Simplified Support Vector Decision Rules

Chris J.C. Burges
Bell Laboratories, Lucent Technologies
Room 4G-302, 101 Crawford's Corner Road
Holmdel, NJ 07733-3030
cjcb@big.att.com

Abstract

A Support Vector Machine (SVM) is a universal learning machine whose decision surface is parameterized by a set of *support vectors*, and by a set of corresponding weights. An SVM is also characterized by a *kernel* function. Choice of the kernel determines whether the resulting SVM is a polynomial classifier, a two-layer neural network, a radial basis function machine, or some other learning machine.
SVMs are currently considerably slower in test phase than other approaches with similar generalization performance. To address this, we present a general method to significantly decrease the complexity of the decision rule obtained using an SVM. The proposed method computes an approximation to the decision rule in terms of a reduced set of vectors. These reduced set vectors are not support vectors and can in some cases be computed analytically. We give experimental results for three pattern recognition problems. The results show that the method can decrease the computational complexity of the decision rule by a factor of ten, with no loss in generalization performance, making the SVM test speed competitive with that of other methods. Further, the method allows the generalization performance/complexity trade-off to be directly controlled. The proposed method is not specific to pattern recognition and can be applied to any problem where the Support Vector algorithm is used (for example, regression).

1 INTRODUCTION

1.1 SUPPORT VECTOR MACHINES

Consider a two-class classifier for which the decision rule takes the form:

$$y = \Theta(\sum_{i=1}^{N_S} \alpha_i K(\mathbf{x}, \mathbf{s}_i) + b), \tag{1}$$

where $\mathbf{x}, \mathbf{s}_i \in \mathbf{R}^d$, $\alpha_i, b \in \mathbf{R}$, and Θ is the step function; $\alpha_i, \mathbf{s}_i, N_S$ and b are parameters and $\mathbf{x}$ is the vector to be classified. The decision rule for a large family of classifiers can be cast in this functional form: for example, $K = (\mathbf{x} \cdot \mathbf{s}_i)^p$ implements a polynomial classifier; $K = \exp(-\|\mathbf{x} - \mathbf{s}_i\|^2/\sigma^2)$ implements a radial basis function machine; and $K = \tanh(\gamma(\mathbf{x} \cdot \mathbf{s}_i) + \delta)$ implements a two-layer neural network [1; 2; 3; 4].

The support vector algorithm is a principled method for training any learning machine whose decision rule takes the form (1): the only condition required is that the kernel K satisfy a general positivity constraint [2; 3]. In contrast to other techniques, the SVM training process determines the entire parameter set $\{\alpha_i, \mathbf{s}_i, b, N_S\}$; the resulting $\mathbf{s}_i, i = 1, \ldots, N_S$ are a subset of the training set and are called *support vectors*.

Support Vector Machines have a number of striking properties. The training procedure amounts to solving a constrained quadratic optimization problem, and the solution found is thus guaranteed to be the unique global minimum of the objective function. SVMs can be used to directly implement Structural Risk Minimization, in which the capacity of the learning machine can be controlled so as to minimize a bound on the generalization error [2; 4]. A support vector decision surface is actually a linear separating hyperplane in a high dimensional space; similarly, SVMs can be

used to construct a regression, which is linear in some high dimensional space [2].

Support Vector Learning Machines have been successfully applied to pattern recognition problems such as OCR [5; 2; 4], text independent speaker identification [9], and object recognition [10]; they are also being investigated for other problems, for example regression.

1.2 REDUCED SET VECTORS

The complexity of the computation (1) scales with the number of support vectors N_S. The expectation of the number of support vectors is bounded below by $(\ell - 1)E(p)$, where $E(p)$ is the expectation of the probability of error on a test vector and ℓ is the number of training samples [2]. Thus N_S can be expected to approximately scale with ℓ. For practical pattern recognition problems, this results in a machine which is considerably slower in test phase than other systems with similar generalization performance [6; 7]. This fact motivated the work reported here: below, we present a method to approximate the SVM decision rule with a much smaller number of *reduced set* vectors. The reduced set vectors have the following properties:

- They appear in the approximate SVM decision rule in the same way that the support vectors appear in the full SVM decision rule;
- They are not support vectors; they do not necessarily lie on the separating margin, and unlike support vectors, they are not training samples;
- They are computed for a given, trained SVM;
- The number of reduced set vectors (and hence the speed of the resulting SVM in test phase) is chosen *a priori*;
- The reduced set method is applicable wherever the support vector method is used (for example, regression). In this paper, we will consider only the pattern recognition case.

1.3 DATA

We quote results on two OCR data sets containing grey level images of the ten digits: a set of 7,291 training and 2,007 test patterns, which we refer to as the "postal set" [6; 11], and a set of 60,000 training and 10,000 test patterns from NIST Special Database 3 and NIST Test Data 1, which we refer to as the "NIST set" [12]. Postal images were 16x16 pixels and NIST images were 28x28 pixels. On the NIST set we restricted ourselves to classifiers that separate digit 0 from all other digits.

2 THE REDUCED SET

Let the training data be elements $\mathbf{x} \in \mathcal{L}, \mathcal{L} = \mathbf{R}^{d_L}$. An SVM performs an implicit mapping $\Phi : \mathbf{x} \rightarrow \bar{\mathbf{x}}, \bar{\mathbf{x}} \in \mathcal{H}, \mathcal{H} = \mathbf{R}^{d_H}, d_H \leq \infty$. In the following, vectors in $\mathcal{H}$ will be denoted with a bar. The mapping Φ is determined by the choice of kernel K. In fact, for any K which satisfies Mercer's positivity constraint [2; 3], there exists a pair $\{\Phi, \mathcal{H}\}$ for which $K(\mathbf{x}_i, \mathbf{x}_j) = \bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_j$. Thus in $\mathcal{H}$, the SVM decision rule is simply a linear separating hyperplane. The mapping Φ is usually not explicitly computed, and the dimension d_H of $\mathcal{H}$ is usually large (for example, for the homogeneous map $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^p, d_H = C(p + d_L - 1, d_L)$; thus for degree 4 polynomials and for $d_L = 256$, d_H is approximately 2.8 million).

The basic SVM pattern recognition algorithm solves a two-class problem [1; 2; 3]. Given training data $\mathbf{x} \in \mathcal{L}$ and corresponding class labels $y_i \in \{-1, 1\}$, the SVM algorithm constructs a decision surface $\bar{\Psi} \in \mathcal{H}$ which separates the $\mathbf{x}_i$ into two classes ($i = 1, \ldots, \ell$):

$$\bar{\Psi} \cdot \bar{\mathbf{x}}_i + b \geq k_0 - \xi_i, \quad y_i = +1 \tag{2}$$

$$\bar{\Psi} \cdot \bar{\mathbf{x}}_i + b \leq k_1 + \xi_i, \quad y_i = -1 \tag{3}$$

where the ξ_i are slack variables, introduced to handle the non-separable case [5]. In the separable case, the SVM algorithm constructs that separating hyperplane for which the margin between the positive and negative examples in $\mathcal{H}$ is maximized. A test vector $\mathbf{x} \in \mathcal{L}$ is then assigned a class label $\{+1, -1\}$ depending on whether $\bar{\Psi} \cdot \Phi(\mathbf{x}) + b$ is greater or less than $(k_0 + k_1)/2$. A support vector $\mathbf{s} \in \mathcal{L}$ is defined as any training sample for which one of the equations (2) or (3) is an equality. (We name the support vectors $\mathbf{s}$ to distinguish them from the rest of the training data). $\bar{\Psi}$ is then given by

$$\bar{\Psi} = \sum_{a=1}^{N_S} \alpha_a y_a \Phi(\mathbf{s}_a) \tag{4}$$

where $\alpha_a \geq 0$ are the weights, determined during training, and $y_a \in \{+1, -1\}$ the class labels of the $\mathbf{s}_a$. Thus in order to classify a test point $\mathbf{x}$ one computes

$$\bar{\Psi} \cdot \bar{\mathbf{x}} = \sum_{a=1}^{N_S} \alpha_a y_a \bar{\mathbf{s}}_a \cdot \bar{\mathbf{x}} = \sum_{a=1}^{N_S} \alpha_a y_a K(\mathbf{s}_a, \mathbf{x}). \tag{5}$$

Consider now a set $\mathbf{z}_a \in \mathcal{L}, a = 1, \ldots, N_Z$ and corresponding weights $\gamma_a \in \mathbf{R}$ for which

$$\bar{\Psi}' \equiv \sum_{a=1}^{N_Z} \gamma_a \Phi(\mathbf{z}_a) \tag{6}$$

minimizes (for fixed N_Z) the distance measure

$$\rho = \|\bar{\Psi} - \bar{\Psi}'\|. \tag{7}$$

We call the $\{\gamma_a, \mathbf{z}_a\}, a = 1, \ldots, N_Z$ the *reduced set*. To classify a test point $\mathbf{x}$, the expansion in Equation (5) is replaced by the approximation

$$\bar{\Psi}' \cdot \bar{\mathbf{x}} = \sum_{a=1}^{N_Z} \gamma_a \bar{\mathbf{z}}_a \cdot \bar{\mathbf{x}} = \sum_{a=1}^{N_Z} \gamma_a K(\mathbf{z}_a, \mathbf{x}). \tag{8}$$

The goal is then to choose the smallest $N_Z \ll N_S$, and corresponding reduced set, such that any resulting loss in generalization performance remains acceptable. Clearly, by allowing $N_Z = N_S$, ρ can be made zero; there are non-trivial cases where $N_Z < N_S$ and $\rho = 0$ (Section 3). In those cases the reduced set leads to a reduction in the decision rule complexity with no loss in generalization performance. If for each N_Z one computes the corresponding reduced set, ρ may be viewed as a monotonic decreasing function of N_Z, and the generalization performance also becomes a function of N_Z. In this paper, we present only empirical results regarding the dependence of the generalization performance on N_Z.

We end this Section with some remarks on the mapping Φ. The image of Φ will not in general be a linear space. Φ will also in general not be surjective, and may not be one-to-one (for example, when K is a homogeneous polynomial of even degree). Further, Φ can map linearly dependent vectors in $\mathcal{L}$ onto linearly independent vectors in $\mathcal{H}$ (for example, when K is an inhomogeneous polynomial), or linearly independent vectors onto linearly dependent vectors ($K = 0$). In general one cannot scale the coefficients γ_a to unity by scaling $\mathbf{z}_a$, even when K is a homogeneous polynomial (for example, if K is homogeneous of even degree, the γ_a can be scaled to $\{+1, -1\}$, but not to unity).

3 EXACT SOLUTIONS

In this Section we consider the problem of computing the minimum of ρ analytically. We start with a simple but non-trivial case.

3.1 HOMOGENEOUS QUADRATIC POLYNOMIALS

For homogeneous degree two polynomials,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathcal{N}(\mathbf{x}_i \cdot \mathbf{x}_j)^2 \tag{9}$$

where $\mathcal{N}$ is a normalization factor. To simplify the exposition we start by computing the first order approximation, $N_Z = 1$. Introducing the symmetric tensor

$$\mathbf{S}_{\mu\nu} \equiv \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{s}_{i\mu} \mathbf{s}_{i\nu} \tag{10}$$

we find that $\rho = \|\bar{\Psi} - \gamma \bar{\mathbf{z}}\|$ is minimized for $\{\gamma, \mathbf{z}\}$ satisfying

$$\mathbf{S}_{\mu\nu} \mathbf{z}_\nu = \gamma \mathbf{z}^2 \mathbf{z}_\mu \tag{11}$$

(repeated indices are assumed summed). With this choice of $\{\gamma, \mathbf{z}\}$, ρ^2 becomes

$$\rho^2 = \mathbf{S}_{\mu\nu} \mathbf{S}^{\mu\nu} - \gamma^2 \mathbf{z}^4. \tag{12}$$

The largest drop in ρ is thus achieved when $\{\gamma, \mathbf{z}\}$ is chosen such that $\mathbf{z}$ is that eigenvector of $\mathbf{S}$ whose eigenvalue $\lambda = \gamma \mathbf{z}^2$ has largest absolute size. Note that we can choose $\gamma = sign\{\lambda\}$ and scale $\mathbf{z}$ so that $\mathbf{z}^2 = \lambda$.

Extending to order N_Z, it can similarly be shown that the $\mathbf{z}_i$ in the set $\{\gamma_i, \mathbf{z}_i\}$ that minimize

$$\rho = \|\bar{\Psi} - \sum_{a=1}^{N_Z} \gamma_a \bar{\mathbf{z}}_a\| \tag{13}$$

are eigenvectors of $\mathbf{S}$, each with eigenvalue $\gamma_i \|\mathbf{z}_i\|^2$. This gives

$$\rho^2 = \mathbf{S}_{\mu\nu} \mathbf{S}^{\mu\nu} - \sum_{a=1}^{N_Z} \gamma_a^2 \|\mathbf{z}_a\|^4 \tag{14}$$

and the drop in ρ is maximized if the $\mathbf{z}_a$ are chosen to be the first N_Z eigenvectors of $\mathbf{S}$, where the eigenvectors are ordered by absolute size of their eigenvalues. Note that, since $trace(\mathbf{S}^2)$ is the sum of the squared eigenvalues of $\mathbf{S}$, by choosing $N_Z = d_L$ the approximation becomes exact, i.e. $\rho = 0$. Since the number of support vectors N_S is often larger than d_L, this shows that the size of the reduced set can be smaller than the number of support vectors, with no loss in generalization performance.

In the general case, in order to compute the reduced set, ρ must be minimized over all $\{\gamma_a, \mathbf{z}_a\}, a = 1, \ldots, N_Z$ simultaneously. It is convenient to consider an incremental approach in which on the ith step, $\{\gamma_j, \mathbf{z}_j\}, j < i$ are held fixed while $\{\gamma_i, \mathbf{z}_i\}$ is computed. In the case of quadratic polynomials, the series of minima generated by the incremental approach also generates a minimum for the full problem. This result is particular to second degree polynomials and is a consequence of the fact that the $\mathbf{z}_i$ are orthogonal (or can be so chosen).

3.1.1 Experiments

Table 1 shows the reduced set size N_Z necessary to attain a number of errors E_Z on the test set, where E_Z differs from the number of errors E_S found using the full set of support vectors by at most one error, for a quadratic polynomial SVM trained on the postal set. Clearly, in the quadratic case, the reduced set can offer a significant reduction in complexity with little loss in accuracy. Note also that many digits have numbers of support vectors larger than $d_L = 256$, presenting in this case the opportunity for a speed up with no loss in accuracy.

Table 1: Reduced Set Generalization Performance for the Quadratic Case.

	Support Vectors		Reduced Set	
Digit	N_S	E_S	N_Z	E_Z
0	292	15	10	16
1	95	9	6	9
2	415	28	22	29
3	403	26	14	27
4	375	35	14	34
5	421	26	18	27
6	261	13	12	14
7	228	18	10	19
8	446	33	24	33
9	330	20	20	21

3.2 GENERAL KERNELS

To apply the reduced set method to an arbitrary support vector machine, the above analysis must be extended for a general kernel. For example, for the homogeneous polynomial $K(\mathbf{x}_1, \mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1 \cdot \mathbf{x}_2)^n$, setting $\partial\rho/\partial\mathbf{z}_{1a_1} = 0$ to find the first pair $\{\gamma_1, \mathbf{z}_1\}$ in the incremental approach gives an equation analogous to Equation (11):

$$\mathbf{S}_{\mu_1\mu_2\cdots\mu_n}\mathbf{z}_{1\mu_2}\mathbf{z}_{1\mu_3}\cdots\mathbf{z}_{1\mu_n} = \gamma_1\|\mathbf{z}_1\|^{(2n-2)}\mathbf{z}_{1\mu_1} \quad (15)$$

where

$$\mathbf{S}_{\mu_1\mu_2\cdots\mu_n} \equiv \sum_{m=1}^{N_S}\alpha_m y_m \mathbf{s}_{m\mu_1}\mathbf{s}_{m\mu_2}\cdots\mathbf{s}_{m\mu_n} \quad (16)$$

In this case, varying ρ with respect to γ gives no new conditions. Having solved Equation (15) for the first order solution $\{\gamma_1, \mathbf{z}_1\}$, ρ^2 becomes

$$\rho^2 = \mathbf{S}_{\mu_1\mu_2\cdots\mu_n}\mathbf{S}^{\mu_1\mu_2\cdots\mu_n} - \gamma_1^2\|\mathbf{z}_1\|^{2n}. \quad (17)$$

One can then define

$$\tilde{\mathbf{S}}_{\mu_1\mu_2\cdots\mu_n} \equiv \mathbf{S}_{\mu_1\mu_2\cdots\mu_n} - \gamma_1\mathbf{z}_{1\mu_1}\mathbf{z}_{1\mu_2}\cdots\mathbf{z}_{1\mu_n} \quad (18)$$

in terms of which the incremental equation for the second order solution $\mathbf{z}_2$ takes the form of Equation (15), with $\mathbf{S}$, $\mathbf{z}_1$ and γ_1 replaced by $\tilde{\mathbf{S}}$, $\mathbf{z}_2$ and γ_2, respectively. (Note that for polynomials of degree greater than 2, the $\mathbf{z}_a$ will not in general be orthogonal). However, these are only the incremental solutions: one still needs to solve the coupled equations where all $\{\gamma_a, \mathbf{z}_a\}$ are allowed to vary simultaneously. Moreover, these equations will have multiple solutions, most of which will lead to local minima in ρ. Furthermore, other choices of K will lead to other fixed point equations.

For the purposes of the work described in this paper, we therefore decided to take a computational approach. We found that, while solutions to Equation (15) could be found by iterating (i.e. by starting with arbitrary $\mathbf{z}$, computing a new $\mathbf{z}$ using Equation (15), and repeating), the method described in the next Section proved more flexible and powerful.

4 UNCONSTRAINED OPTIMIZATION APPROACH

Provided the kernel K has first derivatives defined, the gradients of the objective function $F \equiv \rho^2/2$ with respect to the unknowns $\{\gamma_i, \mathbf{z}_i\}$ can be computed. For example, assuming that $K(\mathbf{s}_m, \mathbf{s}_n)$ is a function of the scalar $\mathbf{s}_m \cdot \mathbf{s}_n$:

$$\frac{\partial F}{\partial \gamma_k} = -\sum_{m=1}^{N_S}\alpha_m y_m K(\mathbf{s}_m\cdot\mathbf{z}_k) + \sum_{j=1}^{N_Z}\gamma_j K(\mathbf{z}_j\cdot\mathbf{z}_k) \quad (19)$$

$$\begin{aligned}\frac{\partial F}{\partial \mathbf{z}_{k\mu}} &= -\sum_{m=1}^{N_S}\gamma_k\alpha_m y_m K'(\mathbf{s}_m\cdot\mathbf{z}_k)\mathbf{s}_{m\mu} \quad (20)\\ &\quad + \sum_{j=1}^{N_Z}\gamma_j\gamma_k K'(\mathbf{z}_j\cdot\mathbf{z}_k)\mathbf{z}_{j\mu}\end{aligned}$$

A (possibly local) minimum can then be found using unconstrained optimization techniques.

4.1 THE ALGORITHM

We start by summarizing the algorithm used. First, the desired order of approximation, N_Z, is chosen. Let $\mathbf{X}_i \equiv \{\gamma_i, \mathbf{z}_i\}$. We used a two-phase approach. In phase 1, the $\mathbf{X}_i$ are computed incrementally, keeping all $\mathbf{X}_j, j < i$ fixed. In phase 2, all $\mathbf{X}_i$ are allowed to vary.

4.1.1 Phase 1

The gradient in Equation (20) is zero if γ_k is zero. This fact can lead to severe numerical instabilities. In order to circumvent this problem, phase 1 relies on a simple "level crossing" theorem. First, γ_i is initialized to $+1$ or -1; $\mathbf{z}_i$ is initialized with random values. $\mathbf{z}_i$ is then allowed to vary, while keeping γ_i fixed. The optimal value for γ_i, given that $\mathbf{z}_i, \mathbf{X}_j, j < i$ are fixed, can then be computed analytically. F is then minimized with respect to both $\mathbf{z}_i$ and γ_i simultaneously. Finally, the optimal γ_j for all $j \le i$ can be computed analytically, and are given by $\Gamma = \mathbf{Z}^{-1}\Delta$, where vectors Δ, Γ and $\mathbf{Z}$ are are given by:

$$\Gamma_j \equiv \gamma_j, \quad (21)$$

$$\Delta_j \equiv \sum_{a=1}^{N_S}\alpha_a y_a K(\mathbf{s}_a, \mathbf{z}_j) \quad (22)$$

and

$$\mathbf{Z}_{jk} \equiv K(\mathbf{z}_j, \mathbf{z}_k). \quad (23)$$

Since $\mathbf{Z}$ is positive definite and symmetric, it can be inverted efficiently using Choleski decomposition.

Numerical instabilities are avoided by preventing γ_i from approaching zero. The above algorithm ensures this automatically: if the first step, in which $\mathbf{z}_i$ is varied while γ_i is kept fixed, results in a decrease in the objective function F, then when γ_i is subsequently allowed to vary, it cannot pass through zero, because doing so would require an increase in F.

Phase 1 is repeated several (T) times, with different initial values for the $\mathbf{X}_i$. T is determined heuristically from the number M of different minima found. For our data, we found M was usually 2 or 3, and we thus chose $T = 10$. M was sufficiently small that more sophisticated techniques (for example, simulated annealing) were not pursued.

4.1.2 Phase 2

In phase 2, all vectors $\mathbf{X}_i$ found in phase 1 are concatenated into a single vector, and the unconstrained minimization process then applied again. We have found that phase 2 often results in roughly a factor of two further reduction in the objective function F.

The following first order unconstrained optimization method was used for both phases. The search direction is found using conjugate gradients. Bracketing points x_1, x_2 and x_3 are found along the search direction such that $F(x_1) > F(x_2) < F(x_3)$. The bracket is then balanced [13]. The minimum of the quadratic fit through these three points is then used as the starting point for the next iteration. The conjugate gradient process is restarted after a fixed, chosen number of iterations, and the whole process stops when the rate of decrease of F falls below a threshold. We checked that this general approach gave the same results as the analytic approach when applied to the quadratic polynomial case.

4.2 EXPERIMENTS

The above approach was applied to the SVM that gave the best performance on the postal set, which was a degree 3 inhomogeneous polynomial machine [2]. The order of approximation, N_Z, was chosen to give a factor of ten speed up in test phase for each two-class classifier. The results are given in Table 2. The reduced set method achieved the speed up with essentially no loss in accuracy. Using the ten classifiers together as a ten-class classifier [2; 5] gave 4.2% error using the full support set, as opposed to 4.3% using the reduced set. Note that for the combined case, the reduced set gives only a factor of six speed up, since different two class classifiers have some support vectors in common, allowing the possibility of caching. To address the question as to whether these techniques can be scaled up to larger problems, we repeated the study for a two-class classifier separating digit 0 from all other digits for the NIST set (60,000 training, 10,000 test patterns). This classifier was also chosen to be that which gave best accuracy using the full support set: a degree 4 polynomial. The full set of 1,273 support vectors gave 19 test errors, while a reduced set of size 127 gave 20 test errors.

Table 2: Postal set, degree 3 polynomial. The third and fifth columns give the number of errors in test mode of the support vector and reduced set systems respectively.

	Support Vectors		Reduced Set	
Digit	N_S	E_s	N_Z	E_z
0	272	13	27	13
1	109	9	11	10
2	380	26	38	26
3	418	20	42	20
4	392	34	39	32
5	397	21	40	22
6	257	11	26	11
7	214	14	21	13
8	463	26	46	28
9	387	13	39	13
Totals:	3289	187	329	188

4.2.1 Comparison with Other Techniques

In terms of combined speed in test phase and generalization performance, the LeNet series is among the best performing systems [6; 7]. For the postal set, the optimal LeNet architecture ("LeNet 1") requires approximately 120,000 multiply-adds for a forward pass. This number can be reduced to approximately 80,000 by some architecture-specific optimization [8]. The generalization performance is similar to that found above (4.3% for the ten-class case) [8]. Thus the factor of ten speed up described above gives the SVM approach a similar test speed to that of the LeNet neural network, for this data set.

Our experiment on the NIST set was designed to check that the reduced set method can be scaled up to larger problems. Even with a factor of ten speed up, the resulting SVM is still considerably slower than the best performing LeNet ("LeNet5") [7]. It is an open question as to how much further reduction in the complexity of the SVM decision rule could be achieved with no loss in generalization performance.

Table 3: Size of Reduced Set N_Z needed to attain various levels of generalization performance, using Phase 1 only. Using the full set of $N_S = 3,289$ support vectors gives 4.2% raw error. Using the reduced set gives a speed up of a factor of N_S/N_Z. Note that here, N_Z and N_S are summed over all digits.

N_Z	$100 * N_Z/N_S$	Error Rate
34	1.0	24.1
65	2.0	16.9
99	3.0	7.1
132	4.0	7.8
165	5.0	6.4
198	6.0	5.7
230	7.0	5.5
263	8.0	5.7
296	9.0	5.3
329	10.0	5.2
363	11.0	5.1
396	12.0	4.6
426	13.0	4.5
461	14.0	4.5
494	15.0	5.0
527	16.0	4.7
560	17.0	4.8
592	18.0	4.8
625	19.0	4.4
657	20.0	4.3
690	21.0	4.5
724	22.0	4.6
755	23.0	4.2
788	24.0	4.2

4.2.2 Reduced Set Size versus Generalization Performance

An interesting question is how generalization performance varies with the size of the reduced set. In particular, since the number of parameters in the reduced set classifier is less than that of the support vector classifier, one might suspect that the method may provide a means of capacity control. However, an effective form of capacity control must control both the empirical risk and the VC dimension of the set of decision functions. To gain an empirical view, we computed reduced sets of several different sizes for the postal set, and measured the generalization performance of each. However, the reduced set was computed for the incremental approach only (phase 1); the error rates quoted here could be further reduced by applying both phases. Results are shown in Table (3), which shows the generalization performance of the combined 10-class classifiers for the postal set. The error rate quoted is that at zero rejection.

5 CONCLUSIONS

We have introduced the reduced set method as a means of approximating the vector $\bar{\Psi} \in \mathcal{H}$ appearing in the decision rule of a Support Vector Machine, and have shown that, for the case of OCR digit recognition, using a reduced set can give at least a factor of ten speed up (over using the full support set) with essentially no loss in accuracy. In the approach described, the size of the reduced set is first specified, and the resulting accuracy loss, if any, is determined experimentally. The support vector method is extremely general and has many applications; we expect the approach described in this paper to be applicable in these different areas. By choosing N_Z, the approach allows direct control over the speed/accuracy trade-off of Support Vector Machines.

Acknowledgements

I wish to thank V. Vapnik for valuable discussions and for commenting on the manuscript. I also wish to thank C. Stenard (Advanced Information Systems Engineering group, Bell Laboratories, Lucent Technologies) and B. Yoon (ARPA) for their support of this work. This work was funded under ARPA contract N00014-94-C-0186.

References

[1] V. Vapnik, *Estimation of Dependencies Based on Empirical Data*, Springer Verlag, 1982.

[2] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, 1995.

[3] Boser, B.E., Guyon, I.M., and Vapnik, V., *A training algorithm for optimal margin classifiers*, Fifth Annual Workshop on Computational Learning Theory, Pittsburgh ACM 144-152, 1992.

[4] B. Schölkopf, C.J.C. Burges, and V. Vapnik, *Extracting Support Data for a Given Task*, Proceedings of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, 1995.

[5] C. Cortes and V. Vapnik, *Support Vector Networks*, Machine Learning, Vol 20, pp 1-25, 1995.

[6] L. Bottou, C. Cortes, H. Drucker, L.D. Jackel, Y. LeCun, U.A. Müller, E. Säckinger, P. Simard, and V. Vapnik, *Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition*, Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 2, IEEE Computer Society Press, Los Alamos, CA, pp. 77-83, 1994.

[7] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U.

Müller, E. Säckinger, P. Simard, and V. Vapnik, *Comparison of Learning Algorithms for Handwritten Digit Recognition*, International Conference on Artificial Neural Networks, Ed. F. Fogelman, P. Gallinari, pp. 53-60, 1995.

[8] Y. LeCun, Private Communication.

[9] M. Schmidt, BBN, Private Communication.

[10] V. Blanz, B. Schölkopf, H. Bülthoff, C.J.C. Burges, V. Vapnik and T. Vetter, *Comparison of View-Based Object Recognition Algorithms Using Realistic 3D Models*, submitted to International Conference on Artificial Neural Networks, 1996.

[11] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, *Backpropagation Applied to Handwritten ZIP Code Recognition*, Neural Computation, 1, 1989, pp. 541-551.

[12] R.A. Wilkinson, J. Geist, S. Janet, P.J. Grother, C.J.C. Burges, R. Creecy, R. Hammond, J.J. Hull, N.J. Larsen, T.P. Vogl and C.L. Wilson, *The First Census Optical Character Recognition System Conference*, US Department of Commerce, NIST, August 1992.

[13] W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery, *Numerical Recipes in C*, Second Edition, Cambridge University Press, 1992.