hypothesis space H to try and learn this concept class. The class C is *pac*-learnable by L if for any target concept $c \in C$, L will with probability $(1-\delta)$ output a hypothesis $h \in H$ with an generalisation error $err_{\mathcal{D}}(h) < \epsilon(m, H, \delta)$ given a sufficient number, m, of training examples and computation time. The *pac* bound $\epsilon(m, H, \delta)$ is derived using probabilistic arguments [1, 62] and bounds the tail of the distribution of the generalisation error $err_{\mathcal{D}}(h)$.

For the case of a thresholding learner L with unit weight vector on an inner product space X and a margin $\gamma \in \Re^{+}$ the following theorem can be derived if the dataset is linearly separable:

Theorem 1: Suppose examples are drawn independently according to a distribution whose support is contained in a ball in $\Re^n$ centered at the origin, of radius R. If we succeed in correctly classifying m such examples by a canonical hyperplane, then with confidence $1-\delta$ the generalization error will be bounded from above by [47]:

$$\epsilon(m, H, \delta) = \frac{2}{m}\left(\frac{64R^2}{\gamma^2}\log\left(\frac{\gamma e m}{8R^2}\right)\log\left(\frac{32m}{\gamma^2}\right) + \log\left(\frac{4}{\delta}\right)\right) \tag{64}$$

provided $64R^2/\gamma^2 < m$. This result is not dependent on the dimensionality of the space and also states that the bound is reduced by maximising the margin γ. Though this is our main result motivating maximisation of the margin for SVMs it does not handle the case of non-separable data or the existence of noise. As pointed out in the main text these instances are handled by introducing an L_1 or L_2 soft margin. The following two bounds do not depend on the training data being linearly separable and cover these two cases [48]:

Theorem 2: Suppose examples are drawn independently according to a distribution whose support is contained in a ball in $\Re^n$ centered at the origin, of radius R. There is a constant c such that with confidence $1-\delta$ the generalization error will be bounded from above by:

$$\epsilon(m, H, \delta) = \frac{c}{m}\left(\frac{R^2 + ||\xi||_1^2 \log(1/\gamma)}{\gamma^2}\log^2(m) + \log\left(\frac{1}{\delta}\right)\right) \tag{65}$$

where ξ is the margin slack vector.

Theorem 3: Suppose examples are drawn independently according to a distribution whose support is contained in a ball in $\Re^n$ centered at the origin, of radius R. There is a constant c such that with confidence $1-\delta$ the generalization error will be bounded from above by:

$$\epsilon(m, H, \delta) = \frac{c}{m}\left(\frac{R^2 + ||\xi||_2^2}{\gamma^2}\log^2(m) + \log\left(\frac{1}{\delta}\right)\right) \tag{66}$$

where ξ is the margin slack vector.

For both these theorems we see that maximising the margin alone does not necessarily reduce the bound and it is necessary to additionally reduce the norms of the slack variables.

Both these theorems can be adapted to the case of regression. However, in constrast to Theorems 1-3 above it is no longer appropriate to fix the norm of the weight vector since invariance under positive rescaling of the weight vector only holds for a thresholding decision function. For regression the relevant theorem for an L_2 norm on the slack variables is then:

Theorem 4: Suppose examples are drawn independently according to a distribution whose support is contained in a ball in $\Re^n$ centered at the origin, of radius R. Furthermore fix $\gamma \leq \theta$ where θ is a positive real number. There is a constant c such that with probability $1-\delta$ over m random examples, the probability that a hypothesis with weight vector $\mathbf{w}$ has output more than θ away from its true value is bounded above by:

$$\epsilon(m, H, \delta) = \frac{c}{m}\left(\frac{||\mathbf{w}||_2^2 R^2 + ||\xi||_2^2}{\gamma^2}\log^2(m) + \log\left(\frac{1}{\delta}\right)\right) \tag{67}$$

where $\xi = \xi(\mathbf{w}, \theta, \gamma)$ is the margin slack vector. This theorem motivates the loss functions used in Section 2.5 on regression.

Finally, we note that the above classification theorems have also been extended to estimation of the support of a distribution [41]. However, current bounds are not good indicators of the probability of occurence of novel points outside a distribution and hence we do not quote them here for this reason.

6.2 Appendix 2: Kernel Substitution and Mercer's Theorem.

In section 2 we introduced the idea of kernel substitution, equivalent to introducing an implicit mapping of the data into a high-dimensional feature space. By this means non-linear datasets which are unlearnable by a linear learning machine in input space become learnable in feature space. In input space the hypothesis modelling the data is of the form:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \tag{68}$$

For binary classification, for example, we saw in section 2.3 that the weight vector $\mathbf{w}$ can be written as:

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \tag{69}$$

If the dataset is separable, the separating hyperplane passes through the convex hull defined by the datapoints and hence it is apparent that $\mathbf{w}$ can be expressed as such an expansion in terms of the datapoints. With this expansion the decision funtion (68) can therefore be rewritten:

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x}_j) + b \tag{70}$$

For the learning task (8,13) and this decision function the datapoints only appear in the form of inner products, justifying kernel substitution and with the choice of kernel implicitly selecting a particular feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \tag{71}$$

This raises the issue of which types of kernel function are allowed. The requirements on the kernel function are defined by the two theorems below. First we observe that the kernel function is symmetric. In addition we also note from that for a real vector $\mathbf{v}$ we have

$$\mathbf{v}^T\mathbf{K}\mathbf{v} = \left\|\sum_{i=1}^{m} v_i\phi(\mathbf{x}_i)\right\|_2^2 \geq 0 \tag{72}$$

where the matrix $\mathbf{K}$ has components $K(\mathbf{x}_i, \mathbf{x}_j)$, $(i = 1, \ldots, m; j = 1, \ldots, m)$. This suggests the following theorem which can be proved:

Theorem 5: Let $K(\mathbf{x}, \mathbf{y})$ be a real symmetric function on a finite input space, then it is a kernel function if and only if the matrix $\mathbf{K}$ with components $K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semi-definite.

More generally, for C a compact subset of $\Re^N$ we have:

Theorem 6 (Mercer's theorem): If $K(\mathbf{x}, \mathbf{y})$ is a continuous symmetric kernel of a positive integral operator T i.e.

$$(Tf)(\mathbf{y}) = \int_C K(\mathbf{x}, \mathbf{y})f(\mathbf{x})d\mathbf{x} \tag{73}$$

with:

$$\int_{C\times C} K(\mathbf{x}, \mathbf{y})f(\mathbf{x})f(\mathbf{y})d\mathbf{x}d\mathbf{y} \geq 0 \tag{74}$$

for all $f \in L_2(C)$ then it can be expanded in a uniformly convergent series in the eigenfunctions ψ_j and positive eigenvalues λ_j of T, thus:

$$K(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{n_e} \lambda_j \psi_j(\mathbf{x})\psi_j(\mathbf{y}) \tag{75}$$

where n_e is the number of positive eigenvalues.

This theorem holds for general compact spaces, and generalises the requirement to infinite feature spaces. (74) generalises the semi-positivity condition for finite spaces given in theorem 5. The expansion in (75) is a generalisation of the usual concept of an inner product in Hilbert space with each dimension rescaled by $\sqrt{\lambda_j}$.

References

[1] M. Anthony and P. Barlett. *Learning in neural networks: theoretical foundations.* Cambridge University Press, 1999.

[2] N. Barabino, M. Pallavicini, A Petrolini, M. Pontil and A. Verri. Support vector machines vs multi-layer perceptrons in particle identification. In *Proceedings of the European Symposium on Artifical Neural Networks '99* (D-Facto Press, Belgium), p. 257-262, 1999.

[3] K.P. Bennett, D. Wu and L. Auslender. On support vector decision trees for database marketing. Research Report No. 98-100, Rensselaer Polytechnic Institute, Troy, NY, 1998.

[4] M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, M. Ares Jr. and D. Haussler. Support vector machine classification of microarray gene expression data. University of California, Santa Cruz, Technical Report UCSC-CRL-99-09.

[5] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2, p. 121-167, 1998.

[6] C. Campbell, N. Cristianini and A. Smola. Instance selection using support vector machines, submitted to *Machine Learning,* 2000.

[7] O. Chapelle and V. Vapnik. Model selection for support vector machines, to appear in *Advances in Neural Information Processing Systems* 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

[8] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning* 20, p. 273-297, 1995.

[9] N. Cristianini, C. Campbell and J. Shawe-Taylor. Dynamically adapting kernels in support vector machines, *Advances in Neural Information Processing Systems* 11, ed. M. Kearns, S. A. Solla, and D. Cohn, MIT Press, p. 204-210, 1999.

[10] N. Cristainini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel-based Learning Methods.* Cambridge University Press, to appear January 2000.

[11] H. Drucker, with D. Wu and V. Vapnik. Support vector machines for spam categorization. *IEEE Trans. on Neural Networks*, 10, p. 1048-1054. 1999.

[12] H. Drucker, C. Burges, L. Kaufman, A. Smola and V. Vapnik. Support vector regression machines. In: M. Mozer, M. Jordan, and T. Petsche (eds.). *Advances in Neural Information Processing Systems*, 9, MIT Press, Cambridge, MA, 1997.

[13] S. Dumais, J. Platt, D. Heckerman and M. Sahami. Inductive Learning Algorithms and Representations for Text Categorization. *7th International Conference on Information and Knowledge Management,* 1998.

[14] R. Fernandez and E. Viennet. Face identification using support vector machines. *Proceedings of the European Symposium on Artificial Neural Networks* (ESANN99), (D.-Facto Press, Brussels) p.195-200, 1999

[15] T.-T. Friess, N. Cristianini and C. Campbell. The kernel adatron algorithm: a fast and simple learning procedure for support vector machines. *15th Intl. Conf. Machine Learning*, Morgan Kaufman Publishers, p. 188-196, 1998.

[16] I. Guyon, N. Matic and V. Vapnik. Discovering informative patterns and data cleaning. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, MIT Press, p. 181–203, 1996.

[17] Cf: http://www.clopinet.com/isabelle/Projects/SVM/applist.html

[18] D. Haussler. Convolution Kernels on Discrete Structures, UC Santa Cruz Technical Report UCS-CRL-99-10, 1999.

[19] R. Herbrich, T. Graepel and C. Campbell. Bayesian learning in reproducing kernel Hilbert spaces, submitted to *Machine Learning,* 1999.

[20] R. Herbrich, T. Graeppel and P. Bollmann-Sdorra. A PAC-Bayesian study of linear classifiers: why SVMs work. Preprint under preparation, Computer Science Department, TU, Berlin.

[21] T. Jaakkola, M. Diekhans and D. Haussler. A discriminative framework for detecting remote protein homologies. MIT Preprint, 1999.

[22] T. Jaakkola and D. Haussler. Probabilistic kernel regression models, in *Proceedings of the 1999 Conference on AI and Statistics,* 1999.

[23] T. Joachims. Text categorization with support vector machines: learning with many relevant features. *Proc. European Conference on Machine Learning* (ECML), 1998.

[24] S. Keerthi, S. Shevade, C. Bhattacharyya and K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. Tech Report, Dept. of CSA, Banglore, India, 1999.

[25] J. Kwok. Moderating the outputs of support vector machine classifiers. *IEEE Transactions on Neural Networks*, 10 p. 1018-1031, 1999.

[26] J. Kwok. Integrating the Evidence Framework and Support Vector Machines, *Proceedings of the European Symposium on Artificial Neural Networks (ESANN99)*, (D.-Facto Press, Brssels), p. 177-182, 1999.

[27] D. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, 1984.

[28] E. Mayoraz and E. Alpaydin. Support Vector Machines for Multiclass Classification, *Proceedings of the International Workshop on Artifical Neural Networks (IWANN99)*, IDIAP Technical Report 98-06, 1999.

[29] S. Mika, G. Ratsch, J. Weston, B. Schölkopf and K.-R. Muller. Fisher Discriminant Analysis with Kernels. *Proceedings of IEEE Neural Networks for Signal Processing Workshop* 1999, 8 pages, 1999.

[30] ORL dataset: Olivetti Research Laboratory, 1994, http://www.orl.co.uk/facedatabase.html.

[31] E. Osuna and F. Girosi. Reducing the Run-time Complexity in Support Vector Machines, in B. Schölkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning,* MIT press, Cambridge, MA, p. 271-284, 1999.

[32] E. Osuna, R. Freund and F. Girosi.Training Support Vector Machines: an Application to Face Detection. *Proceedings of CVPR'97*, Puerto Rico, 1997.

[33] J. Platt. Fast training of SVMs using sequential minimal optimisation. In B. Schölkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, p. 185-208, 1999.

[34] J.Platt, N. Cristianini and J. Shawe-Taylor. Large Margin DAGS for Multiclass Classification. In *Advances in Neural Information Processing Systems,* 12 ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

[35] C. Papageorgiou, M. Oren and T. Poggio. A General Framework for Object Detection. *Proceedings of International Conference on Computer Vision*, p. 555-562, 1998.

[36] D. Roobaert. Improving the Generalisation of Linear Support Vector Machines: an Application to 3D Object Recognition with Cluttered Background. *Proc. Workshop on Support Vector Machines at the 16th International Joint Conference on Artificial Intelligence*, July 31-August 6, Stockholm, Sweden, p. 29-33 1999.

[37] M.Rychetsky, S.Ortmann and M.Glesner. Support Vector Approaches for Engine Knock Detection. *Proc. International Joint Conference on Neural Networks (IJCNN 99)*, July, 1999, Washington, USA

[38] B. Schölkopf, P. Bartlett, A. Smola and R. Williamson. Support vector regression with automatic accuracy control. In L. Niklasson, M. Bóden and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, Berlin, Springer Verlag, 1998.

[39] B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson. Shrinking the Tube: A New Support Vector Regression Algorithm. To appear in: M. S. Kearns, S. A. Solla, and D. A. Cohn (eds.), *Advances in Neural Information Processing Systems*, 11, MIT Press, Cambridge, MA, 1999.

[40] B. Schölkopf, C. Burges and A. Smola. *Advances in Kernel Methods: Support Vector Machines.* MIT Press, Cambridge, MA. 1998.

[41] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson. Estimating the support of a high-dimensional distribution. Microsoft Research Corporation Technical Report MSR-TR-99-87, 1999.

[42] B. Schölkopf, J. Shawe-Taylor, A. Smola and R. Williamson. Kernel-dependent support vector error bounds. *Ninth International Conference on Artificial Neural Networks*, IEE Conference Publications No. 470, p. 304 - 309, 1999.

[43] B. Schölkopf, A. Smola, and K.–R. Muller. Kernel principal component analysis. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning.* MIT Press, Cambridge, MA, 1999b. 327 – 352.

[44] B. Schölkopf, A. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. To appear in *Neural Computation*, 1999.

[45] 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 Transactions on Signal Processing*, 45, p. 2758-2765, 1997.

[46] J. Shawe-Taylor. Confidence estimates of classification accuracy on new examples. In S. Ben-David (ed.), EuroCOLT97, *Lecture Notes in Artificial Intelligence*, 1208, p.260-271, 1997.

[47] J. Shawe-Taylor, P.L. Bartlett, R.C. Williamson and M. Anthony. Structural risk minimisation over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44 p. 1926-1940, 1998.

[48] J. Shawe-Taylor and N. Cristianini. Margin distribution and soft margin. In A. Smola, P. Barlett, B. Schölkopf and C. Schuurmans (eds), *Advances in Large Margin Classifiers*, Chapter 2, MIT Press, 1999.

[49] A. Smola and B. Schölkopf. A tutorial on support vector regression. NeuroColt2 TR 1998-03, 1998.

[50] A. Smola and B. Schölkopf, B. From Regularization Operators to Support Vector Kernels. In: M. Mozer, M. Jordan, and T. Petsche (eds). *Advances in Neural Information Processing Systems*, 9, MIT Press, Cambridge, MA, 1997.

[51] A. Smola, B. Schölkopf and K.-R. Müller. The connection between regularisation operators and support vector kernels. *Neural Networks*, 11 p. 637-649, 1998.

[52] A. Smola, R. C. Williamson, S. Mika, and B. Schölkopf. Regularized principal manifolds. In Computational Learning Theory: 4th European Conference, volume 1572 of *Lecture Notes in Artificial Intelligence* (Springer), p. 214-229, 1999.

[53] P. Sollich. Probabilistic methods for Support Vector Machines. In *Advances in Neural Information Processing Systems, 12.* S. Solla, T. Leen nd K.-R. Muller (eds) MIT Press, Cambridge, MA, 2000 (to appear).

[54] D. Tax and R. Duin. Data domain description by Support Vectors. In *Proceedings of ESANN99*, ed. M Verleysen, D. Facto Press, Brussels, p. 251-256, 1999.

[55] D. Tax, A. Ypma, and R. Duin. Support vector data description applied to machine vibration analysis. In: M. Boasson, J. Kaandorp, J.Tonino, M. Vosselman (eds.), *Proc. 5th Annual Conference of the Advanced School for Computing and Imaging* (Heijen, NL, June 15-17), 1999, 398-405.

[56] M. Tipping. The Relevance Vector Machine. In *Advances in Neural Information Processing Systems, 12.* S. Solla, T. Leen nd K.-R. Muller (eds) MIT Press, Cambridge, MA, 2000 (to appear).

[57] http://www.ics.uci.edu/~mlearn/MLRepository.html

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

[59] V. Vapnik. *Statistical Learning Theory.* Wiley, 1998.

[60] V. Vapnik and O. Chapelle. Bounds on error expectation for Support Vector Machines. Submitted to *Neural Computation*, 1999

[61] K. Veropoulos, C. Campbell and N. Cristianini. Controlling the sensitivity of support vector machines. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Stockholm, Sweden, 1999.

[62] M. Vidyasagar. *A theory of learning and generalisation.* Springer-Verlag, Berlin, 1997.

[63] C. Watkins. Dynamic Alignment Kernels, Technical Report, UL Royal Holloway, CSD-TR-98-11, 1999.

[64] J. Weston and C. Watkins. Multi-Class Support Vector Machines, in *Proceedings of ESANN99,* ed. M Verleysen, D. Facto Press, Brussels, p. 219-224, 1999.

[65] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, C. Lemmen, A. Smola, T. Lengauer and K.-R. Müller. Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. Presented at the German Conference on Bioinformatics, 1999.