Query Learning with Large Margin Classifiers

Colin Campbell C.Campbell@bris.ac.uk
Department of Engineering Mathematics, University of Bristol, Bristol BS8 1TR, United Kingdom

Nello Cristianini Nello.Cristianini@bris.ac.uk
Department of Engineering Mathematics, University of Bristol, Bristol BS8 1TR, United Kingdom

Alex Smola Alex.Smola@anu.edu.au
Department of Engineering and RSISE, Australian National University, Canberra, ACT 0200, Australia

Abstract

The active selection of instances can significantly improve the generalisation performance of a learning machine. Large margin classifiers such as support vector machines classify data using the most informative instances (the support vectors). This makes them natural candidates for instance selection strategies. In this paper we propose an algorithm for the training of support vector machines using instance selection. We give a theoretical justification for the strategy and experimental results on real and artificial data demonstrating its effectiveness. The technique is most efficient when the data set can be learnt using few support vectors.

1. Introduction

The labour-intensive task of labelling data is a serious bottleneck for many data mining tasks. Often cost or time constraints mean that only a fraction of the available instances can be labeled. For this reason there has been increasing interest in the problem of handling partially labeled data sets.

One approach to this problem is *query learning* (Angluin, 1988) where the learning machine is allowed to actively interrogate its environment or data source rather than just passively waiting for data. In particular, for learning with *membership queries* the algorithm creates or selects unlabeled instances for the human expert to label. One problem with creating queries is that sometimes the most informative ones are meaningless and impossible for the human expert to label (Baum, 1991) e.g.the most informative query may be an improper character for an OCR task. A better approach may be to use query learning in conjunction with partially labeled data sets so that the unlabeled part becomes a reserve of potential queries (Freund et al., 1997). The task is then reduced to querying the most informative points in the data: a problem known as *instance selection* or *selective sampling.*

Large margin classifiers (Smola et al., 2000) based on boosting (Freund & Schapire, 1997) or support vector machines (SVMs) (Vapnik, 1998; Cristianini & Shawe-Taylor, in press) have the property of focusing on a subset of the most informative patterns in a data set and only using these to construct the hypothesis. Such patterns may be called *support patterns* or in the context of SVMs *support vectors.* In this sense Large Margin Classifiers are a natural candidate for selective sampling strategies. If one knew *a priori* the identity of the support patterns in a data set, it would be possible to discard all the other patterns, and still recover the same final hypothesis.

Theoretical results and artificial examples discussed in Rivest and Eisenberg (1990) show that it is possible to invent malicious distributions for which the number of queries is comparable to the sample size, hence removing any advantage. In practice, such adversarial distributions may not occur frequently. Indeed we find that the *sparsity* of the solution is the most important factor: if the hypothesis requires comparatively few support vectors in relation to the total data set size then selective sampling works well.

Following a brief overview of large margin techniques and SVMs in Section 2 we outline a strategy for selecting support patterns efficiently by considering the minimax contribution of patterns to the regularized risk functional in Section 3. An efficient stopping criterion is considered in in Section 3.3. Experimental results are presented in Section 4 showing that the algorithm efficiently identifies the most informative data points.

2. Large Margin Classifiers

2.1 Risk Functionals

Let X be the space of patterns, $Y := \{-1, 1\}$ the space of labels (i.e. target values) and $p(\mathbf{x}, y)$ a probability distribution on $X \times Y$. Then the classification problem consists of finding a mapping $g : X \to Y$ which minimizes the risk of misclassification

$$R[g] = \int |y - g(\mathbf{x})| p(\mathbf{x}, y) d\mathbf{x} = \Pr(y \neq g(\mathbf{x})) \quad (1)$$

In practice one uses real valued functions $f(\mathbf{x}) : X \to \Re$ rather than binary functions g and substitutes $g(\mathbf{x}) = \operatorname{sign}(f(\mathbf{x}))$. Furthermore, $p(\mathbf{x}, y)$ is not readily available and one has to use a (labelled) training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\}$ of ℓ points in order to infer a suitable function f. This is generally done by minimizing the empirical estimate (or training error) of $R[f]$, i.e.

$$R_{emp}[f] = \frac{1}{\ell} \sum_{i=1}^{\ell} |y_i - \operatorname{sign}(f(\mathbf{x}_i))| \quad (2)$$

However, minimization of (2) is usually ill posed and does not guarantee small $R[f]$. Good bounds on the generalization error (Vapnik, 1998; Cristianini & Shawe-Taylor, in press) can be obtained by seeking a function f that achieves a large margin on the training set, i.e. there exists a constant γ such that $y_i f(\mathbf{x}_i) \geq \gamma$ for all $1 \leq i \leq \ell$. This goal is usually achieved by replacing the binary loss function in (2)

$$c(\mathbf{x}, y, f(\mathbf{x})) = |y - \operatorname{sign}(f(\mathbf{x}))| \quad (3)$$

by a margin-type loss function such as the soft margin loss

$$c(\mathbf{x}, y, f(\mathbf{x})) = \max(0, 1 - yf(\mathbf{x}))^s \text{ where } s \geq 1 \quad (4)$$

or adaptive loss settings such as ν-loss (Schölkopf et al., 1999). However, minimizing such a modified loss function still may be an ill posed problem and minimizing it may just as well lead to poor generalization performance.

Consequently a term $\Omega[f]$ controlling the class of admissible functions has to be added and one therefore obtains a regularized risk functional (cf. Smola (1998))

$$\begin{aligned} R_{reg}[f] &= R_{emp}[f] + \lambda\Omega[f] \\ &= \frac{1}{\ell} \sum_{i=1}^{\ell} c(\mathbf{x}_i, y_i, g(\mathbf{x}_i)) + \lambda\Omega[f]. \quad (5) \end{aligned}$$

Here $\lambda > 0$ is the regularization (or trade–off) constant which determines how much functions minimizing complexity are favoured over functions minimizing $R_{emp}[f]$.

The problem of selective sampling can now be viewed as follows. The minimizer of $R_{reg}[f]$ has to be found by selecting a set $I \subset \{1, \ldots \ell\}$ of indices (if possible with $|I| \ll \ell$) such that minimization of

$$\begin{aligned} R^I_{reg}[f] &= R_{emp}[f] + \lambda\Omega[f] \\ &= \frac{1}{\ell} \sum_{i \in I} c(\mathbf{x}_i, y_i, g(\mathbf{x}_i)) + \lambda\Omega[f] \quad (6) \end{aligned}$$

leads to some f^* that comes close to minimizing $R_{reg}[f]$, too. Given some index I_{old} one strategy would be to test all $2^{\ell - |I_{\text{old}}|}$ possible combinations of the labellings so that the minimal improvement taken over all possible labellings is then maximised. Clearly the complexity of such an operation is prohibitively high and we must instead opt for a computationally cheaper *heuristic* strategy. This will be done in Section 3. Before going into details let us briefly review the special type of learning machines that we will use for our experiments.

2.2 Support Vector Machines

Support vector machines (SVMs) implement complex decision rules by using a non-linear function ϕ to map training points to a high-dimensional feature space, where an optimal hyperplane is found which will minimise the regularized risk functional $R_{reg}[f]$. The procedure for finding this optimal hyperplane can be reduced to a quadratic programming task involving maximisation of a corresponding dual objective function. The solution found by this optimisation task has the remarkable property that it can be exclusively expressed using a subset of the data (the *support vectors*). The task of finding this subset and omitting the labelling of the rest gives an obvious motivation for considering query learning using SVMs.

For a SVM one considers the class of (non)linear functions given by

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \text{ or } f(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b. \quad (7)$$

In particular, the separating hyperplane in X can be characterised by the equation $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = 0$,

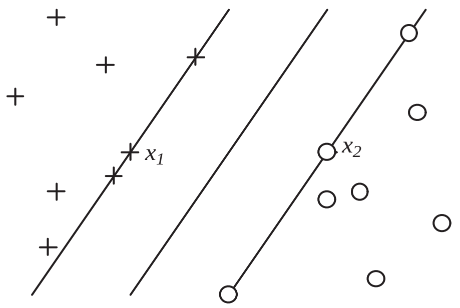

Figure 1. The *margin*, γ, is the perpendicular distance between the separating hyperplane and the closest points (these are the *support vectors*). The region between the hyperplanes through the support vectors on each side is called the *margin band.* $\mathbf{x}_1$ and $\mathbf{x}_2$ are examples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

with $f(\mathbf{x}_i) \geq 0$ if $y_i = +1$ and $f(\mathbf{x}_i) < 0$ if $y_i = -1$. The distance between the separating hyperplane and the closest points is called the *margin* and denoted γ (Figure 1). The margin can be written $\gamma = \|\mathbf{w}\|^{-1}$ (cf. Vapnik, 1998) so we may write the SV optimization problem as minimization of the regularized risk functional

$$R_{reg}[f] = \frac{1}{\ell}\sum_{i=1}^{\ell} c(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) + \frac{\lambda}{2}\|\mathbf{w}\|^2. \quad (8)$$

Computing the Wolfe dual from (8) yields the standard SV optimization task (Vapnik, 1998), namely maximise:

$$W(\alpha) = -\sum_{i=1}^{\ell}\alpha_i + \frac{1}{2}\sum_{i,j=1}^{\ell} y_i y_j \alpha_i \alpha_j \left[\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)\right] \quad (9)$$

subject to $\sum_{i=1}^{\ell}\alpha_i y_i = 0$. For the hard margin case: $\alpha_i \in [0, \infty)$, while for the soft margin loss with $s = 1$: $\alpha_i \in [0, \lambda\ell]$ and: $\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_i) \to \phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_i) + \lambda\ell/2$ for $s = 2$. The weight vector $\mathbf{w}$ is found to be

$$\mathbf{w} = \sum_{i=1}^{\ell}\alpha_i y_i \phi(\mathbf{x}_i). \quad (10)$$

One can show (see e.g. Smola (1998), Smola & Schölkopf., 1998) for details) that the size of the loss function $c(\mathbf{x}_i, y_i, f(\mathbf{x}_i))$ and of the corresponding Lagrange multiplier α_i is connected via

$$\alpha_i \in \bar{\partial}_{\alpha_i} c(\mathbf{x}_i, y_i, f(\mathbf{x}_i)), \quad (11)$$

where $\bar{\partial}_{\alpha_i}$ denotes the *subdifferential*[1] with respect to α_i. For the case of a soft margin loss this means that we can infer two things. Firstly the size of α_i indicates whether the sample is inside the margin or not, provided we have already trained on it. Secondly it allows us to predict whether adding a new sample would increase the dual objective function (and decrease the primal objective) since for $\alpha_i = 0$, which is the default for patterns not yet trained on, only if $c(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) \neq 0$ can any gain be achieved.

In our later experimental study we define the *sparsity ratio* ω as the ratio of the number of support vectors over total data set size (i.e. the fraction of points with non-zero subdifferential). Certain data sets can be modelled by a *sparse* hypothesis with relatively few support vectors in the decision function, whereas for other data sets the hypothesis modelling the data can be *dense* with a sparsity ratio nearer unity.

To complete the implementation of a SVM the final task is to perform an implicit mapping into feature space by replacing the inner product $\phi(\mathbf{x})\cdot\phi(\mathbf{x}')$ by a closed form expression $k(\mathbf{x}, \mathbf{x}')$ which satisfies Mercer's condition (Vapnik, 1998; Cristianini & Shawe-Taylor in press). Possible choices of kernel are Gaussians:

$$k(\mathbf{x}', \mathbf{x}) = e^{-||\mathbf{x}-\mathbf{x}||^2/2\sigma^2} \quad (12)$$

or polynomial kernels $k(\mathbf{x}', \mathbf{x}) = (\mathbf{x}\cdot\mathbf{x}' + 1)^d$.

3. An Algorithm for Selecting Instances

3.1 Selection Strategies

Given a subset I of the data we want to infer which instance to choose next. Since we do not know the labels y_i on the rest of the data set we have to make assumptions about the distribution of the y_i. As already noted we cannot afford to solve the optimization problem for all $2^{\ell-|I|}$ possible sets of labellings $y_i = \{\pm 1\}$ before picking one point. Thus we must use a heuristic and we will assume that all terms $c(\mathbf{x}_i, y_i, f(\mathbf{x}_i))$ where $i \neq i_{\text{new}}$ have an equal influence. This approximation still leaves room for several strategies:

Random Labels. We assume that the predictions of the (so far) found classifier f_I (which minimizes $R^I_{reg}[f]$) are completely uncorrelated with the labels on the unlabelled data. In this case we want to find the index i corresponding to the largest expected error

[1] The advantage of the latter is that is it is well defined even for continuous functions which are nondifferentiable only on a countable set. There the subdifferential yields the interval between the lhs and rhs limit of the standard derivatives. In other words, $\bar{\partial}_\alpha|\alpha|$ yields $\text{sign}(\alpha)$ for $\alpha \neq 0$ and $[-1, 1]$ for $\alpha = 0$.

contribution, i.e.

$$i = \arg\max_{i \notin I} \frac{1}{2} \left[c\left(\mathbf{x}_i, 1, f\left(\mathbf{x}_i\right)\right) + c\left(\mathbf{x}_i, -1, f\left(\mathbf{x}_i\right)\right) \right]. \tag{13}$$

In the soft margin case this simplifies to

$$\begin{aligned} i &= \arg\max_{i \notin I} \frac{1}{2} \left[\max(0, 1 - f(\mathbf{x}_i))^s \right. \\ & \left. \quad + \max(0, 1 + f(\mathbf{x}_i))^s \right]. \end{aligned} \tag{14}$$

The assumption of randomness, however, may only be good in the initial stage of training, where we can assume that the estimate has little to do with the actual data. Later on, however, we *expect* that $\text{sign}(f(\mathbf{x}_i))$ and y_i are positively correlated since the algorithm will have *learned* something about the training set, so the assumption of randomness is a bad choice.

Worst Case. This implies, however, that the error $c(\mathbf{x}_i, y_i, f(\mathbf{x}_i))$ can be expected to be small since the labelling most likely will be correct (and possibly only the margin small), i.e. y_i is more likely to be $\text{sign} f(\mathbf{x}_i)$ rather than $-\text{sign} f(\mathbf{x}_i)$. This is, however, where $c(\mathbf{x}_i, y_i, f(\mathbf{x}_i))$ is smallest. Thus we choose i according to a worst case strategy on y_i:

$$i = \arg\max_{i \notin I} \min(c(\mathbf{x}_i, 1, f(\mathbf{x}_i)), c(\mathbf{x}_i, -1, f(\mathbf{x}_i))). \tag{15}$$

In the soft margin case this can be simplified to

$$i = \arg\min_{i \notin I} |f(\mathbf{x}_i)|. \tag{16}$$

In other words, the points closest to the decision boundary are chosen. However, this also entails that in some cases no pattern $\mathbf{x}_i$ will have a nonzero minimax error (if all $|f(\mathbf{x}_i)| \geq 1$) i.e. the margin is empty.

Empty Margin. In the case of an "empty" margin we have two conflicting issues. On the one hand, (13) suggests that we choose an $\mathbf{x}_i$ with a very large function value, $f(\mathbf{x}_i)$. On the other hand, (15) would suggest choosing a pattern *close* to the margin. Neither of the two strategies appears to be optimal, thus we pick a pattern randomly from $\{1, \ldots \ell\} \backslash I$. We will use the labels of random patterns in Section 3.3 to decide when to stop.

Hard Margin. Finally, for the hard margin cost function, c assumes only two function values: 0 and ∞ (0 if $y_i f(\mathbf{x}_i) \geq 1$ and ∞ otherwise). This makes both (13) and (15) fail equally. Hence we have to do some more analysis in that case (however, we limit ourselves to SV machines). We assume that a feasible solution of the optimization problem exists, in other words, that the data set is separable.

By analyzing the objective function $\Omega[f] = \frac{1}{2}\|\mathbf{w}\|^2$, subject to constraints $y_i f(\mathbf{x}_i) \geq 1$, one can see that training on a subset I is equivalent to ignoring some of the constraints, and therefore adding data to I (adding constraints) will only increase the objective function $\|\mathbf{w}\|^2$. Hence, at every step we would like to add in the constraint that causes $\Omega[f]$ to increase most (since we start with an f that is *not a feasible solution* of the optimization problem yet and a larger w will be closer to the overall feasible solution). The largest increase comes from a large decrease of the minimum margin $\gamma = 1/\|\mathbf{w}\|$, which is achieved by patterns lying as close as possible to the decision boundary. Hence the selection rule is identical to that of the 'worst case' strategy presented above.

3.2 The Algorithm

So far we have omitted any description of stopping criteria for the algorithm. We will come to this issue in Section 3.3 since it is largely detached from the description of the training algorithm itself. Following the above argument our strategy is to start by requesting the labels of a random subset of instances *and subsequently iteratively requesting the label of that data point which is closest to the current hyperplane.* For learning with a hard margin we list the algorithm on the next page.

Many real life data sets contain noise and outliers in the data. These can be readily handled using the soft margin technique for (9): either using the $s = 2$ loss amounting to an addition to the diagonal components of the kernel matrix (Shawe-Taylor & Cristianini, 1999) $K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \nu$ (ν, a positive constant) or by limiting the influence of individual Lagrange multipliers $0 \geq \alpha_i \geq \nu$ (for $s = 1$).

In practical situations cost or time constraints may cause early stopping of the labelling process. However, if this is not an immediate concern, an important question is the stopping criterion for effective termination of the process. A naive strategy would be to stop when the dual objective function stops increasing, i.e. when the queried points stop yielding additional information. As we illustrate in section 4 this does work in practice but only for noise-free data. A better heuristic is to stop when the margin band is empty as we outline in the algorithm below and detail in the next Section:

An Algorithm for Hard Margin Query Learning

```
Input the unlabelled training patterns $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$.
Program Set $\mathbf{w}, b = 0$.
    Randomly select an initial starting set $I$ of $l$ instances from the training data.
Repeat
    Train on the set $\{\mathbf{x}_i | i \in I\}$:
        $\max \left[ \sum_{i \in I} \alpha_i - \frac{1}{2} \sum_{i,j \in S} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right]$
        subject to $\sum_{i \in I} y_i \alpha_i = 0$ and $\alpha_i \geq 0$
    Compute the decision function $f(\mathbf{x}_i)$ for all unlabelled members of the training set:
        $f(\mathbf{x}) = \sum_{i \in I} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b$
        where $b$ is chosen according to:
        $b = \frac{1}{2} \left[ \min_{\{i | i \in I, y_i = +1\}} (\mathbf{w} \cdot \phi(\mathbf{x}_i)) + \max_{\{i | i \in I, y_i = -1\}} (\mathbf{w} \cdot \phi(\mathbf{x}_i)) \right]$
    Pick the training pattern, $i^\star$, such that:
        $i^\star = \arg \min_{i \notin I} |f(\mathbf{x}_i)|$
    If $|f(\mathbf{x}_i)| \geq 1$ then
        The margin band is empty, hence select a validation set of $m$ instances at random from the unlabelled training set (see Section 3.3).
    else Query the label $y_{i^\star}$ and increment the set: $I = I \cup \{i^\star\}$
    endif
until all labels have been requested, or the stopping criterion has been met on the validation set of $m$ instances (Section 3.3)
Output the number of queries, query set $I$ and coefficients $\alpha_i$ and $b$.
```

3.3 A Stop and Query Termination Criterion

A good stopping rule should be able to give an upper bound on the probability of error on the remaining (unlabelled) training set, given the current hypothesis and the outcomes of previous queries (i.e. the label set $\{y_i : i \in I\}$). The main tool in the subsequent analysis is a theorem by Hoeffding (1963) bounding the deviation of empirical means from their expectations.

Theorem 1: Let $\xi_1, \ldots, \xi_m$ be independent bounded random variables such that ξ_i falls in the interval $[a_i, a_i + b_i]$ with probability one. Denote their average by $S_m = \frac{1}{m} \sum_i \xi_i$. Then for any $\epsilon > 0$ one has

$$\left. \begin{array}{l} \Pr\{S_m - E[S_m] \geq \epsilon\} \\ \Pr\{E[S_m] - S_m \geq \epsilon\} \end{array} \right\} \leq \exp\left\{ -\frac{2m^2\epsilon^2}{\sum_{i=1}^m b_i^2} \right\} \tag{17}$$

We have to adapt this statement to a large deviations statement concerning sample averages over different sample size. This can be readily constructed as follows. We quote a theorem Smola, Mangasarian and Schölkopf (1999) which contains exactly the statement we need.

Theorem 2: 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 $\xi_{s(1)}, \ldots, \xi_{s(\widetilde{M})}$ with $\widetilde{M} < M$ a subset of the same random variables (with $s : \{1, \ldots, \widetilde{M}\} \to \{1, \ldots, M\}$ being an injective map), and $S_{\widetilde{M}} = 1/\widetilde{M} \sum_i \xi_{s(i)}$. Then for any $\epsilon > 0$ one has

$$\left. \begin{array}{l} \Pr\{S_M - S_{\widetilde{M}} \geq \epsilon\} \\ \Pr\{S_{\widetilde{M}} - S_M \geq \epsilon\} \end{array} \right\} \leq \exp\left\{ -\frac{2M\widetilde{M}\epsilon^2}{(M - \widetilde{M})b^2} \right\} \tag{18}$$

In the present case we set $b = 1$ (training errors count as 1), m denotes the size of the (so far) unlabelled training set $M = m - |I|$, and $\widetilde{M}$ is the size of a subset of $\{1, \ldots m\} \backslash I$ on which no training has been performed but on which the labels already have been queried.

Thus with confidence $1 - \eta$ the training error (the same statement also holds for the margin error) on the whole untested subset can be bounded by

$$S_{\widetilde{M}} \leq S_M + \epsilon \text{ where } -\log \eta \leq \frac{2M\widetilde{M}\epsilon^2}{M - \widetilde{M}}. \tag{19}$$

In other words, if, at some point, we perform $\widetilde{M}$ queries without updating the current hypothesis and observe a (margin) error $S_{\widetilde{M}}$ on these queries, (19) can be applied. However, we may not be satisfied with the first test query and decide to do some more training (and subsequently perform another query) before stopping the algorithm. There exists always a probability that we just might have been lucky (the η term), and these terms could add up. This can be countered by systematically enlarging $\widetilde{M}$ every time we perform such a query such that the bound holds for $\eta_k = 2^{-k}\eta$. Then, even if these 'lucky' events are independent, we can still make a statement with $1 - \sum_k \eta_k \leq 1 - \eta \sum_{k=1}^\infty 2^{-k} = 1 - \eta$ confidence. Hence we obtain

$$
\begin{aligned}
S_{\widetilde{M}} &\leq S_M + \sqrt{\frac{(M-\widetilde{M})(k\log 2 - \log\eta)}{2M\widetilde{M}}} &(20)\\
&< S_M + \sqrt{\frac{k\log 2 - \log\eta}{2\widetilde{M}}}. &(21)
\end{aligned}
$$

A practical algorithm using (20) will proceed as follows: train until either the margin band is empty or until a stopping point specified by the user has been met. Perform a random sampling of unlabelled samples such that with a confidence η (specified by the user) a bound on the remaining training error can be stated. If the user is satisfied then stop or continue with training/querying if desired. Since we would like our confidence rating to increase as we continue sampling, we fix ϵ to be stated in terms of the remaining error on the unlabelled data rather than the entire data set.

4. Experiments

We now compare selective sampling and random sampling on 2 artificial and 2 real data sets using the stopping criteria stated above. The computational cost of finding the data point closest to the current hyperplane is small compared to the cost involved in learning so there is little difference between training times for the two techniques. Predictably the number of queries needed to achieve optimal performance depends on the sparsity ratio, ω.

Artificial Data. As our first artificial example we will choose a data set generated by the *majority rule* (a 1 if the majority of bits in the input string are +1 and a −1 if the majority consists of −1). In Figure 2 we see that selective sampling has a clear advantage over random sampling of data points. In particular, after querying a subset of about 60 samples, the decision function is learnt correctly.

Figure 3 shows the rapid increase in the dual objective function (9) for query learning with the optimum achieved after about 60 instances. Thus for noiseless data sets - as in this case - monitoring the dual objective function provides a good stopping criterion

Figure 4 shows the corresponding performance for the *shift detection* problem (Nowlan & Hinton, 1992). Compared to the majority rule, selective sampling appears to have a less dramatic effect. For shift detection the sparsity ratio is $\omega = 0.53$ in contrast to the majority rule with $\omega = 0.28$. Thus selective sampling appears to be most effective if the target concept is sparse. This makes sense since, with few support vectors, the dual objective function will stop increasing if these support vectors can be found early in the process. Indeed, for the majority rule, for example, we notice that the rule is efficient in identifying support vectors since, with a training set of 200 (Figure 2) there are an average 56 support vectors against an average 60 queries made.

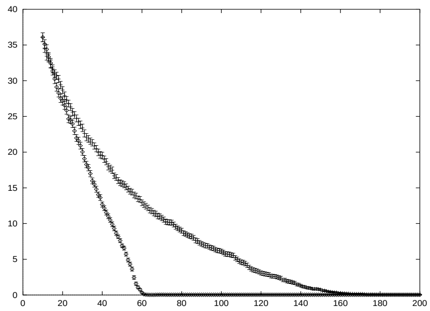

Figure 2. Generalisation error (y axis) as a percentage versus number of patterns learnt (x axis) for random sampling (top, right curve) and query learning (bottom, left curve). **Majority rule** (with 20 components per instance) randomly split into 200 training and 200 test instances averaged over 100 samplings; Gaussian kernels with $\sigma = 20.0$.

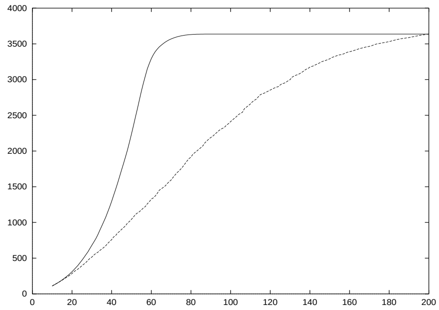

Figure 3. The averaged value of the objective function (y axis) versus number of instances learnt (x axis) for the **majority rule**. The upper (solid) curve corresponds to query learning; lower (dashed) curve to random sampling.

Real World Data. In Figure 5 we plot the corresponding curves for the *ionosphere data set* from the UCI Repository (Blake, Keogh & Merz, 1998). The ionosphere data set had a sparsity ratio of 0.29 so the advantages of selective sampling are clear. A plot of the averaged distance to the separating hyperplane indicates the closest points are outside the margin band after an average 94 instances (Figure 6). We notice that the generalisation error has a minimum for selective sampling with a hard margin loss: after passing through this minimum the generalisation error increases as the system learns outliers. In fact the cri-

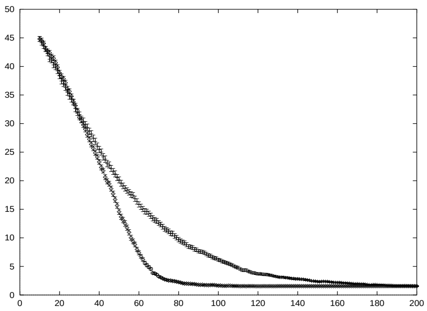

Figure 4. Generalisation error (y axis) as a percentage versus number of patterns learnt (x axis) for random sampling (top, right curve) and query learning (bottom, left curve). **Shift detection** data set randomly split into 200 training and 200 test instances averaged over 100 samplings; Gaussians kernel with $\sigma = 5.0$.

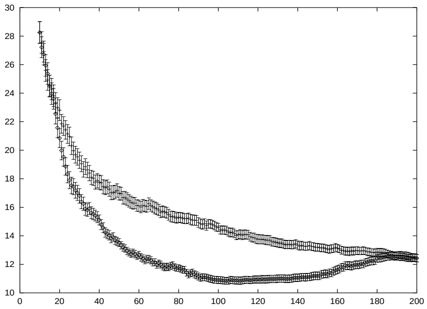

Figure 5. Generalisation error (y axis) as a percentage versus number of patterns learnt (x axis) for random selection (top curve) and query learning (bottom curve). UCI **ionosphere** data set randomly split into 200 training and 151 test instances averaged over 100 samplings; Gaussian kernel with $\sigma = 2.0$.

terion of stopping when the margin band is empty appears a good heuristic for avoiding the learning of outliers for this data set. However, as mentioned earlier, the influence of noise and outliers is best reduced by introducing a soft margin loss function (4), for example, by using the replacement $K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \nu$ (ν positive) throughout the algorithm outlined in Section 3.2.

As a second example for real-life data sets we have investigated classification of single handwritten digits (one-against-all) for data from the USPS set of handwritten postal codes (Lecun et al., 1989). For the digit illustrated in Figure 7 we find the generalisation error achieved by learning the entire data set (0.59%) is already achieved with approximately 1,000 queries.

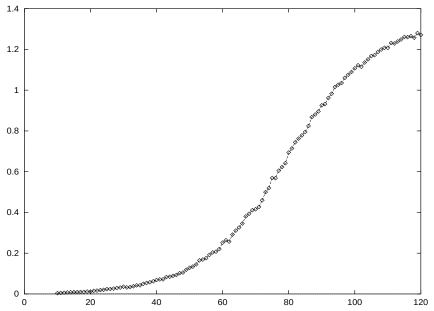

Figure 6. The average minimal distance between the closest point and the current separating hyperplane (y axis) versus number of instances learnt (x axis) for the **ionosphere** data set averaged over 100 samplings. After an average 94 instances the closest points lie outside the margin band.

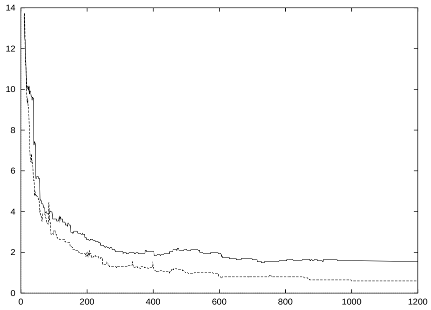

Figure 7. Generalisation error (y axis) as a percentage versus number of patterns learnt (x axis) for random sampling (top, solid curve) and query learning (bottom, dashed curve). **USPS** data set with one-against-all classification of digit 0. Label queries were made from a set of 7,291 digits (each consisting of 16×16 images with each component lying in the range -1 to 1). The test set consisted of 2,007 digits. Gaussian kernels were used with $\sigma = 1.8$.

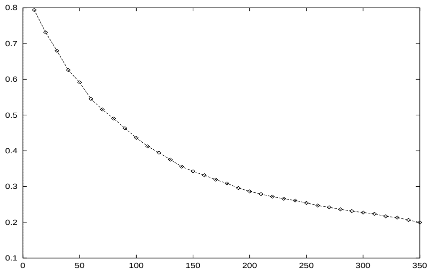

Figure 8. The sparsity ratio (y axis) versus number of patterns learnt (x axis) for the UCI **ionosphere** data set (each data point has been averaged over 100 samplings of the data). As the size of the data set increases the sparsity ratio tends to decrease. We have observed the same phenonenon with all the data sets considered here.

5. Conclusion

In all the numerical experiments considered selective sampling led to a more rapid decrease in generalisation error compared to random selection, though the advantages of selective sampling can be small if the hypothesis modelling the data is dense. This is not very surprising since in cases where much data is *needed* to model the hypothesis precisely, by definition one simply *cannot* achieve the same result with less data. However, sparsity ratios generally decrease as the volume of data increases (Figure 8) reaching a lower limit (Smola, 1996). Thus the method should be particularly useful for large data sets. Despite the fact that you can initiate the algorithm with the previous α_i solution every time a new data point is learnt, training times could still be too long as more data is learnt for a large data set. In this case one could query the labels of a *set* of points nearest the current hyperplane rather than a single data point, reducing training time at the expense of a small decrease in performance.

References

Angluin, D. (1988). Queries and concept learning. *Machine Learning*, *2*, 319–342.

Baum, E. B. (1991). Neural networks which learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks*, *2*, 5–19.

Blake, C., Keogh, E., & Merz, C.J. (1998). UCI repository of machine learning databases (www.ics.uci.edu/∼mlearn/MLRepository.html).

Cristianini, N. & Shawe-Taylor, J. (in press). *An introduction to support vector machines.* Cambridge, UK: Cambridge University Press.

Freund, Y. & Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Freund, Y., Seung, S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning*, *28*, 133–168.

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

Le Cun, Y., Boser, B., Denker J.S., Henderson D., Howard R.E., Hubbard W., & Jackel L.J. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation 1*, 541-551.

Nowlan S.J. & Hinton, G.E. (1992). Simplifying neural networks by soft weight-sharing. *Neural Computation*, *4*, 473–493.

Rivest R.L. & Eisenberg, B. (1990). On the sample complexity of pac-learning using random and chosen examples. In *Proceedings of the 1990 Workshop on Computational LearningTheory* (pp. 154–162). San Mateo, CA: Morgan Kaufmann.

Schölkopf, B. Bartlett, P.L., Smola, A. & Williamson, R. (1999). Shrinking the tube: a new support vector regression algorithm. In Kearns, M.S., Solla, S.A., and Cohn, D.A., editors, *Advances in Neural Information Processing Systems 11* (pp. 330 – 336). Cambridge, MA: MIT Press.

Shawe-Taylor J. & Cristianini, N. (1999). Further results on the margin distribution. In *Proceedings of the Twelth Annual Conference on Computational Learning Theory, COLT'99.*

Smola, A.J. (1996). Regression estimation with support vector learning machines. Master's thesis, Department of Physics, Technische Universität München.

Smola, A.J. (1998). *Learning with kernels.* PhD thesis, Department of Computer Science, Technische Universität Berlin.

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

Smola, A.J. & Schölkopf, B. (1998). On a kernel–based method for pattern recognition, regression, approximation and operator inversion. *Algorithmica*, *22*, 211–231.

Smola, A.J., Bartlett, P.L., Schölkopf, B., & Schuurmans, D. (in press). *Advances in Large Margin Classifiers.* Cambridge, MA: MIT Press.

Vapnik, V. (1998). *Statistical learning theory.* New York: Wiley.