A Training Algorithm for Optimal Margin Classifiers

Bernhard E. Boser[*]
EECS Department
University of California
Berkeley, CA 94720
boser@eecs.berkeley.edu

Isabelle M. Guyon
AT&T Bell Laboratories
50 Fremont Street, 6th Floor
San Francisco, CA 94105
isabelle@neural.att.com

Vladimir N. Vapnik
AT&T Bell Laboratories
Crawford Corner Road
Holmdel, NJ 07733
vlad@neural.att.com

Abstract

A training algorithm that maximizes the margin between the training patterns and the decision boundary is presented. The technique is applicable to a wide variety of classifiaction functions, including Perceptrons, polynomials, and Radial Basis Functions. The effective number of parameters is adjusted automatically to match the complexity of the problem. The solution is expressed as a linear combination of supporting patterns. These are the subset of training patterns that are closest to the decision boundary. Bounds on the generalization performance based on the leave-one-out method and the VC-dimension are given. Experimental results on optical character recognition problems demonstrate the good generalization obtained when compared with other learning algorithms.

1 INTRODUCTION

Good generalization performance of pattern classifiers is achieved when the capacity of the classification function is matched to the size of the training set. Classifiers with a large number of adjustable parameters and therefore large capacity likely learn the training set without error, but exhibit poor generalization. Conversely, a classifier with insufficient capacity might not be able to learn the task at all. In between, there is an optimal capacity of the classifier which minimizes the expected generalization error for a given amount of training data. Both experimental evidence and theoretical studies [GBD92, Moo92, GVB$^+$92, Vap82, BH89, TLS89, Mac92] link the generalization of a classifier to the error on the training examples and the complexity of the classifier. Methods such as structural risk minimization [Vap82] vary the complexity of the classification function in order to optimize the generalization.

In this paper we describe a training algorithm that automatically tunes the capacity of the classification function by maximizing the margin between training examples and class boundary [KM87], optionally after removing some atypical or meaningless examples from the training data. The resulting classification function depends only on so-called supporting patterns [Vap82]. These are those training examples that are closest to the decision boundary and are usually a small subset of the training data.

It will be demonstrated that maximizing the margin amounts to minimizing the maximum loss, as opposed to some average quantity such as the mean squared error. This has several desirable consequences. The resulting classification rule achieves an errorless separation of the training data if possible. Outliers or meaningless patterns are identified by the algorithm and can therefore be eliminated easily with or without supervision. This contrasts classifiers based on minimizing the mean squared error, which quietly ignore atypical patterns. Another advantage of maximum margin classifiers is that the sensitivity of the classifier to limited computational accuracy is minimal compared to other separations with smaller margin. In analogy to [Vap82, HLW88] a bound on the generalization performance is obtained with the "leave-one-out" method. For the maximum margin classifier it is the ratio of the number of linearly independent supporting patterns to the number of training examples. This bound is tighter than a bound based on the capacity of the classifier family.

The proposed algorithm operates with a large class of decision functions that are linear in their parameters but not restricted to linear dependences in the input components. Perceptrons [Ros62], polynomial classifiers, neural networks with one hidden layer, and Radial Basis Function (RBF) or potential function classifiers [ABR64, BL88, MD89] fall into this class. As pointed out by several authors [ABR64, DH73, PG90], Percep-

[*] Part of this work was performed while B. Boser was with AT&T Bell Laboratories. He is now at the University of California, Berkeley.

trons have a dual kernel representation implementing the same decision function. The optimal margin algorithm exploits this duality both for improved efficiency and flexibility. In the dual space the decision function is expressed as a linear combination of basis functions parametrized by the supporting patterns. The supporting patterns correspond to the class centers of RBF classifiers and are chosen automatically by the maximum margin training procedure. In the case of polynomial classifiers, the Perceptron representation involves an untractable number of parameters. This problem is overcome in the dual space representation, where the classification rule is a weighted sum of a kernel function [Pog75] for each supporting pattern. High order polynomial classifiers with very large training sets can therefore be handled efficiently with the proposed algorithm.

The training algorithm is described in Section 2. Section 3 summarizes important properties of optimal margin classifiers. Experimental results are reported in Section 4.

2 MAXIMUM MARGIN TRAINING ALGORITHM

The maximum margin training algorithm finds a decision function for pattern vectors $\mathbf{x}$ of dimension n belonging to either of two classes A and B. The input to the training algorithm is a set of p examples $\mathbf{x}_i$ with labels y_i:

$$(\mathbf{x}_1, y_1),\ (\mathbf{x}_2, y_2),\ (\mathbf{x}_3, y_3),\ \ldots,\ (\mathbf{x}_p, y_p) \tag{1}$$
$$\text{where } \begin{cases} y_k = 1 & \text{if } \mathbf{x}_k \in \text{class A} \\ y_k = -1 & \text{if } \mathbf{x}_k \in \text{class B.} \end{cases}$$

From these training examples the algorithm finds the parameters of the decision function $D(\mathbf{x})$ during a learning phase. After training, the classification of unknown patterns is predicted according to the following rule:

$$\begin{array}{ll} \mathbf{x} \in \text{A} & \text{if } D(\mathbf{x}) > 0 \\ \mathbf{x} \in \text{B} & \text{otherwise.} \end{array} \tag{2}$$

The decision functions must be linear in their parameters but are not restricted to linear dependences of $\mathbf{x}$. These functions can be expressed either in direct, or in dual space. The direct space notation is identical to the Perceptron decision function [Ros62]:

$$D(\mathbf{x}) = \sum_{i=1}^{N} w_i \varphi_i(\mathbf{x}) + b. \tag{3}$$

In this equation the φ_i are predefined functions of $\mathbf{x}$, and the w_i and b are the adjustable parameters of the decision function. Polynomial classifiers are a special case of Perceptrons for which $\varphi_i(\mathbf{x})$ are products of components of $\mathbf{x}$.

In the dual space, the decision functions are of the form

$$D(\mathbf{x}) = \sum_{k=1}^{p} \alpha_k K(\mathbf{x}_k, \mathbf{x}) + b, \tag{4}$$

The coefficients α_k are the parameters to be adjusted and the $\mathbf{x}_k$ are the training patterns. The function K is a predefined kernel, for example a potential function [ABR64] or any Radial Basis Function [BL88, MD89]. Under certain conditions [CH53], symmetric kernels possess finite or infinite series expansions of the form

$$K(\mathbf{x}, \mathbf{x}') = \sum_i \varphi_i(\mathbf{x})\, \varphi_i(\mathbf{x}'). \tag{5}$$

In particular, the kernel $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + 1)^q$ corresponds to a polynomial expansion $\boldsymbol{\varphi}(\mathbf{x})$ of order q [Pog75].

Provided that the expansion stated in equation 5 exists, equations 3 and 4 are dual representations of the same decision function and

$$w_i = \sum_{k=1}^{p} \alpha_k \varphi_i(\mathbf{x}_k). \tag{6}$$

The parameters w_i are called direct parameters, and the α_k are referred to as dual parameters.

The proposed training algorithm is based on the "generalized portrait" method described in [Vap82] that constructs separating hyperplanes with maximum margin. Here this algorithm is extended to train classifiers linear in their parameters. First, the margin between the class boundary and the training patterns is formulated in the direct space. This problem description is then transformed into the dual space by means of the Lagrangian. The resulting problem is that of maximizing a quadratic form with constraints and is amenable to efficient numeric optimization algorithms [Lue84].

2.1 MAXIMIZING THE MARGIN IN THE DIRECT SPACE

In the direct space the decision function is

$$D(\mathbf{x}) = \mathbf{w} \cdot \boldsymbol{\varphi}(\mathbf{x}) + b, \tag{7}$$

where $\mathbf{w}$ and $\boldsymbol{\varphi}(\mathbf{x})$ are N dimensional vectors and b is a bias. It defines a separating hyperplane in $\boldsymbol{\varphi}$-space. The distance between this hyperplane and pattern $\mathbf{x}$ is $D(\mathbf{x})/\|\mathbf{w}\|$ (Figure 1). Assuming that a separation of the training set with margin M between the class boundary and the training patterns exists, all training patterns fulfill the following inequality:

$$\frac{y_k D(\mathbf{x}_k)}{\|\mathbf{w}\|} \geq M. \tag{8}$$

The objective of the training algorithm is to find the parameter vector $\mathbf{w}$ that maximizes M:

$$M^* = \max_{\mathbf{w}, \|\mathbf{w}\|=1} M \tag{9}$$
$$\text{subject to } \ y_k D(\mathbf{x}_k) \geq M, \qquad k = 1, 2, \ldots, p.$$

The bound M^* is attained for those patterns satisfying

$$\min_k y_k D(\mathbf{x}_k) = M^*. \tag{10}$$

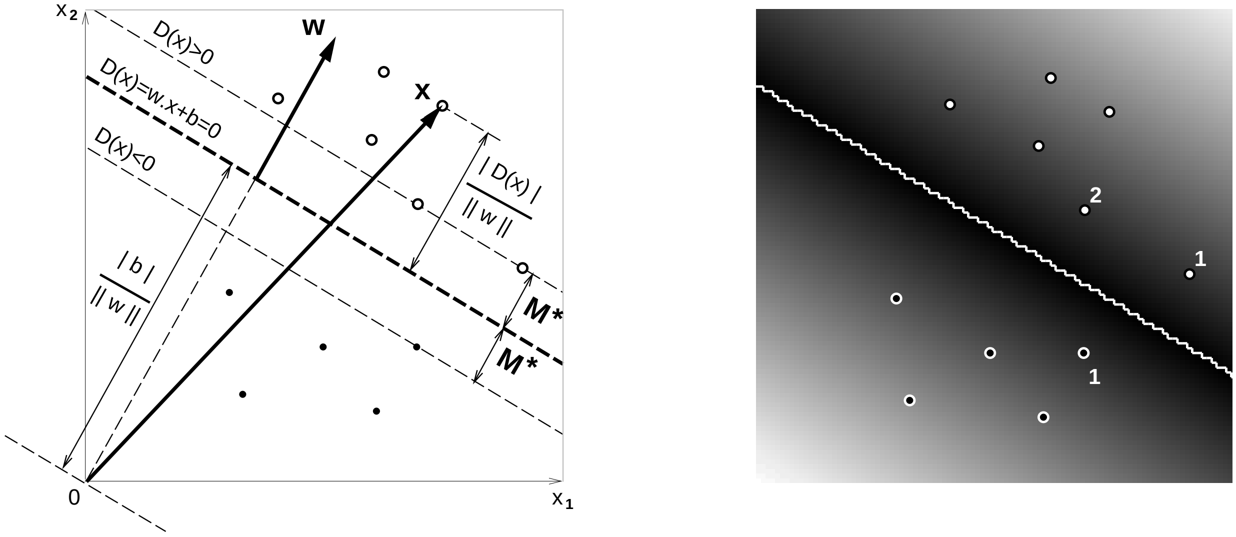

Figure 1: Maximum margin linear decision function $D(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ ($\boldsymbol{\varphi} = \mathbf{x}$). The gray levels encode the absolute value of the decision function (solid black corresponds to $D(\mathbf{x}) = 0$). The numbers indicate the supporting patterns.

These patterns are called the supporting patterns of the decision boundary.

A decision function with maximum margin is illustrated in figure 1. The problem of finding a hyperplane in $\boldsymbol{\varphi}$-space with maximum margin is therefore a minimax problem:

$$\max_{\mathbf{w}, \|\mathbf{w}\|=1} \min_{k} y_k D(\mathbf{x}_k). \tag{11}$$

The norm of the parameter vector in equations 9 and 11 is fixed to pick one of an infinite number of possible solutions that differ only in scaling. Instead of fixing the norm of $\mathbf{w}$ to take care of the scaling problem, the product of the margin M and the norm of a weight vector $\mathbf{w}$ can be fixed.

$$M\|\mathbf{w}\| = 1. \tag{12}$$

Thus, maximizing the margin M is equivalent to minimizing the norm $\|\mathbf{w}\|$.[1] Then the problem of finding a maximum margin separating hyperplane $\mathbf{w}^*$ stated in 9 reduces to solving the following quadratic problem:

$$\begin{aligned} &\min_{\mathbf{w}} \|\mathbf{w}\|^2 && (13) \\ &\text{under conditions } \; y_k D(\mathbf{x}_k) \geq 1, \quad k = 1, 2, \ldots, p. \end{aligned}$$

The maximum margin is $M^* = 1/\|\mathbf{w}^*\|$.

In principle the problem stated in 13 can be solved directly with numerical techniques. However, this approach is impractical when the dimensionality of the $\boldsymbol{\varphi}$-space is large or infinite. Moreover, no information is gained about the supporting patterns.

2.2 MAXIMIZING THE MARGIN IN THE DUAL SPACE

Problem 13 can be transformed into the dual space by means of the Lagrangian [Lue84]

$$\begin{aligned} &L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{k=1}^{p} \alpha_k \left[y_k D(\mathbf{x}_k) - 1 \right] (14) \\ &\text{subject to } \; \alpha_k \geq 0, \quad k = 1, 2, \ldots, p. \end{aligned}$$

The factors α_k are called Lagrange multipliers or Kühn-Tucker coefficients and satisfy the conditions

$$\alpha_k \left(y_k D(\mathbf{x}_k) - 1 \right) = 0, \quad k = 1, 2, \ldots, p. \tag{15}$$

The factor one half has been included for cosmetic reasons; it does not change the solution.

The optimization problem 13 is equivalent to searching a saddle point of the function $L(\mathbf{w}, b, \boldsymbol{\alpha})$. This saddle point is a the minimum of $L(\mathbf{w}, b, \boldsymbol{\alpha})$ with respect to $\mathbf{w}$, and a maximum with respect to $\boldsymbol{\alpha}$ ($\alpha_k \geq 0$). At the solution, the following necessary condition is met:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w}^* - \sum_{k=1}^{p} \alpha_k^* y_k \boldsymbol{\varphi}_k = 0,$$

hence

$$\mathbf{w}^* = \sum_{k=1}^{P} \alpha_k^* y_k \boldsymbol{\varphi}_k. \tag{16}$$

The patterns which satisfy $y_k D(\mathbf{x}_k) = 1$ are the supporting patterns. According to equation 16, the vector $\mathbf{w}^*$ that specifies the hyperplane with maximum margin is a linear combination of only the supporting patterns, which are those patterns for which $\alpha_k^* \neq 0$. Usually the number of supporting patterns is much smaller than the number p of patterns in the training set.

[1] If the training data is not linearly separable the maximum margin may be negative. In this case, $M\|\mathbf{w}\| = -1$ is imposed. Maximizing the margin is then equivalent to maximizing $\|\mathbf{w}\|$.

The dependence of the Lagrangian $L(\mathbf{w}, b, \boldsymbol{\alpha})$ on the weight vector $\mathbf{w}$ is removed by substituting the expansion of $\mathbf{w}^*$ given by equation 16 for $\mathbf{w}$. Further transformations using 3 and 5 result in a Lagrangian which is a function of the parameters $\boldsymbol{\alpha}$ and the bias b only:

$$J(\boldsymbol{\alpha}, b) = \sum_{k=1}^{p} \alpha_k (1 - b y_k) - \frac{1}{2} \boldsymbol{\alpha} \cdot \boldsymbol{H} \cdot \boldsymbol{\alpha}, \quad (17)$$

$$\text{subject to} \quad \alpha_k \geq 0, \qquad k = 1, 2, \ldots, p.$$

Here $\boldsymbol{H}$ is a square matrix of size $p \times p$ with elements

$$H_{kl} = y_k y_l K(\mathbf{x}_k, \mathbf{x}_l).$$

In order for a unique solution to exist, $\boldsymbol{H}$ must be positive definite. For fixed bias b, the solution $\boldsymbol{\alpha}^*$ is obtained by maximizing $J(\boldsymbol{\alpha}, b)$ under the conditions $\alpha_k \geq 0$. Based on equations 7 and 16, the resulting decision function is of the form

$$\begin{aligned} D(\mathbf{x}) &= \mathbf{w}^* \cdot \boldsymbol{\varphi}(\mathbf{x}) + b & (18) \\ &= \sum_k y_k \alpha_k^* K(\mathbf{x}_k, \mathbf{x}) + b, \qquad \alpha_k^* \geq 0, \end{aligned}$$

where only the supporting patterns appear in the sum with nonzero weight.

The choice of the bias b gives rise to several variants of the algorithm. The two considered here are

1. The bias can be fixed a priori and not subjected to training. This corresponds to the "Generalized Portrait Technique" described in [Vap82].
2. The cost function 17 can be optimized with respect to $\mathbf{w}$ and b. This approach gives the largest possible margin M^* in $\boldsymbol{\varphi}$-space [VC74].

In both cases the solution is found with standard nonlinear optimization algorithms for quadratic forms with linear constraints [Lue84, Loo72]. The second approach gives the largest possible margin. There is no guarantee, however, that this solution exhibits also the best generalization performance.

A strategy to optimize the margin with respect to both $\mathbf{w}$ and b is described in [Vap82]. It solves problem 17 for differences of pattern vectors to obtain $\boldsymbol{\alpha}^*$ independent of the bias, which is computed subsequently. The margin in $\boldsymbol{\varphi}$-space is maximized when the decision boundary is halfway between the two classes. Hence the bias b^* is obtained by applying 18 to two arbitrary supporting patterns $\mathbf{x}_A \in$ class A and $\mathbf{x}_B \in$ class B and taking into account that $D(\mathbf{x}_A) = 1$ and $D(\mathbf{x}_B) = -1$.

$$\begin{aligned} b^* &= -\frac{1}{2} (\mathbf{w}^* \cdot \boldsymbol{\varphi}(\mathbf{x}_A) + \mathbf{w}^* \cdot \boldsymbol{\varphi}(\mathbf{x}_B)) & (19) \\ &= -\frac{1}{2} \sum_{k=1}^{p} y_k \alpha_k^* \left[K(\mathbf{x}_A, \mathbf{x}_k) + K(\mathbf{x}_B, \mathbf{x}_k) \right]. \end{aligned}$$

The dimension of problem 17 equals the size of the training set, p. To avoid the need to solve a dual problem of exceedingly large dimensionality, the training data is divided into chunks that are processed iteratively [Vap82]. The maximum margin hypersurface is constructed for the first chunk and a new training set is formed consisting of the supporting patterns from the solution and those patterns $\mathbf{x}_k$ in the second chunk of the training set for which $y_k D(\mathbf{x}_k) < 1 - \epsilon$. A new classifier is trained and used to construct a training set consisting of supporting patterns and examples from the first three chunks which satisfy $y_k D(\mathbf{x}_k) < 1 - \epsilon$. This process is repeated until the entire training set is separated.

3 PROPERTIES OF THE ALGORITHM

In this Section, we highlight some important aspects of the optimal margin training algorithm. The description is split into a discussion of the qualities of the resulting classifier, and computational considerations. Classification performance advantages over other techniques will be illustrated in the Section on experimental results.

3.1 PROPERTIES OF THE SOLUTION

Since maximizing the margin between the decision boundary and the training patterns is equivalent to maximizing a quadratic form in the positive quadrant, there are no local minima and the solution is always unique if $\boldsymbol{H}$ has full rank. At the optimum

$$J(\boldsymbol{\alpha}^*) = \frac{1}{2} \|\mathbf{w}^*\|^2 = \frac{1}{2\,(M^*)^2} = \frac{1}{2} \sum_{k=1}^{p} \alpha_k^*. \qquad (20)$$

The uniqueness of the solution is a consequence of the maximum margin cost function and represents an important advantage over other algorithms for which the solution depends on the initial conditions or other parameters that are difficult to control.

Another benefit of the maximum margin objective is its insensitivity to small changes of the parameters $\mathbf{w}$ or $\boldsymbol{\alpha}$. Since the decision function $D(\mathbf{x})$ is a linear function of $\mathbf{w}$ in the direct, and of $\boldsymbol{\alpha}$ in the dual space, the probability of misclassifications due to parameter variations of the components of these vectors is minimized for maximum margin. The robustness of the solution—and potentially its generalization performance—can be increased further by omitting some supporting patterns from the solution. Equation 20 indicates that the largest increase in the maximum margin M^* occurs when the supporting patterns with largest α_k are eliminated. The elimination can be performed automatically or with assistance from a supervisor. This feature gives rise to other important uses of the optimum margin algorithm in database cleaning applications [MGB+92].

Figure 2 compares the decision boundary for a maximum margin and mean squared error (MSE) cost functions. Unlike the MSE based decision function which simply ignores the outlier, optimal margin classifiers are very sensitive to atypical patterns that are close to the

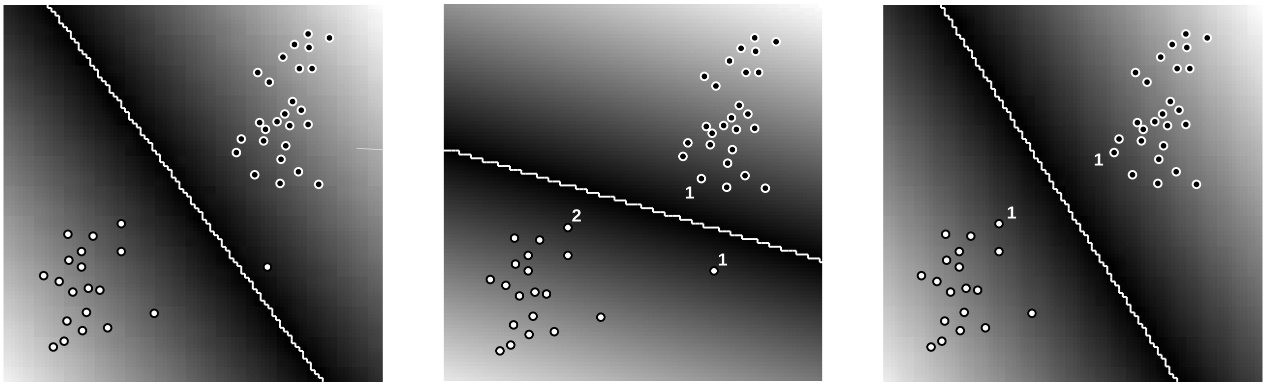

Figure 2: Linear decision boundary for MSE (left) and maximum margin cost functions (middle, right) in the presence of an outlier. In the rightmost picture the outlier has been removed. The numbers reflect the ranking of supporting patterns according to the magnitude of their Lagrange coefficient α_k for each class individually.

decision boundary. These examples are readily identified as those with the largest α_k and can be eliminated either automatically or with supervision. Hence, optimal margin classifiers give complete control over the handling of outliers, as opposed to quietly ignoring them.

The optimum margin algorithm performs automatic capacity tuning of the decision function to achieve good generalization. An estimate for an upper bound of the generalization error is obtained with the "leave-one-out" method: A pattern $\mathbf{x}_k$ is removed from the training set. A classifier is then trained on the remaining patterns and tested on $\mathbf{x}_k$. This process is repeated for all p training patterns. The generalization error is estimated by the ratio of misclassified patterns over p. For a maximum margin classifier, two cases arise: If $\mathbf{x}_k$ is not a supporting pattern, the decision boundary is unchanged and $\mathbf{x}_k$ will be classified correctly. If $\mathbf{x}_k$ is a supporting pattern, two cases are possible:

1. The pattern $\mathbf{x}_k$ is linearly dependent on the other supporting patterns. In this case it will be classified correctly.
2. $\mathbf{x}_k$ is linearly independent from the other supporting patterns. In this case the outcome is uncertain. In the worst case m' linearly independent supporting patterns are misclassified when they are omitted from the training data.

Hence the frequency of errors obtained by this method is at most m'/p, and has no direct relationship with the number of adjustable parameters. The number of linearly independent supporting patterns m' itself is bounded by $\min(N, p)$. This suggests that the number of supporting patterns is related to an effective capacity of the classifier that is usually much smaller than the VC-dimension, $N + 1$ [Vap82, HLW88].

In polynomial classifiers, for example, $N \approx n^q$, where n is the dimension of $\mathbf{x}$-space and q is the order of the polynomial. In practice, $m \leq p \ll N$, i. e. the number of supporting patterns is much smaller than the dimension of the $\boldsymbol{\varphi}$-space. The capacity tuning realized by the maximum margin algorithm is essential to get generalization with high-order polynomial classifiers.

3.2 COMPUTATIONAL CONSIDERATIONS

Speed and convergence are important practical considerations of classification algorithms. The benefit of the dual space representation to reduce the number of computations required for example for polynomial classifiers has been pointed out already. In the dual space, each evaluation of the decision function $D(\mathbf{x})$ requires m evaluations of the kernel function $K(\mathbf{x}_k, \mathbf{x})$ and forming the weighted sum of the results. This number can be further reduced through the use of appropriate search techniques which omit evaluations of K that yield negligible contributions to $D(\mathbf{x})$ [Omo91].

Typically, the training time for a separating surface from a database with several thousand examples is a few minutes on a workstation, when an efficient optimization algorithm is used. All experiments reported in the next section on a database with 7300 training examples took less than five minutes of CPU time per separating surface. The optimization was performed with an algorithm due to Powell that is described in [Lue84] and available from public numerical libraries.

Quadratic optimization problems of the form stated in 17 can be solved in polynomial time with the Ellipsoid method [NY83]. This technique finds first a hyperspace that is guaranteed to contain the optimum; then the volume of this space is reduced iteratively by a constant fraction. The algorithm is polynomial in the number of free parameters p and the encoding size (i. e. the accuracy of the problem and solution). In practice, however, algorithms without guaranteed polynomial convergence are more efficient.

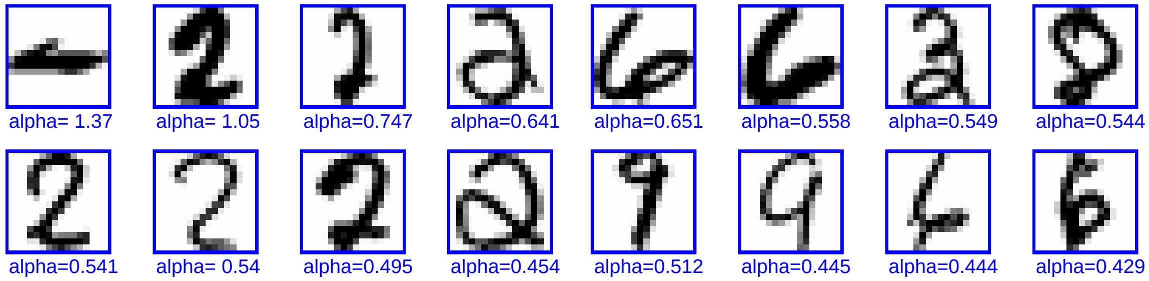

Figure 3: Supporting patterns from database DB2 for class 2 before cleaning. The patterns are ranked according to α_k.

4 EXPERIMENTAL RESULTS

The maximum margin training algorithm has been tested on two databases with images of handwritten digits. The first database (DB1) consists of 1200 clean images recorded from ten subjects. Half of this data is used for training, and the other half is used to evaluate the generalization performance. A comparative analysis of the performance of various classification methods on DB1 can be found in [GVB+92, GPP+89, GBD92]. The other database (DB2) used in the experiment consists of 7300 images for training and 2000 for testing and has been recorded from actual mail pieces. Results for this data have been reported in several publications, see e.g. [CBD+90]. The resolution of the images in both databases is 16 by 16 pixels.

In all experiments, the margin is maximized with respect to $\mathbf{w}$ and b. Ten hypersurfaces, one per class, are used to separate the digits. Regardless of the difficulty of the problem—measured for example by the number of supporting patterns found by the algorithm—the same similarity function $K(\mathbf{x}, \mathbf{x}')$ and preprocessing is used for all hypersurfaces of one experiment. The results obtained with different choices of K corresponding to linear hyperplanes, polynomial classifiers, and basis functions are summarized below. The effect of smoothing is investigated as a simple form of preprocessing.

For linear hyperplane classifiers, corresponding to the similarity function $K(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}'$, the algorithm finds an errorless separation for database DB1. The percentage of errors on the test set is 3.2 %. This result compares favorably to hyperplane classifiers which minimize the mean squared error (backpropagation or pseudo-inverse), for which the error on the test set is 12.7 %.

Database DB2 is also linearly separable but contains several meaningless patterns. Figure 3 shows the supporting patterns with large Lagrange multipliers α_k for the hyperplane for class 2. The percentage of misclassifications on the test set of DB2 drops from 15.2 % without cleaning to 10.5 % after removing meaningless and ambiguous patterns.

Better performance has been achieved with both databases using multilayer neural networks or other classification functions with higher capacity than linear subdividing planes. Tests with polynomial classifiers of order q, for which $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + 1)^q$, give the following error rates and average number of supporting patterns per hypersurface, $<m>$. This average is computed as the total number of supporting patterns divided by the number of decision functions. Patterns that support more than one hypersurface are counted only once in the total. For comparison, the dimension N of φ-space is also listed.

	DB1		DB2		
q	error	$<m>$	error	$<m>$	N
1 (linear)	3.2 %	36	10.5 %	97	256
2	1.5 %	44	5.8 %	89	$3 \cdot 10^4$
3	1.7 %	50	5.2 %	79	$8 \cdot 10^7$
4			4.9 %	72	$4 \cdot 10^9$
5			5.2 %	69	$1 \cdot 10^{12}$

The results obtained for DB2 show a strong decrease of the number of supporting patterns from a linear to a third order polynomial classification function and an equivalently significant decrease of the error rate. Further increase of the order of the polynomial has little effect on either the number of supporting patterns or the performance, unlike the dimension of φ-space, N, which increases exponentially. The lowest error rate, 4.9 % is obtained with a forth order polynomial and is slightly better than the 5.1 % reported for a five layer neural network with a sophisticated architecture [CBD+90], which has been trained and tested on the same data.

In the above experiment, the performance changes drastically between first and second order polynomials. This may be a consequence of the fact that maximum VC-dimension of an q-th order polynomial classifier is equal to the dimension n of the patterns to the q-th power and thus much larger than n. A more gradual change of the VC-dimension is possible when the function K is chosen to be a power series, for example

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(\gamma\, \mathbf{x} \cdot \mathbf{x}'\right) - 1. \qquad (21)$$

In this equation the parameter γ is used to vary the VC-dimension gradually. For small values of γ, equation 21 approaches a linear classifier with VC-dimension at

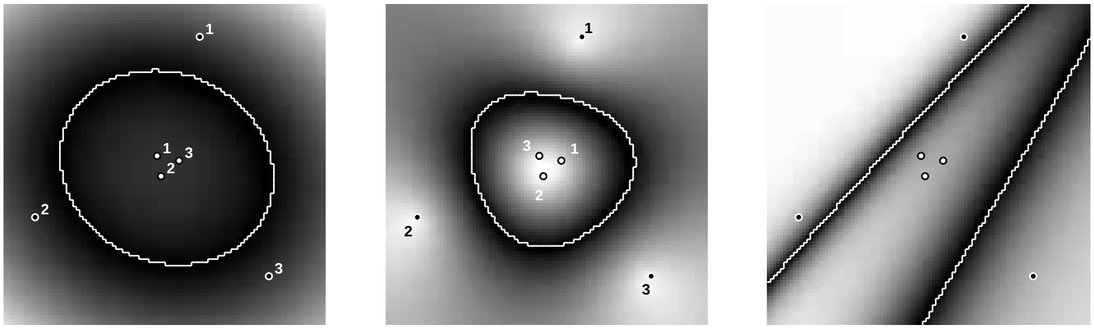

Figure 4: Decision boundaries for maximum margin classifiers with second order polynomial decision rule $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + 1)^2$ (left) and an exponential RBF $K(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|/2)$ (middle). The rightmost picture shows the decision boundary of a two layer neural network with two hidden units trained with backpropagation.

most equal to the dimension n of the patterns plus one. Experiments with database DB1 lead to a slightly better performance than the 1.5 % obtained with a second order polynomial classifier:

γ	DB1
0.25	2.3 %
0.50	2.2 %
0.75	1.3 %
1.00	1.5 %

When $K(\mathbf{x}, \mathbf{x}')$ is chosen to be the hyperbolic tangent, the resulting classifier can be interpreted as a neural network with one hidden layer with m hidden units. The supporting patterns are the weights in the first layer, and the coefficients α_k the weights of the second, linear layer. The number of hidden units is chosen by the training algorithm to maximize the margin between the classes A and B. Substituting the hyperbolic tangent for the exponential function did not lead to better results in our experiments.

The importance of a suitable preprocessing to incorporate knowledge about the task at hand has been pointed out by many researchers. In optical character recognition, preprocessings that introduce some invariance to scaling, rotation, and other distortions are particularly important [SLD92]. As in [GVB$^+$92], smoothing is used to achieve insensitivity to small distortions. The table below lists the error on the test set for different amounts of smoothing. A second order polynomial classifier was used for database DB1, and a forth order polynomial for DB2. The smoothing kernel is Gaussian with standard deviation σ.

	DB1		DB2	
σ	error	$<m>$	error	$<m>$
no smoothing	1.5 %	44	4.9 %	72
0.5	1.3 %	41	4.6 %	73
0.8	0.8 %	36	5.0 %	79
1.0	0.3 %	31	6.0 %	83
1.2	0.8 %	31		

The performance improved considerably for DB1. For DB2 the improvement is less significant and the optimum was obtained for less smoothing than for DB1. This is expected since the number of training patterns in DB2 is much larger than in DB1 (7000 versus 600). A higher performance gain can be expected for more selective hints than smoothing, such as invariance to small rotations or scaling of the digits [SLD92].

Better performance might be achieved with other similarity functions $K(\mathbf{x}, \mathbf{x}')$. Figure 4 shows the decision boundary obtained with a second order polynomial and a radial basis function (RBF) maximum margin classifier with $K(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|/2)$. The decision boundary of the polynomial classifier is much closer to one of the two classes. This is a consequence of the nonlinear transform from $\boldsymbol{\varphi}$-space to $\mathbf{x}$-space of polynomials which realizes a position dependent scaling of distance. Radial Basis Functions do not exhibit this problem. The decision boundary of a two layer neural network trained with backpropagation is shown for comparison.

5 CONCLUSIONS

Maximizing the margin between the class boundary and training patterns is an alternative to other training methods optimizing cost functions such as the mean squared error. This principle is equivalent to minimizing the maximum loss and has a number of important features. These include automatic capacity tuning of the classification function, extraction of a small number of supporting patterns from the training data that are relevant for the classification, and uniqueness of the solution. They are exploited in an efficient learning algorithm for classifiers linear in their parameters with very large capacity, such as high order polynomial or RBF classifiers. Key is the representation of the decision function in a dual space which is of much lower dimensionality than the feature space.

The efficiency and performance of the algorithm have been demonstrated on handwritten digit recognition problems. The achieved performance matches that of sophisticated classifiers, even though no task specific knowledge has been used. The training algorithm is polynomial in the number of training patterns, even in cases when the dimension of the solution space (φ-space) is exponential or infinite. The training time in all experiments was less than an hour on a workstation.

Acknowledgements

We wish to thank our colleagues at UC Berkeley and AT&T Bell Laboratories for many suggestions and stimulating discussions. Comments by L. Bottou, C. Cortes, S. Sanders, S. Solla, A. Zakhor, and the reviewers are gratefully acknowledged. We are especially indebted to R. Baldick and D. Hochbaum for investigating the polynomial convergence property, S. Hein for providing the code for constrained nonlinear optimization, and D. Haussler and M. Warmuth for help and advice regarding performance bounds.

References

[ABR64] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[BH89] E. B. Baum and D. Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, 1989.

[BL88] D. S. Broomhead and D. Lowe. Multivariate functional interpolation and adaptive networks. *Complex Systems*, 2:321 – 355, 1988.

[CBD$^+$90] Yann Le Cun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Larry D. Jackel. Handwritten digit recognition with a back-propagation network. In David S. Touretzky, editor, *Neural Information Processing Systems*, volume 2, pages 396–404. Morgan Kaufmann Publishers, San Mateo, CA, 1990.

[CH53] R. Courant and D. Hilbert. *Methods of mathematical physics.* Interscience, New York, 1953.

[DH73] R.O. Duda and P.E. Hart. *Pattern Classification And Scene Analysis.* Wiley and Son, 1973.

[GBD92] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4 (1):1 – 58, 1992.

[GPP$^+$89] I. Guyon, I. Poujaud, L. Personnaz, G. Dreyfus, J. Denker, and Y. LeCun. Comparing different neural network architectures for classifying handwritten digits. In *Proc. Int. Joint Conf. Neural Networks.* Int. Joint Conference on Neural Networks, 1989.

[GVB$^+$92] Isabelle Guyon, Vladimir Vapnik, Bernhard Boser, Leon Bottou, and Sara Solla. Structural risk minimization for character recognition. In David S. Touretzky, editor, *Neural Information Processing Systems*, volume 4. Morgan Kaufmann Publishers, San Mateo, CA, 1992. To appear.

[HLW88] David Haussler, Nick Littlestone, and Manfred Warmuth. Predicting 0,1-functions on randomly drawn points. In *Proceedings of the 29th Annual Symposium on the Foundations of Computer Science*, pages 100–109. IEEE, 1988.

[KM87] W. Krauth and M. Mezard. Learning algorithms with optimal stability in neural networks. *J. Phys. A: Math. gen.*, 20:L745, 1987.

[Loo72] F. A. Lootsma, editor. *Numerical Methods for Non-linear Optimization.* Academic Press, London, 1972.

[Lue84] David Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, 1984.

[Mac92] D. MacKay. A practical bayesian framework for backprop networks. In David S. Touretzky, editor, *Neural Information Processing Systems*, volume 4. Morgan Kaufmann Publishers, San Mateo, CA, 1992. To appear.

[MD89] J. Moody and C. Darken. Fast learning in networks of locally tuned processing units. *Neural Computation*, 1 (2):281 – 294, 1989.

[MGB$^+$92] N. Matic, I. Guyon, L. Bottou, J. Denker, and V. Vapnik. Computer-aided cleaning of large databases for character recognition. In *Digest ICPR.* ICPR, Amsterdam, August 1992.

[Moo92] J. Moody. Generalization, weight decay, and architecture selection for nonlinear learning systems. In David S. Touretzky, editor, *Neural Information Processing Systems*, volume 4. Morgan Kaufmann Publishers, San Mateo, CA, 1992. To appear.

[NY83] A.S. Nemirovsky and D. D. Yudin. *Problem Complexity and Method Efficiency in Optimization.* Wiley, New York, 1983.

[Omo91] S.M. Omohundro. Bumptrees for efficient function, constraint and classification learning. In R.P. Lippmann and et al., editors, *NIPS-90*, San Mateo CA, 1991. IEEE, Morgan Kaufmann.

[PG90] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 247:978 – 982, February 1990.

[Pog75] T. Poggio. On optimal nonlinear associative recall. *Biol. Cybernetics*, Vol. 19:201–209, 1975.

[Ros62] F. Rosenblatt. *Principles of neurodynamics.* Spartan Books, New York, 1962.

[SLD92] P. Simard, Y. LeCun, and J. Denker. Tangent prop—a formalism for specifying selected invariances in an adaptive network. In David S. Touretzky, editor, *Neural Information Processing Systems*, volume 4. Morgan Kaufmann Publishers, San Mateo, CA, 1992. To appear.

[TLS89] N. Tishby, E. Levin, and S. A. Solla. Consistent inference of probabilities in layered networks: Predictions and generalization. In *Proceedings of the International Joint Conference on Neural Networks*, Washington DC, 1989.

[Vap82] Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer Verlag, New York, 1982.

[VC74] V.N. Vapnik and A.Ya. Chervonenkis. *The theory of pattern recognition.* Nauka, Moscow, 1974.