Machine Learning, 38, 243–255, 2000.
© 2000 Kluwer Academic Publishers. Printed in The Netherlands.

Improved Generalization Through Explicit Optimization of Margins

LLEW MASON
PETER L. BARTLETT
JONATHAN BAXTER
Research School of Information Sciences and Engineering, Australian National University, Canberra, ACT 0200, Australia

Editor: Douglas Fisher

Abstract. Recent theoretical results have shown that the generalization performance of thresholded convex combinations of base classifiers is greatly improved if the underlying convex combination has large *margins* on the training data (i.e., correct examples are classified well away from the decision boundary). Neural network algorithms and AdaBoost have been shown to implicitly maximize margins, thus providing some theoretical justification for their remarkably good generalization performance. In this paper we are concerned with maximizing the margin explicitly. In particular, we prove a theorem bounding the generalization performance of convex combinations in terms of general cost functions of the margin, in contrast to previous results, which were stated in terms of the particular cost function $\mathrm{sgn}(\theta - \text{margin})$. We then present a new algorithm, DOOM, for directly optimizing a piecewise-linear family of cost functions satisfying the conditions of the theorem. Experiments on several of the datasets in the UC Irvine database are presented in which AdaBoost was used to generate a set of base classifiers and then DOOM was used to find the optimal convex combination of those classifiers. In all but one case the convex combination generated by DOOM had lower test error than AdaBoost's combination. In many cases DOOM achieves these lower test errors by sacrificing training error, in the interests of reducing the new cost function. In our experiments the margin plots suggest that the size of the minimum margin is not the critical factor in determining generalization performance.

Keywords: voting methods, ensembles, margins analysis, boosting

1. Introduction

In pattern classification problems, learning algorithms aim to choose a classifier with small error, where the error of a classifier is the probability of misclassifying a random example. Many learning algorithms do this by optimizing some cost function that is defined in terms of the training data. This cost function can be thought of as an error estimate. One example of such a cost function is the training error, i.e. the proportion of training data that is misclassified. When a classifier is chosen from some set to optimize the training error, it gives a biased estimate of the error of the classifier. The magnitude of the bias depends on the complexity of the set of classifiers, which can be quantified in terms of its VC-dimension, and can be excessive.

Recent results have examined alternative cost functions that provide better error estimates in some cases. For example, Bartlett (1998) gives bounds on error for sigmoid networks in terms of the proportion of training examples which are classified correctly with a large margin. The *margin* of a real-valued function $f : X \to \mathbb{R}$ on a training example $(x, y) \in X \times \{-1, 1\}$ is defined as $yf(x)$, so that the sign of f is correct whenever the margin is positive, and the further the value $f(x)$ is from the threshold at zero, the larger the magnitude of the margin. The size of the margin can be interpreted as an indication of the confidence of the classification. For two-layer sigmoid networks, the bounds in (Bartlett, 1998) show that the error of a classifier is no more than the sample average of the *margin cost function* $\operatorname{sgn}(\theta - yf(x))$, which otherwise takes value 1 when the margin is no more than θ and 0 otherwise, plus a complexity penalty term that scales as $\|w\|_1/\theta$, where $\|w\|_1$ is the sum of the magnitudes of the output weights. Neural network learning algorithms typically minimize squared error on the training examples, which tends to increase the margins. This may well explain why neural networks are much better generalizers than the VC bounds would suggest.

More recently, Schapire et al. (1998) have shown a similar result for convex combinations of classifiers, such as those produced by voting methods (Freund & Schapire, 1997; Breiman, 1997). In this theorem, $\operatorname{co}(H)$ denotes the set of convex combinations of functions from H.

Theorem 1 (*Schapire et al.* (1998))**.** *Let P be a distribution over $X \times \{-1, 1\}$, and let S be a sample of m examples chosen independently at random according to P. Assume that the base-hypothesis space H is finite, and let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S, every function $f \in \operatorname{co}(H)$ satisfies the following bound for all $\theta > 0$:*

$$\Pr_P[yf(x) \le 0] \le \Pr_S[yf(x) \le \theta] + O\left(\frac{1}{\sqrt{m}}\left(\frac{\log m \, \log|H|}{\theta^2} + \log\left(\frac{1}{\delta}\right)\right)^{1/2}\right). \tag{1}$$

Hence, the error of a convex combination of classifiers is no more than the sample average of the cost function $\operatorname{sgn}(\theta - yf(x))$ plus a complexity penalty term that scales as $1/\theta$.

One way to think of these results is as a technique for adjusting the effective complexity of the function class by adjusting a parameter of the cost function. Large values of θ correspond to low complexity and small values to high complexity. If the learning algorithm were to optimize the parametrized cost function $\Pr_S[yf(x) \le \theta]$ for large values of θ, it would not be able to make fine distinctions between different functions in the class, and so the effective complexity of the class would be reduced. Note that any variation in $yf(x)$ on a scale less than θ is "collapsed" to zero by the cost function. If the algorithm is able to find a solution with small cost at large θ, then the second term in the error bounds (i.e., the regularization term involving the complexity parameter θ and the size of the base hypothesis class H) would be correspondingly reduced and we would obtain a good bound on the generalization error of f.

Even though neural network and boosting algorithms do not *explicitly* minimize $\Pr_S[yf(x) \le \theta]$, they have been shown to *implicitly* minimize such cost functions of the margin (Bartlett, 1998; Schapire et al., 1998), typically giving reasonable results for the whole range of values of the complexity parameter θ. In this case we use different values of the complexity parameter in the cost functions only in explaining their generalization performance.

Given that algorithms performing implicit margin optimization generalize so well, it is natural to consider whether we can do better by explicitly optimizing some cost function of the margin. This is the main subject of the present paper. In particular, we address the questions: what are suitable cost functions for convex combinations of classifiers, and how useful are they as error estimates? In the next section we give general conditions on parametrized families of cost functions that ensure that they can be used to give error bounds for convex combinations of classifiers. These cost functions are all defined as the sample average of some function of the margin of an example. We prove that the error of a combined classifier is no more than the sample average of the cost function plus a regularization term involving the complexity parameter and the size of the base hypothesis class. The proof uses similar ideas to previous proofs, especially (Schapire et al., 1998), but is simpler.

In the remainder of the paper, we investigate learning algorithms that choose the convex coefficients of a combined classifier by minimizing these cost functions. To overcome some severe computational difficulties, instead of the optimal family, we consider a related family of piecewise linear cost functions. Section 3 describes a gradient descent algorithm for this purpose. In Sections 4 and 5 we describe experiments with this algorithm on classification problems from the UC Irvine database. Even when the base hypotheses are chosen by the AdaBoost algorithm, and we only use the new cost functions to adjust the convex coefficients, we obtained an improvement on the test error of AdaBoost in all but one of the experiments. Margin distribution plots show that in many cases the algorithm achieves these lower errors by sacrificing training error, in the interests of reducing the new cost function. Section 6 presents some conclusions and ideas for further work.

2. Theory

In this section, we derive upper bounds on the misclassification probability of a thresholded convex combination of classifiers, in terms of the sample average of certain functions of the margin, which we call *margin cost functions*. These are functions mapping from the interval $[-1, 1]$ to $\mathbb{R}^+$. Theorem 1 can be derived from this result by considering margin cost functions that are decreasing step functions of the form $\operatorname{sgn}(\theta - yf(x)) + c_\theta$, for some threshold $\theta > 0$ and constant c_θ. In this case θ defines the resolution at which we examine the margins, and hence defines the effective complexity of the convex combination; the penalty term in Theorem 1 arises because a convex combination looks more complex on a small scale than on a larger scale. Similarly, the more general result derived in this section involves a family of cost functions, indexed by an integer-valued parameter N, which measures the resolution at which we examine the margins. We shall see that the cost function $\operatorname{sgn}(\theta - yf(x))$ from the results in Section 1 for neural networks and boosted classifiers is an example of a suitable parametrized family of cost functions, with N growing roughly as $1/\theta^2$. A large value of N (i.e., high resolution and high effective complexity of the convex combination) gives a margin cost function that is close to the threshold function $\operatorname{sgn}(-yf(x))$. Equivalently, the sample average of the cost function is a close approximation to the training error. A small value of N gives a larger margin cost function. There is a trade-off between the effective complexity and how much larger the margin cost function is than the function $\mathrm{sgn}(-yf(x))$. The following definition gives suitable conditions on the margin cost functions that ensure this trade-off is not violated. The particular form of this definition is not important; it arises from the proof technique that is used for the main theorem. Later in this section, we discuss how to optimize over the families of margin cost functions that satisfy this condition; in later sections we shall be concerned with only one family. In particular, the functions Ψ_N are only used in the analysis in this section, and will not concern us later in the paper.

Definition 2. A family $\{C_N : N \in \mathbb{N}\}$ of margin cost functions is *B-admissible* for $B \geq 0$ if for all $N \in \mathbb{N}$ there is an interval $Y \subset \mathbb{R}$ of length no more than B and a function $\Psi_N : [-1, 1] \to Y$ that satisfies

$$\mathrm{sgn}(-\alpha) \leq \mathbf{E}_{Z\sim Q_{N,\alpha}}(\Psi_N(Z)) \leq C_N(\alpha)$$

for all $\alpha \in [-1, 1]$, where $\mathbf{E}_{Z\sim Q_{N,\alpha}}(\cdot)$ denotes the expectation when Z is chosen randomly as $Z = (1/N)\sum_{i=1}^{N} Z_i$ with $Z_i \in \{-1, 1\}$ and $\Pr(Z_i = 1) = (1 + \alpha)/2$.

As an example, consider the following modification of the margin cost functions that appear in the results for neural networks and boosted classifiers. Define $C_N(\alpha) = \mathrm{sgn}(\theta - \alpha) + 1/N^2$ where $\theta = c\sqrt{\log(N)/N}$ with c a constant. It is straightforward to show that this is a B-admissible family of margin cost functions, for some B. This is exhibited by the functions $\Psi_N(\alpha) = \mathrm{sgn}(\theta/2 - \alpha)(1 + 1/(2N^2))$; the proof involves two applications of Chernoff bounds. Notice that, for larger values of N, the cost functions C_N are closer to the threshold function $\mathrm{sgn}(-\alpha)$.

The following theorem is the main result of this section. In this theorem, $\mathrm{co}(H)$ is the set of convex combinations of functions from H.

Theorem 3. *For any B-admissible family $\{C_N : N \in \mathbb{N}\}$ of margin cost functions, any finite hypothesis class H and any distribution P on $X \times \{-1, 1\}$, with probability at least $1 - \delta$ over a random sample S of m labelled examples chosen according to P, every N and every f in $\mathrm{co}(H)$ satisfies*

$$\Pr[yf(x) \leq 0] < \mathbf{E}_S[C_N(yf(x))] + \epsilon_N,$$

where

$$\epsilon_N = \sqrt{\frac{B^2}{2m}\left(N \ln|H| + \ln\left(\frac{N(N+1)}{\delta}\right)\right)}.$$

A similar result applies for infinite classes H with finite VC dimension, using a similar proof. (Ignoring constant factors, $\mathrm{VCdim}(H)\log m$ replaces $\ln|H|$.) We omit the details.

Proof: Fix N and $f \in \mathrm{co}(H)$, and suppose that $f = \sum_i \alpha_i h_i$ for $h_i \in H$. Define

$$\mathrm{co}_N(H) = \left\{ \frac{1}{N} \sum_{j=1}^{N} h_j : h_j \in H \right\},$$

and notice that $|\mathrm{co}_N(H)| \le |H|^N$. As in the proof of Theorem 1 in (Schapire et al., 1998), we shall show that there is a function g in $\mathrm{co}_N(H)$ that approximates f, in the sense that a large difference between the misclassification probability of f and the sample average of $C_N(yf(x))$ leads to a large difference between the expectation and sample average of $\Psi_N(yg(x))$. We prove the existence of the function g using the probabilistic method. Let Q be the distribution on $\mathrm{co}_N(H)$ corresponding to the average of N independent draws from $\{h_i\}$ according to the distribution $\{\alpha_i\}$, and let $Q_{N,\alpha}$ be the distribution of the average of N independent draws from $\{-1, 1\}$ with $\Pr(Z = 1) = (1 + \alpha)/2$, as in Definition 2. Then for any fixed pair x, y, the distribution of $yg(x)$ when g is chosen according to Q is $Q_{N,yf(x)}$. Now, fix the function Ψ_N implied by the B-admissibility condition. By the definition of B-admissibility,

$$\begin{aligned}
\mathbf{E}_{g\sim Q}\mathbf{E}_P[\Psi_N(yg(x))] &= \mathbf{E}_P\mathbf{E}_{g\sim Q}[\Psi_N(yg(x))] \\
&= \mathbf{E}_P\mathbf{E}_{Z\sim Q_{N,yf(x)}}[\Psi_N(Z)] \\
&\ge \mathbf{E}_P\,\mathrm{sgn}(-yf(x)) \\
&= \Pr[yf(x) \le 0].
\end{aligned}$$

Similarly, $\mathbf{E}_S[C_N(yf(x))] \ge \mathbf{E}_{g\sim Q}\mathbf{E}_S[\Psi_N(yg(x))]$. Hence, if

$$\Pr[yf(x) \le 0] - \mathbf{E}_S[C_N(yf(x))] \ge \epsilon_N,$$

then,

$$\mathbf{E}_{g\sim Q}[\mathbf{E}_P[\Psi_N(yg(x))] - \mathbf{E}_S[\Psi_N(yg(x))]] \ge \epsilon_N.$$

It follows that

$$\begin{aligned}
&\Pr[\exists f \in \mathrm{co}(H) : \Pr[yf(x) \le 0] \ge \mathbf{E}_S[C_N(yf(x))] + \epsilon_N] \\
&\quad \le \Pr[\exists g \in \mathrm{co}_N(H) : \mathbf{E}_P[\Psi_N(yg(x))] \ge \mathbf{E}_S[\Psi_N(yg(x))] + \epsilon_N] \\
&\quad \le |H|^N \exp\left(\frac{-2m\epsilon_N^2}{B^2}\right),
\end{aligned}$$

where the last inequality follows from the union bound and Hoeffding's inequality. Setting this probability to $\delta_N = \delta/(N(N+1))$, solving for ϵ_N, and summing over values of N completes the proof, since $\sum_{N\in\mathbb{N}} \delta_N = \delta$.

Theorem 1 follows if we choose $C_N(\alpha) = \mathrm{sgn}(\theta - \alpha) + 1/N^2$, with θ chosen roughly as $\sqrt{(\ln N)/N}$. We have seen that this family is B-admissible, with $\Psi_N(\alpha) = \mathrm{sgn}(\theta/2 -$

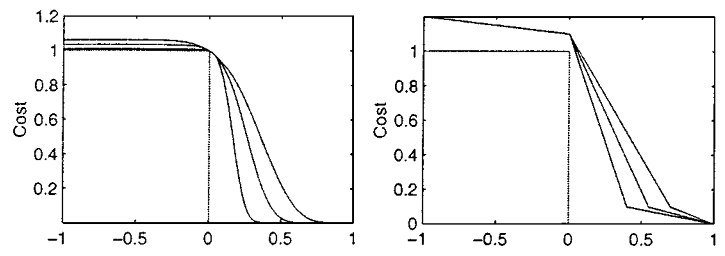

Figure 1. (a) The functions $C_N(\alpha) = \mathbf{E}_{Z\sim Q_{N,\alpha}}[\Psi_N(Z)]$, for $N = 20, 50$ and 200, compared to the function $\operatorname{sgn}(-\alpha)$. Larger values of N correspond to closer approximations to $\operatorname{sgn}(-\alpha)$. (b) Piecewise linear upper bounds on the functions $C_N(\alpha)$, and the function $\operatorname{sgn}(-\alpha)$.

$\alpha)(1 + 1/(2N^2))$. Choosing $C_N(\alpha) = \mathbf{E}_{Z\sim Q_{N,\alpha}}[\Psi_N(Z)]$ gives an immediate improvement on the result of Theorem 1. Figure 1(a) compares $\operatorname{sgn}(-\alpha)$ with the cost functions $C_N(\alpha) = \mathbf{E}_{Z\sim Q_{N,\alpha}}\Psi_N(Z)$, for $N = 20, 50$ and 200. As in the example above, for large values of N the function $C_N(\alpha)$ is close to $\operatorname{sgn}(-\alpha)$. In this case, the sample average of the cost function is close to the training error. As N is decreased, the cost function $C_N(\alpha)$ moves away from the step function, and when we use it to compare functions in $\operatorname{co}(H)$, fine distinctions become more blurred.

3. Algorithm

We now consider how to select convex coefficients $w_1, \ldots, w_T$ for a sequence of $\{-1, 1\}$ classifiers $h_1, \ldots, h_T$ so that the combined classifier $f(x) = \sum_{t=1}^{T} w_t h_t(x)$ has small error (i.e., $P[yf(x) \leq 0]$ is small). We do not supply a procedure for also selecting the base hypotheses. In all experiments we simply used the hypotheses provided by AdaBoost, since the aim of the experiments was to investigate how useful are the error estimates provided by the cost functions of the previous section.

3.1. *Approximation to the cost function*

If we take Theorem 3 at face value and ignore log terms, the best error bound is obtained if the weights $w_1, \ldots, w_T$ and the complexity N are chosen to minimize

$$\frac{1}{m}\sum_{i=1}^{m} C_N(y_i f(x_i)) + \kappa\sqrt{\frac{N}{m}}, \tag{2}$$

where κ is a constant and $\{C_N\}$ is the family of cost functions plotted in figure 1(a). However, there are two main difficulties with optimizing (2) directly. First, the cost-functions C_N are rather flat near ± 1, which makes things difficult for local methods such as gradient descent. Second, although Theorem 3 provides an expression for the constant κ, in practical problems this will almost certainly be an overestimate and so our penalty for even moderately complex models will be too great.

To solve the first problem we bounded C_N above by a monotone decreasing function and used this upper bound as our margin cost function. We considered the use of a sigmoid as a bounding cost function, but the existence of local minima at $\pm\infty$ caused difficulties for gradient descent approaches. Instead we adopted a piecewise linear family of cost functions of the form

$$C_\theta(\alpha) = \begin{cases} (1.2-\gamma)-\gamma\alpha & \text{for } -1 \leq \alpha \leq 0 \\ (1.2-\gamma)-(1.2-2\gamma)\alpha/\theta & \text{for } 0 < \alpha \leq \theta \\ \gamma/(1-\theta)-\gamma\alpha/(1-\theta) & \text{for } \theta < \alpha \leq 1 \end{cases}$$

for $\theta \in (0, 1)$ (as shown in figure 1(b)). Note that θ plays the role of a complexity parameter, only in this case smaller values of θ correspond to higher complexity classes. For all experiments, γ was fixed at 0.1.

To solve the second problem, instead of optimizing the average cost of the margins plus a penalty term over all values of the parameter θ, we estimated the optimal value of θ using a cross-validation set. That is, for fixed values of θ in a discrete but fairly dense set we selected weights optimizing the average cost $\frac{1}{m}\sum_{i=1}^m C_\theta(y_i f(x_i))$ and then chose the solution with smallest error on an independent cross-validation set.

3.2. Optimizing C_θ

Unfortunately, even with the restriction to piecewise linear cost functions, the problem of optimizing $\frac{1}{m}\sum_{i=1}^m C_\theta(y_i f(x_i))$ is still NP-hard. Fortunately, the nature of this cost function makes it possible to find successful heuristics, which is why we chose it.

The algorithm we have devised to optimize the C_θ family of cost functions is called Direct Optimization Of Margins (DOOM). DOOM is basically a form of gradient descent, with two complications: we have to take account of the fact that our cost function is not differentiable at 0 and θ, and we have to ensure that the weight vector lies on the unit ball in l_1. Both these problems are addressed in DOOM, the pseudo-code of which is shown in figure 2.

In order to avoid problems with local minima we actually allow the weight vector to lie within the l_1-ball throughout optimization rather than on the l_1-ball. Since an increase in the l_1-norm of the weight vector generally corresponds to a decrease in the value of the cost function, the weight vector tends to approach the surface of the l_1-ball as the optimization proceeds. If the weight vector reaches the surface of the l_1-ball and the update direction points out of the l_1-ball, it is simply projected back to the surface of the l_1-ball.

To understand the algorithm's operation, firstly observe that the gradient of $\frac{1}{m}\sum_{i=1}^m C_\theta(y_i f(x_i))$ is a constant function of the weights $w = (w_1, \ldots, w_T)$ provided no example (x_i, y_i) "crosses" one of the discontinuities at 0 or θ (i.e., provided the margin $y_i f(x_i)$ does not cross 0 or θ. Examples cannot cross the discontinuities at ± 1 because the weight vector is constrained to lie within the l_1 ball). Hence, the central operation of DOOM is to step in the negative gradient direction until an example's margin hits one of the discontinuities, projecting where necessary to ensure the weight vector lies within the l_1 ball. At this point the gradient vector becomes multi-valued (generally two-valued but if more than

```
globals: Integer:                 MAX_CONSTRAINTS
         Base Hypotheses:         $h_1, \ldots, h_T$
         Training Set:            $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
         Complexity:              $\theta$
         2×Machine Precision:     $\epsilon$

procedure doom($w, E$, reset)
    For all $(x_i, y_i) \in S$, let $f_w(x_i) := \sum_{j=1}^{T} w_j h_j(x_i)$
    Let cost($w$):= $\sum_{(x_i,y_i) \in S-E} C_\theta(y_i f_w(x_i))$
    Let $A := \{(x_i, y_i) \in S - E\colon y_i f_w(x_i) = 0 \text{ or } \theta\}$
    if ($A$ == $\emptyset$) then
        Let $g := -\nabla_w$cost($w$). project($w, g, E$)
    else
        Let $B := A$
        if ($|A| >$ MAX_CONSTRAINTS) then
            Let $B$ be a random subset of $A$ of size MAX_CONSTRAINTS
        endif
        Let $N := |B|$
        for each $(b_1, \ldots, b_N) \in \{\pm 1\}^N$
            Let $g^{(b_1,\ldots,b_N)} := -\nabla_w^{(b_1,\ldots,b_N)}$cost($w$), where $\nabla_w^{(b_1,\ldots,b_N)}$ means
            take the left or right derivative at the $ith$ example in $B$,
            according as $b_i$ is $+1$ or $-1$. project($w, g, E$)
            Let $c^{(b_1,\ldots,b_N)} :=$ cost($w$) - cost($w + \epsilon g$)
        end
        Let $(b_1^*, \ldots, b_N^*) := \operatorname{argmax}_{(b_1,\ldots,b_N)} \left\{c^{(b_1,\ldots,b_N)}\right\}$
        Let $c := c^{(b_1^*,\ldots,b_N^*)}$, $g := g^{(b_1^*,\ldots,b_N^*)}$
        if ($c \leq 0$) then
            Let $E := E \cup A$. project($w, g, E$)
        endif
    endif
    if ($|g| < \epsilon$) then
        if (reset = FALSE) then doom($w, \emptyset$, TRUE) else return
    else
        Let $\delta_1 := \operatorname{argmin}_\delta \{\delta\colon \exists (x_i, y_i) \in S - E \cup A\colon y_i f_{w+\delta g}(x_i) = 0 \text{ or } \theta\}$
        Let $\delta_2$ be the distance to the closest face of the $l_1$-ball to
        $w$, in the direction $g$ (ignoring those faces which already
        intersect $w$).
        doom($w + \min(\delta_1, \delta_2) g, E$, FALSE)
    endif
end

procedure project($w, g, E$)
    Project $g$ onto $\left\{v \in \mathbb{R}^T : \forall (x_i, y_i) \in E, \sum_{j=1}^{T} v_j h_j(x_i) = 0\right\}$
    Project $g$ onto all faces of the $l_1$ ball intersecting $w$.
end
```

Figure 2. Pseudocode for the DOOM (Direct Optimization Of Margins) algorithm.

one point hits a discontinuity simultaneously then 2^N-valued where N is the number of points). Each of the possible gradient directions is then tested by taking a small step in that direction. A random subset of the gradient directions is chosen if there are too many gradient directions. If none of the directions lead to a decrease in the cost, the examples whose margins lie on discontinuities of the cost function are added to a constraint set E. In subsequent iterations the same stepping procedure above is followed except that the direction step is modified to ensure that the examples in E do not move (i.e., they remain on the discontinuity points of C_θ). This is achieved for a constraint set $E = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ by projecting any update direction onto the orthogonal subspace of the space spanned by the vectors $\{[h_1(x_1), \ldots, h_T(x_1)], \ldots, [h_1(x_k), \ldots, h_T(x_k)]\}$ where $h_1, \ldots, h_T$ are the base hypotheses. If no progress is made in any iteration, the constraint set E is reset to zero. If still no progress is made the procedure terminates.

For each of the experiments reported in the next section, to increase the chance of finding the global minimum, the DOOM algorithm was called with 1000 random initial weight vectors and the solution with minimum cost was selected.

4. Experiments

For the experiments presented in this paper we used a selection of two-class data sets from the UC Irvine database (Blake, Keogh, & Merz, 1998). Each data set was randomly separated into train, test and validation sets, with the test and validation sets being equal in size. This process was repeated 10 times and the results averaged. The data sets are listed in Table 1. Each experiment consisted of the following steps. First, AdaBoost was run on the training data to produce a sequence of base classifiers and their corresponding weights. In all of the experiments the base classifiers were axis-orthogonal hyperplanes (also known as decision stumps). This choice ensured that the complexity of the class of base classifiers was constant. Boosting was halted when adding a new classifier failed to decrease the error on the validation set. DOOM was then run on the classifiers produced by AdaBoost for a large

Table 1. The UCI datasets used in the experiments.

Data Set	Training	Test	Attributes
Cleveland Heart Disease	103	100	14
Credit Application	100	295	15
German	300	350	24
Glass	70	72	10
Ionosphere	101	125	34
King Rook vs. King Pawn	100	1000	36
Pima Indians Diabetes	200	284	8
Sonar	58	75	60
Tic-Tac-Toe	300	329	9
Wisconsin Breast Cancer	199	250	10

range of θ values and 1000 random initial weight vectors for each value of θ. The weight vector (and θ value) with minimum misclassification on the validation set was chosen as the final solution.

In some cases the training sets were reduced in size to make overfitting more likely (so that complexity regularization with DOOM could have an effect). In three of the datasets (Credit Application, Wisconsin Breast Cancer and Pima Indians Diabetes), AdaBoost gained no advantage from using more than a single classifier. In these datasets, the number of classifiers was chosen so that the validation error was reasonably stable.

5. Results

Figure 3 shows cumulative training margin distribution graphs for four of the datasets for both AdaBoost and DOOM (with optimal θ chosen by cross-validation). For a given margin the value on the curve corresponds to the proportion of training examples with margin less than or equal to this value. The test errors for both algorithms are also shown for comparison, as short horizontal lines on the vertical axis.

There are several things worth noting about the margin distributions generated using DOOM as compared to those generated using AdaBoost. First, they show that the value of the minimum training margin has no real impact on generalization performance. The idea

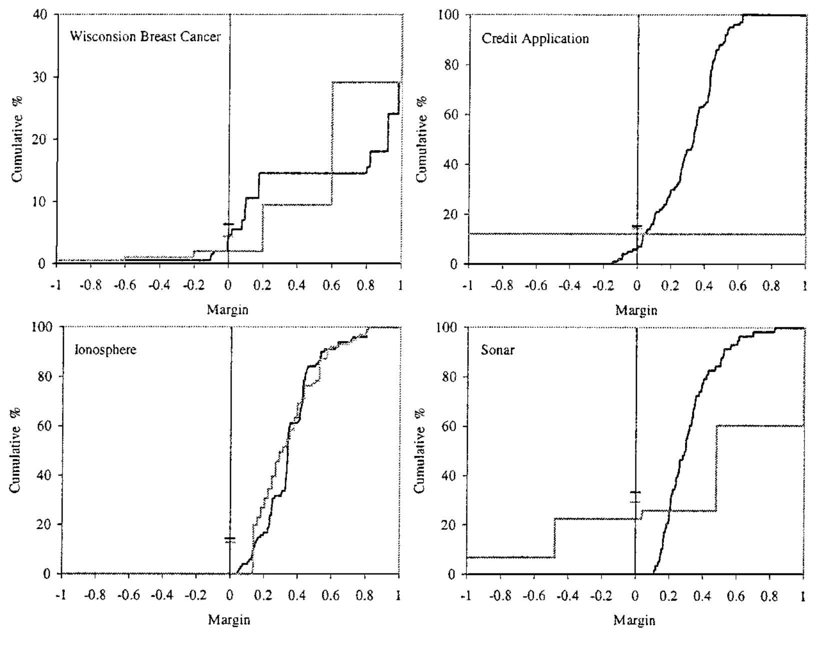

Figure 3. Cumulative training margin distributions for four of the tested data sets. The dark curve corresponds to weights chosen by AdaBoost, while the light curve corresponds to weights chosen by DOOM with optimal θ selected by cross-validation. The test errors for AdaBoost and DOOM are marked on the vertical axis at margin 0.

of maximizing the minimum margin has been examined by Breiman (1997) and Grove & Schuurmans (1998). Breiman presented several voting methods which provably maximize the minimum margin, while Grove and Schuurmans maximized the minimum margin explicitly by linear programming. In both cases a maximization of minimum margin at the expense of all other margins generally gave worse generalization performance than AdaBoost. As can be seen in figure 3, results for the Credit Application and Sonar data sets illustrates that the generalization performance of the combined classifier produced by DOOM can be as good or better than that of the classifier produced by AdaBoost, despite having dramatically worse minimum training margin. Conversely, figure 3 (Ionosphere data set) shows that an improved minimum margin can result in improved generalization performance. These results clearly demonstrate that the minimum margin is not the important quantity.

Second, the margin distributions show that there is a balance to be found between training error and complexity (as measured by θ). DOOM is willing to sacrifice training error in order to reduce complexity and thereby obtain a better margin distribution. For instance, in figure 3 (Sonar data set), DOOM's training error is over 20% while AdaBoost's is 0%. Despite sacrificing this training error, DOOM's test error is 5% less than that of AdaBoost's. The reason for this success can be seen in figure 4, which illustrates the changes in the cost

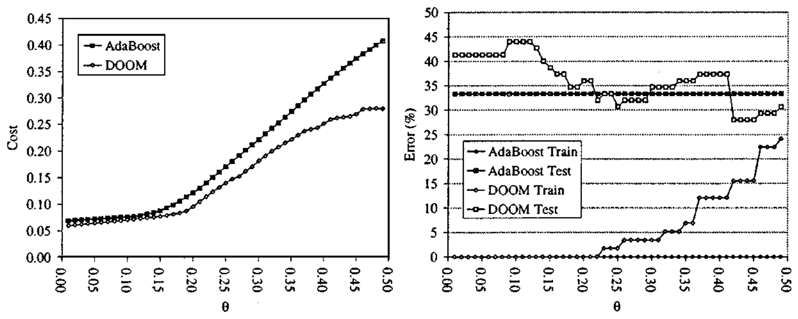

Figure 4. Sonar data set. (a) Plot of cost ($\frac{1}{m}\sum_{i=1}^{m} C_\theta(y_i f(x_i))$) against θ for both AdaBoost and DOOM. (b) Plot of training and test error against θ for both AdaBoost and DOOM.

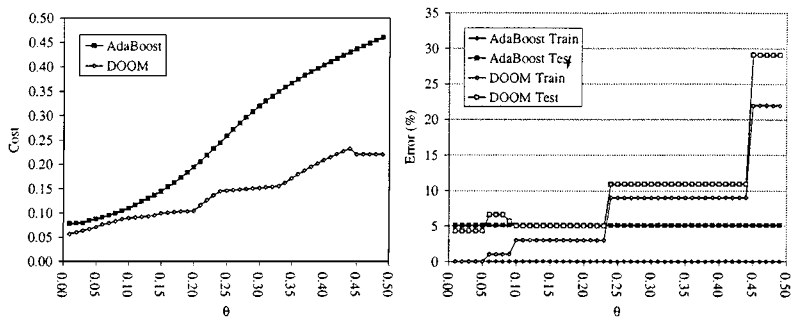

Figure 5. King Rook vs. King Pawn data set. (a) Plot of cost ($\frac{1}{m}\sum_{i=1}^{m} C_\theta(y_i f(x_i))$) against θ for both AdaBoost and DOOM. (b) Plot of training and test error against θ for both AdaBoost and DOOM.

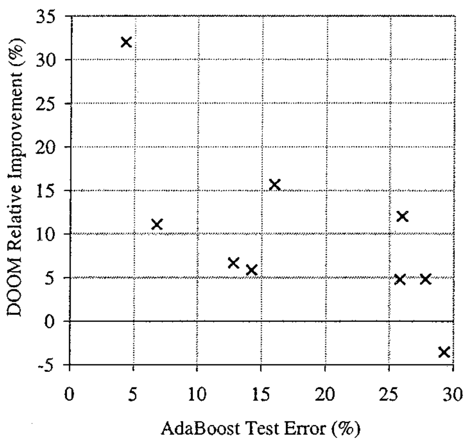

Figure 6. Relative improvement of DOOM over AdaBoost for all examined datasets. The vertical axis represents the percentage improvement in test error which DOOM exhibited over AdaBoost.

function, training error, and test error as a function of θ. The optimal complexity for this data set is low (as indicated by a large optimal θ). For this data set, a reduction in complexity is more important to generalization error than a reduction in training error. The relationship between the cost function, training error, and test error for the King Rook vs. King Pawn data set is shown in figure 5.

A comparison between the test errors generated by AdaBoost and DOOM is shown in figure 6. In only one data set did DOOM produce a classifier which performed worse than AdaBoost in terms of test error; for most data sets DOOM's test error was a significant improvement over AdaBoost's.

6. Conclusions

In this paper we have addressed the question: what are suitable cost functions for convex combinations of base hypotheses? For general families of cost functions that are functions of the *margin* of a sample, we proved (Theorem 3) that the error of a convex combination is no more than the sample average of the cost function plus a regularization term involving the complexity of the cost function and the size of the base hypothesis class.

We constructed a piecewise linear family of cost functions satisfying the conditions of Theorem 3 and presented a heuristic algorithm (DOOM) for optimizing the sample average of the cost.

We ran experiments on several of the datasets in the UC Irvine database, in which AdaBoost was used to generate a set of base classifiers and then DOOM was used to find the optimal convex combination of those classifiers. In all but one case the convex combination generated by DOOM had lower test error than AdaBoost's combination. Margin distribution plots show that in many cases DOOM achieves these lower test errors by sacrificing training error, in the interests of reducing the new cost function. The margin plots also show that the size of the minimum margin is not a critical factor in determining generalization performance.

It should be noted that all of the experiments presented used decision stumps as base classifiers. Recent results (Breiman, 1997) have shown that using more complex base classifiers (such as decision trees) can significantly affect the relative performance of voting methods. Understanding the possible interactions between a voting method and its class of base classifiers is an important open problem.

One obvious direction for further work is the design of algorithms that choose the base hypotheses as well as the convex coefficients (in general AdaBoost's base hypotheses will be suboptimal insofar as minimizing our cost function is concerned). The base hypotheses and the convex coefficients could be optimized either sequentially, as in AdaBoost, or globally. We are currently investigating algorithms that use a generalization of the cost function formulation of AdaBoost described in (Breiman, 1997; Frean & Downs, 1998). For more details see Mason et al. (to appear).

Acknowledgments

Thanks to Yoav Freund, Wee Sun Lee and Rob Schapire for helpful comments and suggestions. This research was supported in part by a grant from the Australian Research Council. Jonathan Baxter was supported by an Australian Research Council Fellowship and Llew Mason was supported by an Australian Postgraduate Award.

References

Bartlett, P. L. (1998). The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, *44*(2), 525–536.

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

Breiman, L. (1997). *Prediction games and arcing algorithms*. Technical Report 504, Department of Statistics, University of California, Berkeley.

Frean, M. & Downs, T. (1998). *A simple cost function for boosting*. Technical Report, Department of Computer Science and Electrical Engineering, University of Queensland.

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*(1), 119–139.

Grove, A. & Schuurmans, D. (1998). Boosting in the limit: maximizing the margin of learned ensembles. Proceedings of the Fifteenth National Conference on Artificial Intelligence (pp. 692–699).

Mason, L., Baxter, J., Bartlett, P. L., & Frean, M. (to appear). Functional gradient techniques for combining hypotheses. In: A. J. Smola, P. Bartlett, B. Schölkopf, & C. Schuurmans (Eds.), *Advances in large margin classifiers*. Cambridge, MA: MIT Press.

Schapire, R. E., Freund, Y., Bartlett, P. L., & Lee, W. S. (1998). Boosting the margin : a new explanation for the effectiveness of voting methods. *Annals of Statistics*, *26*(5), 1651–1686.

Received June 8, 1998
Accepted July 9, 1999
Final manuscript July 9, 1999