Recursive training methods for robust classification: A sequential analytic centering approach to the support vector machine

Katherine Comanor[a] and Lieven Vandenberghe[a]

[a]Electrical Engineering Department, University of California Los Angeles

ABSTRACT

The support vector machine (SVM) is a supervised learning algorithm used in a variety of applications, including robust target classification. The SVM training problem can be formulated as dense quadratic programming problem (QP). In practice, this QP is solved in batch mode, using general-purpose interior-point solvers. Although quite efficient, these implementations are not well suited in situations where the training vectors are made available sequentially. In this paper we discuss a recursive algorithm for SVM training. The algorithm is based on efficient updates of approximate solutions on the dual central path of the QP and can be analyzed using the convergence theory recently developed for interior-point methods. The idea is related to cutting-plane methods for large-scale optimization and sequential analytic centering techniques used successfully in set-membership estimation methods in signal processing.

Keywords: Support Vector Machines, classification, recursive training, sequential analytic centering, convex optimization

1. INTRODUCTION

The support vector machine (SVM) is a supervised learning algorithm for pattern classification, introduced by Vapnik in 1992.[1] It has received considerable attention in the last decade and is widely used in a variety of applications (see Schölkopf and Schmola[2] for a comprehensive survey). Support vector classifiers are based on linearly parametrized decision functions of the form

$$f(z) = \theta_1 F_1(z) + \cdots + \theta_n F_n(z) = \theta^T F(z),$$

where the components of $F : \mathbf{R}^p \to \mathbf{R}^n$ form a set of basis functions on $\mathbf{R}^p$. In the simplest case, the decision function f is affine, and we take $n = p + 1$,

$$F(z) = \begin{bmatrix} z_1 & z_2 & \cdots & z_p & 1 \end{bmatrix}^T.$$

To determine the decision function f (*i.e.*, the parameters θ) we use training data in the form of N vectors $v_i \in \mathbf{R}^p$, partitioned into two sets by N binary labels $y_i \in \{-1, 1\}$. The coefficients θ are calculated by solving the quadratic programming problem (QP)

$$\begin{array}{ll} \text{minimize} & (1/2)\theta^T\theta + \gamma \mathbf{1}^T u \\ \text{subject to} & X\theta \geq \mathbf{1} - u \\ & u \geq 0, \end{array} \tag{1}$$

Address: UCLA Electrical Engineering Department, Los Angeles, CA 90095-1594. E-mail: kcomanor@ee.ucla.edu, vandenbe@ee.ucla.edu.

with variables $\theta \in \mathbf{R}^n$, and $u \in \mathbf{R}^N$. The matrix X is defined as

$$X = \begin{bmatrix} y_1 F(v_1)^T \\ y_2 F(v_2)^T \\ \vdots \\ y_N F(v_N)^T \end{bmatrix} \in \mathbf{R}^{N \times n}, \tag{2}$$

and we will write its rows as $x_i^T = y_i F(v_i)^T$. The symbol $\mathbf{1}$ denotes a vector with all its components equal to one. The coefficient $\gamma > 0$ is a parameter set by the user, and controls the relative weight of the two terms in the objective.

The constraints in (1) have the following interpretation. The training vector v_i is considered correctly classified by the decision function $f = \theta^T F$ if $x_i^T \theta = y_i \theta^T F(v_i) \geq 1$, *i.e.*,

$$f(v_i) = \theta^T F(v_i) \geq 1 \text{ if } y_i = 1, \qquad \text{and} \qquad f(v_i) = \theta^T F(v_i) \leq -1 \text{ if } y_i = -1.$$

The variable u is the *slack vector* in these inequalities and measures the amount of constraint violation: at the optimum, u and θ satisfy

$$u_i = \max\{0, 1 - x_i^T \theta\}, \quad i = 1, \ldots, N.$$

In other words, $u_i = 0$ if the training point v_i is correctly classified and $u_i > 0$ otherwise. The second term $\mathbf{1}^T u$ in the objective function is the total slack, *i.e.*, total constraint violation, which is used as a measure of classification error.

The first term in the objective function penalizes large θ and has an intuitive geometric meaning. It can be shown that $2/\|\theta\|$ is the distance between the hyperplanes in $\mathbf{R}^n$ defined by $\theta^T x = 1$ and $\theta^T x = -1$. This distance is a good measure of the robustness of the classifier with respect to perturbations to the training set. By minimizing $\|\theta\|$ we maximize the margin between the two hyperplanes. In the QP (1), we control the trade-off between classification error (as measured by the total slack violation) and robustness (inversely proportional to the $\|\theta\|$) by the parameter γ.

Our purpose is to describe a recursive method for solving the training problem (1). By this we mean a method in which we efficiently update the solution θ, after adding a new row to X. Recursive methods are important when the training vectors are made available sequentially.

We will also consider the simpler QP

$$\begin{array}{ll} \text{minimize} & (1/2)\theta^T \theta \\ \text{subject to} & X\theta \geq \mathbf{1}. \end{array} \tag{3}$$

In this QP we seek a classifier $\theta^T F$ that separates the two sets, and maximizes the distance between the hyperplanes defined by $\theta^T x = \pm 1$.

The outline of the paper is as follows. In §2 and §3 we review some basic facts of quadratic programming duality and the primal-dual interior-point methods that are commonly used to solve the QPs (1) and (3). In §4 and §5 we describe a dual sequential analytic centering method based on Newton's method. Some numerical examples are provided in §6.

2. DUALITY

The dual problem of the training problem (1) is defined as

$$\begin{array}{ll} \text{maximize} & -(1/2)\alpha^T X X^T \alpha + \mathbf{1}^T \alpha \\ \text{subject to} & 0 \leq \alpha \leq \gamma \mathbf{1}. \end{array} \tag{4}$$

The dual problem is also a QP, with variable $\alpha \in \mathbf{R}^N$. The main properties relating the primal and dual QPs are the following.

- The optimal values of both problems are equal: $p^\star = d^\star$ where $p^\star$ is the optmal value of (1) and $d^\star$ is the optimal value of (4).

- If θ, u are primal feasible (*i.e.*, $X\theta \geq \mathbf{1} - u$ and $u \geq 0$) and α is dual feasible (*i.e.*, $0 \leq \alpha \leq \gamma\mathbf{1}$), then

$$\left(\frac{1}{2}\theta^T\theta + \gamma\mathbf{1}^T u\right) - \left(-\frac{1}{2}\alpha^T XX^T\alpha + \mathbf{1}^T\alpha\right) \quad = \quad \frac{1}{2}\|\theta - X^T\alpha\|^2 + \alpha^T(X\theta - \mathbf{1} + u) + (\gamma\mathbf{1} - \alpha)^T u. \quad (5)$$

  This quantity is known as the *duality gap* associated with θ, u, and α. The duality gap associated with a pair of feasible points is the difference between the primal and dual objective values. It is always nonnegative, and zero only if θ, u is primal optimal and α is dual optimal.

- The primal optimal solution $\theta^\star$, $u^\star$ and the dual optimal solution $\alpha^\star$ are related via the Karush-Kuhn-Tucker (KKT) conditions

$$X\theta^\star \geq \mathbf{1} - u^\star, \qquad u^\star \geq 0, \qquad 0 \leq \alpha^\star \leq \gamma\mathbf{1}$$
$$\theta^\star = X^T\alpha^\star$$
$$\alpha_i^\star(x_i^T\theta^\star + u_i^\star - 1) = 0, \qquad (\gamma - \alpha_i^\star)u_i^\star = 0, \qquad i = 1, \ldots, N.$$

  These last conditions are called *complementary slackness* conditions.

For future reference, we also give the dual problem and optimality conditions for the QP (3). The dual problem is

$$\begin{array}{ll} \text{maximize} & -(1/2)\alpha^T XX^T\alpha + \mathbf{1}^T\alpha \\ \text{subject to} & \alpha \geq 0, \end{array}$$

with variable $\alpha \in \mathbf{R}^N$. The KKT conditions are

$$X\theta^\star \geq \mathbf{1}, \qquad \alpha^\star \geq 0$$
$$\theta^\star = X^T\alpha^\star$$
$$\alpha_i^\star(x_i^T\theta^\star - 1) = 0, \qquad i = 1, \ldots, N.$$

3. PRIMAL-DUAL INTERIOR-POINT METHODS

Primal-dual interior-point methods are widely regarded as very efficient methods for solving LPs or QPs such as (1).[3,4] Applied to problem (1), each iteration of a primal-dual method requires solving a set of positive definite linear equations of the form

$$(XX^T + D)\Delta\alpha = r \quad (6)$$

where D is a positive diagonal matrix, and the values of D and r change at each iteration. The matrix XX^T is often referred to as the *kernel matrix* in the SVM literature. The kernel matrix needs to be computed only once, at the start of the algorithm. Moreover, although it is defined as a product of an $N \times n$ matrix with its transpose, it can be constructed very efficiently (in $O(pN^2)$ operations) for commonly used basis functions. The resulting algorithm converges fast, often in less than 20 iterations, almost independent of problem size. As a rule of thumb, the cost of solving (1) is therefore roughly equal to the cost of solving about 20 systems of equations of the form (6).

While most SVM implementations are based on general-purpose packages (such as LOQO[5] or MOSEK[6]), further improvements are possible in special purpose implementations of the interior-point method. For example, approximating the dense coefficient matrix of (6) by a sparse matrix, or by a sum of a diagonal and a low-rank matrix, allows us to efficiently compute approximate solutions of (6).[7–9]

4. SEQUENTIAL ANALYTIC CENTERING

The standard approach to the SVM training problem, as outlined in the previous section, operates in batch mode, *i.e.*, it requires all training data to be available before the classifier can be found. In some applications, however, it is desirable to update the classifier in a recursive fashion as incoming data become available. While interior-point methods are very efficient, they are not particularly well suited for this recursive scenario. First of all, they require the complete training data matrix X. Secondly, although they often allow the user to specify a starting point, interior-point methods do not necessarily work better with so-called warm starts. In other words, the optimal θ for the training set $\{v_1, \ldots, v_N\}$ is not necessarily a good starting point when searching for the optimal θ for the training set $\{v_1, \ldots, v_{N+1}\}$. Roughly speaking, this phenomenon is due to the fact that interior point methods perform better with starting points that are well centered in the feasible region as opposed to ones close to the boundary.

Alternative methods that are better suited for recursive implementation include the classical *row-action methods*,[10] which are related to the well-known perceptron convergence rule of the neural network literature. While very simple and easy to implement, and recursive in nature, these methods often suffer from slow convergence. A more promising class of recursive algorithms are the more recent sequential analytic centering methods. These methods include the the *Analytic Centering Cutting-Plane Method* (ACCPM),[11–13] which is popular both as a general-purpose convex optimization algorithm, and, when combined with decomposition techniques, also in distributed optimization. Other examples are the analytic centering techniques for recursive parameter estimation in signal processing and control.[14, 15] Sequential analytic centering methods can be analyzed rigorously using the techniques developed for interior-point methods (in particular, the convergence analysis of Newton's method for logarithmic barrier functions).[13, 16, 17]

In this section we formulate a simple sequential analytic centering method to solve the QPs (1) and (4). The method is a variation of the parameter estimation method of Bay, Ye, and Tempo.[14]

4.1. The dual central path

The dual central path for problem (1) is defined as the set $\{\alpha(t) \mid t > 0\}$, where $\alpha(t)$ is the minimizer of the strictly convex function

$$\phi_t(\alpha) = t(\frac{1}{2}\alpha^T X X^T \alpha - \mathbf{1}^T \alpha) - \sum_{i=1}^{N} \log(\gamma - \alpha_i) - \sum_{i=1}^{N} \log \alpha_i. \tag{7}$$

Dual central points $\alpha(t)$ have a number of interesting properties, which all follow from the optimality condition $\nabla\phi_t(\alpha(t)) = 0$, where $\nabla\phi_t(\alpha)$ is the gradient of ϕ_t:

$$\nabla\phi_t(\alpha) = t(XX^T\alpha - \mathbf{1}) + (\mathbf{diag}(\gamma\mathbf{1} - \alpha)^{-1} - \mathbf{diag}(\alpha)^{-1})\mathbf{1}.$$

Define

$$\theta(t) = X^T\alpha(t), \qquad u_i(t) = \frac{1}{t(\gamma - \alpha_i(t))}, \quad i = 1, \ldots, N. \tag{8}$$

Then $u(t) > 0$ and, from $\nabla\phi_t(\alpha(t)) = 0$,

$$X\theta(t) = \mathbf{1} - u(t) + \mathbf{diag}(t\alpha)^{-1}\mathbf{1} > \mathbf{1} - u(t). \tag{9}$$

In other words, $\theta(t)$, $u(t)$ are strictly primal feasible for the problem (1). The duality gap associated with the primal and dual feasible points $\theta(t)$, $u(t)$, $\alpha(t)$ is, using the expresssion (5),

$$\alpha(t)^T\left(X\theta(t) - \mathbf{1} + u(t)\right) + u(t)^T\left(\gamma\mathbf{1} - \alpha(t)\right) = 2N/t.$$

This shows that by minimizing (7) we compute primal and dual feasible points with a duality gap $2N/t$. In particular, this implies that

$$\frac{1}{2}\theta(t)^T\theta(t) + \gamma\mathbf{1}^T u(t) - p^\star \leq \frac{2N}{t},$$

where $p^\star$ is the optimal value of (1).

To summarize, by minimizing (7) we can compute a suboptimal solution for (1) with an (absolute) accuracy of at least $2N/t$.

The corresponding definitions for the QP (3) are similar. We define the dual central points $\alpha(t)$ as the minimizers of

$$\phi_t(\alpha) = t(\frac{1}{2}\alpha^T XX^T\alpha - \mathbf{1}^T\alpha) - \sum_{i=1}^N \log\alpha_i, \tag{10}$$

and define $\theta(t) = X^T\alpha(t)$. It can be shown that $\theta(t)$ is primal feasible and $(1/2)\theta(t)^T\theta(t) - p^\star \leq N/t$, where $p^\star$ is the optimal value of (3).

4.2. Sequential analytic centering

In this section we explicitly denote the dimension N by superscripts N: $X^{(N)}$ is the matrix defined in (2); $\theta^{(N)}$, $u^{(N)}$, $\alpha^{(N)}$, will denote (approximate) solutions of problems (1) and (4). The idea of the dual analytic centering method is as follows.

For each $N = 1, 2, \ldots$, we solve the unconstrained minimization problem

$$\text{minimize} \quad \phi_t^{(N)}(\alpha) = t(\tfrac{1}{2}\alpha^T X^{(N)}X^{(N)T}\alpha - \mathbf{1}^T\alpha) - \textstyle\sum_{i=1}^N \log(\gamma-\alpha_i) - \sum_{i=1}^N \log\alpha_i, \tag{11}$$

with $t = (2N)/\epsilon$, where ϵ is a specified tolerance. To solve (11) we use Newton's method with a starting point $\alpha \in \mathbf{R}^N$ of the form $\alpha = (\alpha^{(N-1)}, \alpha_N)$ where $0 < \alpha_N < \gamma$. (We will make some practical suggestions for selecting α_N in §6. The implementation of Newton's method is discussed in §5.) Then we take

$$\theta^{(N)} = {X^{(N)}}^T\alpha^{(N)}, \qquad u_i^{(N)} = \max\{0, 1 - x_i^T\theta^{(N)}\}, \quad i = 1, \ldots, N,$$

where $\alpha^{(N)} = \operatorname{argmin}\phi_t^{(N)}(\alpha)$ is the solution of (11).

It follows from the duality results in §4.1 that $\theta^{(N)}$, $u^{(N)}$ are primal feasible, with

$$\frac{1}{2}{\theta^{(N)}}^T\theta^{(N)} + \gamma\mathbf{1}^T u^{(N)} - p^{\star(N)} \leq \frac{2N}{t} = \epsilon,$$

i.e., they are ϵ-suboptimal solutions of (1).

The algorithm for the second QP (3) is the same, except that we define $\phi_t^{(N)}$ as (10), *i.e.*, solve a sequence of centering problems

$$\text{minimize} \quad \phi_t^{(N)}(\alpha) = t(\tfrac{1}{2}\alpha^T X^{(N)}X^{(N)T}\alpha - \mathbf{1}^T\alpha) - \textstyle\sum_{i=1}^N \log\alpha_i, \tag{12}$$

with $t = N/\epsilon$. This requires only minor changes to the algorithm, so we will limit our further discussion of the algorithm to problem (1).

5. NEWTON'S METHOD

The key step in the algorithm is the solution of the unconstrained minimization problem (11). In this section we discuss the details of implementing Newton's method for (11). Since N is fixed throughout the section, we omit the superscripts N.

Each iteration of Newton's method consists of the following steps:

1. Compute the *Newton direction* $\Delta\alpha_{\rm nt} = -\nabla^2\phi_t(\alpha)^{-1}\nabla\phi_t(\alpha)$.
2. Compute the *Newton decrement* $\mu = (-\nabla\phi(\alpha))^T\Delta\alpha_{\rm nt})^{1/2}$. If $\mu^2/2 \leq \epsilon_{\rm nt}$, terminate and return α.

3. Choose a *step size* s by the following line search algorithm: Starting with $s := 1$, divide s by 2 until s satisfies the inequalities

$$0 < \alpha + s\Delta\alpha_{\rm nt} < \gamma\mathbf{1}, \qquad \phi_t(\alpha + s\Delta\alpha_{\rm n}) < \phi_t(\alpha) - 0.01\mu^2 s.$$

4. *Update:* $\alpha := \alpha + s\Delta\alpha_{\rm nt}$.

The only parameter in this algorithm outline is the tolerance $\epsilon_{\rm nt}$, which will be discussed in §5.2. The constants 2 and 0.01 in the line search are typical values, but can be changed to other values.[17] It can be shown that α satisfies $\phi_t(\alpha) - \inf \phi_t(\alpha) \leq \epsilon_{\rm nt}$ when the algorithm exits in step 2.[17]

5.1. The Newton equation

The key step in each iteration of Newton's method is the computation of $\Delta\alpha_{\rm nt}$, *i.e.*, the solution of the set of linear equations

$$\nabla^2\phi_t(\alpha)\Delta\alpha = -\nabla\phi_t(\alpha).$$

The Hessian of the function ϕ_t in (7) is given by

$$\nabla^2\phi_t(\alpha) = tXX^T + \mathbf{diag}(\gamma\mathbf{1} - \alpha)^{-2} + \mathbf{diag}(\alpha)^{-2},$$

so the Newton equation has the form

$$(tXX^T + D)\Delta\alpha = -g \tag{13}$$

where $D = \mathbf{diag}(\gamma\mathbf{1}-\alpha)^{-2} + \mathbf{diag}(\alpha)^{-2}$ is a positive diagonal matrix. The best method for solving (13) depends on the dimensions of X. We can distinguish two cases, using the terminology of Marron and Todd.[18]

- *High dimension low sample size data (HDLSS).* When $N \leq n$, we solve (13) using the Cholesky factorization of $tXX^T + D$. The cost of this method is $O(Nn)$ floating point operations for updating XX^T (or less, depending on the choice of basis functions), plus $(1/3)N^3$ for the Cholesky factorizations factorization of $tXX^T + D$.

- *High sample size low dimension data (HSSLD).* If $N > n$, we solve (13) by first solving

$$((1/t)I + X^TD^{-1}X)\Delta v = -(1/\sqrt{t})X^TD^{-1}g, \tag{14}$$

  and then computing $\Delta\alpha$ as

$$\Delta\alpha = -D^{-1}(g + \sqrt{t}X\Delta v).$$

  It is easily shown by substitution that the resulting $\Delta\alpha$ satisfies (13).

  The cost of this method is n^2N operations for constructing the matrix $(1/t)I + X^TD^{-1}X$, pluse $(1/3)n^3$ for the Cholesky factorization. Note that D changes at each iteration, so we cannot obtain $X^TD^{-1}X$ by a simple update from the previous Newton iteration (see however §5.3 below).

5.2. Incomplete centering

It is not necessary to compute the minimizers $\alpha(t)$ with great accuracy, *i.e.*, in step 2 of the Newton algorithm we can use a fairly high value of the exit tolerance $\epsilon_{\rm nt}$ (for example, $\epsilon_{\rm nt} = 10^{-4}$). We refer to this as *incomplete centering*. We will see later that the sequential analytic centering method with incomplete centering often requires very few Newton iterations per update. Since the solution of the Newton equations is the most expensive step in the algorithm, this allows us to substantially reduce the computation time.

Incomplete centering means, however, that the point $\theta = X^T\alpha$ may not be exactly feasible, and a correction is required. (Recall that in §4.1 we showed that $X^T\alpha(t)$ is primal feasible, where $\alpha(t)$ is on the central path.) Define

$$\theta = X^T(\alpha + \Delta\alpha_{\rm nt}), \qquad u_i = \frac{1}{t(\gamma - \alpha_i)}(1 + \frac{\Delta\alpha_{{\rm nt},i}}{\gamma - \alpha_i}), \quad i = 1, \ldots, N. \tag{15}$$

Using these definitions, the Newton equation at α,

$$(tXX^T + \mathbf{diag}(\gamma\mathbf{1} - \alpha)^{-2} + \mathbf{diag}(\alpha)^{-2})\Delta\alpha_{\rm nt} = -t(XX^T\alpha - \mathbf{1}) - \mathbf{diag}(\gamma\mathbf{1} - \alpha)^{-1}\mathbf{1} + \mathbf{diag}(\alpha)^{-1}\mathbf{1},$$

can be written as

$$X\theta = \mathbf{1} - u + \mathbf{diag}(t\alpha)^{-1}(I - \mathbf{diag}(\alpha)^{-1}\Delta\alpha_{\rm nt})\mathbf{1}.$$

It follows that θ, u are strictly primal feasible (*i.e.*, $u \geq 0$, $X\theta \geq \mathbf{1} - u$), provided $\alpha - \gamma\mathbf{1} \leq \Delta\alpha_{\rm nt} \leq \alpha$, *i.e.*, $\Delta\alpha_{\rm nt}$ is sufficiently small. If $\alpha = \alpha(t)$ (hence, $\Delta\alpha_{\rm nt} = 0$), the expressions (15) reduce to (8).

5.3. Approximate solution of Newton equations

Finally, we should mention the possibility of accelerating the algorithm by using approximate solutions of the Newton equations (13). It is observed in practice that when N is large, relatively few of the diagonal elements of D, which are given by

$$d_{ii} = \frac{1}{(\gamma - \alpha_i)^2} + \frac{1}{\alpha_i^2},$$

change significantly from one Newton iteration to the next. This suggests approximating the Hessian matrix $tXX^T + D$ at α, by a low-rank update of the Hessian matrix at the previous iterate. If the rank of the update is k, then the Cholesky factorization of the Hessian can be computed in $O(kN^2)$ operations, as opposed to the $O(N^3)$ operations needed to factor it from scratch.[19] Similar comments apply to the matrix $(1/t)I + X^TD^{-1}X$ that needs to be factored to solve (14).

6. NUMERICAL EXAMPLES

In the first example we solve the QP (3) for a linear classification problem in $\mathbf{R}^{50}$ (*i.e.*, $n = 51$). The training data in each class are randomly generated from a mixture of two normal distributions. In figure 1 we compare the number of Newton iterations to solve the centering problems (12) for $N = 1, \ldots, 1000$. The left plot shows the Newton iterations with a 'cold' starting point $\alpha_i = 10^{-3}/N$, $i = 1, \ldots, N$. The right plot shows the number of iterations with the 'warm' starting point $\alpha = (\alpha^{(N-1)}, \alpha_N)$, where

$$\alpha_N = \begin{cases} 1/(t(x_N^T\theta^{(N-1)} - 1)) & x_N^T\theta^{(N-1)} > 1 \\ 1 & \text{otherwise.} \end{cases}$$

This choice is motivated by the following considerations. Suppose $x_N^T\theta^{(N-1)} > 1$, *i.e.*, the new data point is correctly classified. Then it is reasonable to assume that $u_N(t) \approx 0$, $\theta(t) \approx \theta^{(N-1)}$, and hence, from (9),

$$\alpha_N(t) = \frac{1}{t(x_N^T\theta(t) - 1 + u_N(t))} \approx \frac{1}{t(x_N^T\theta^{(N-1)} - 1)},$$

so this provides a reasonable starting value for α_N. If $x_N^T\theta^{(N-1)} \leq 1$, we expect $\theta^{(N)}$ to be substantially different from $\theta^{(N-1)}$, and we take a default positive value for α_N.

Except for the starting point, the implementation of Newton's algorithm is identical for the two figures. We used $\epsilon_{\rm nt} = 10^{-4}$, and $t = 10^4N$. The final problem (for $N = 1000$) was solved with a relative accuracy of 0.2%.

Figure 1 clearly shows the benefit of using warm starts in the centering method. With a cold start the number of iterations is roughly constant for large N, ranging between 15 and 20. With the warm start, only one or even zero iterations are needed for most updates. Occasionally a larger number of iterations is required. This occurs

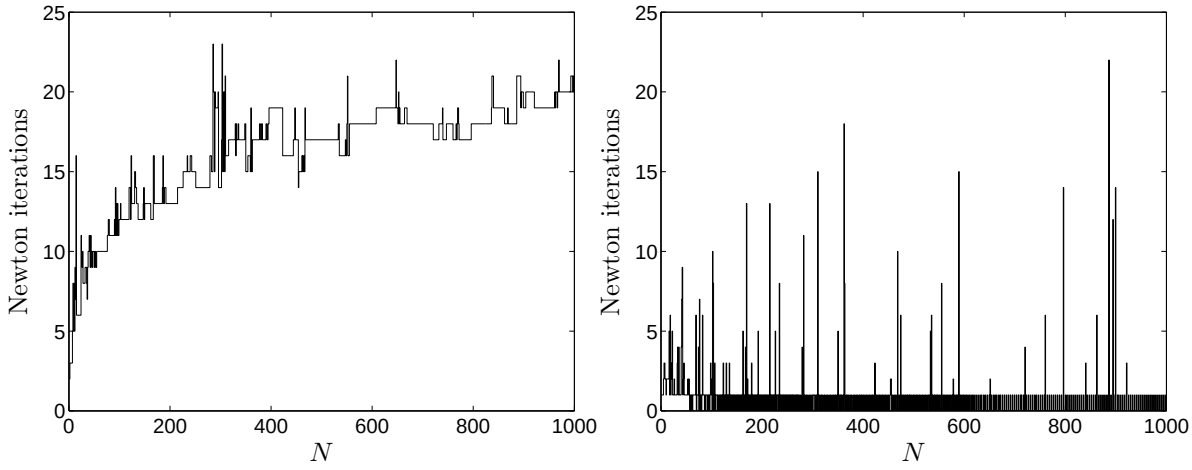

Figure 1. Number of Newton iterations in the sequential analytic centering method for a linear classification problem in $\mathbf{R}^{50}$. The left plot shows the number of iterations with a cold start. The right plot shows the number of iterations with a warm start, using the previous solution as starting point.

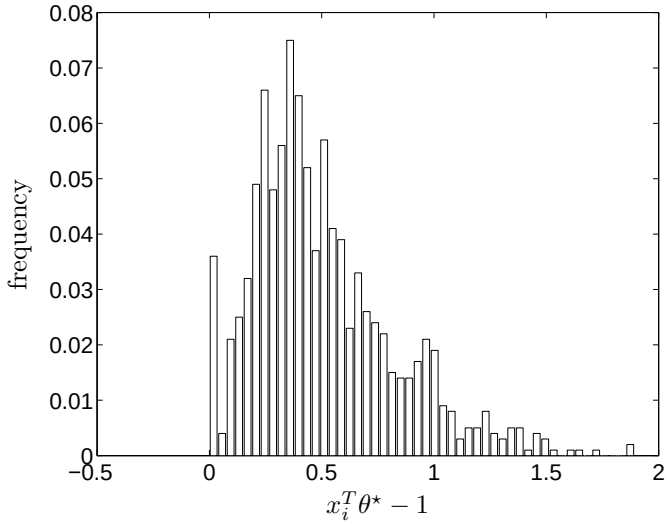

Figure 2. Distribution of the slacks $X\theta^\star - 1$ for a separable data set, where $\theta^\star$ is the maximum margin linear classifier, *i.e.*, the solution of the QP (3).

when the new data point is misclassified by the previous classifier $x_N^T\theta^{(N-1)} < 1$, and as a result $\theta^{(N)}$ differs substantially from $\theta^{(N-1)}$.

Figure 2 shows the histogram of the slacks $x_i^T\theta^\star - 1$ at the optimal solution $\theta^\star$ for $N = 1000$. At the optimum about 3.5% of the data points lie in the hyperplane $\theta^T x = 1$.

Figures 3 and 4 show the results of a similar experiment in which we solve the SVM training problem (1) for a linear classification problem in $\mathbf{R}^{200}$. In this problem we attempt to discriminate between two types of vehicles based on the magnitude spectrum of their acoustic signals. A total of 782 training vectors were acquired. The cold and warm start parameters are set in precisely the same manner as in the previous experiment. We used $\epsilon_{\rm nt} = 10^{-4}$, $t = 2\,10^2 N$, and $\gamma = 1$. As seen in figure 4, the data are nearly separable. At the optimum, about

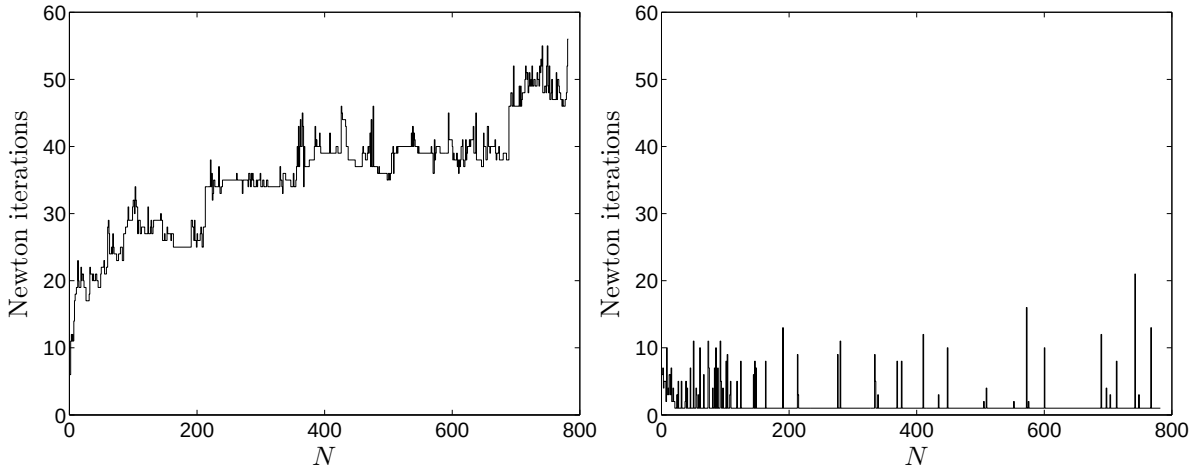

Figure 3. Number of Newton iterations in the sequential analytic centering method for a linear classification problem in $\mathbf{R}^{200}$. The left plot shows the number of iterations with a cold start. The right plot shows the number of iterations with a warm start, using the previous solution as starting point.

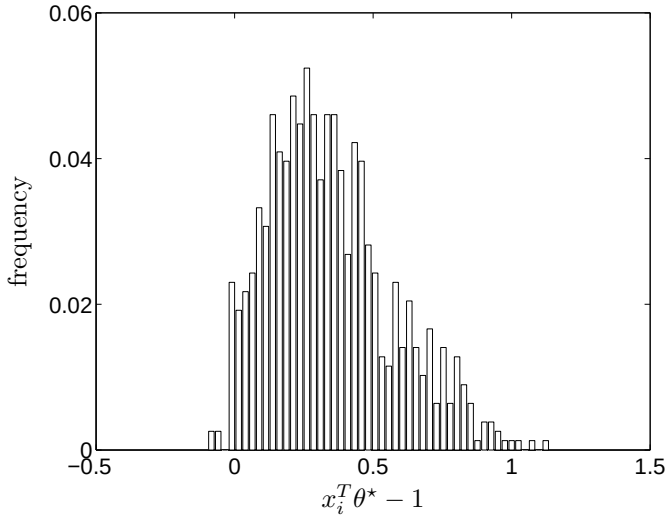

Figure 4. Distribution of the slacks $X\theta^\star - 1$ for a nearly separable data set, where $\theta^\star$ is the maximum margin linear classifier, *i.e.*, the solution of the QP (1).

2.3% of the data points lie in the hyperplane $\theta^T x = 1$. The final problem (for $N = 782$) was solved with a relative accuracy of 0.2%.

7. CONCLUSIONS

We have described a recursive method for SVM training, based on dual analytic centering. Numerical experiments demonstrate that the analytic centering method can take advantage of warm starts and quickly update the classifier when new data points are added. This is an advantage over standard training algorithms based on general-purpose interior-point methods.

Among the potential improvements and extensions that we plan to study are adaptive strategies for selecting the centering parameter t, the use of primal or primal-dual methods, methods based on self-dual embeddings,[13] and rigorous methods for pruning data sets.

Similar techniques should also be useful for target classification problems with slowly changing target characteristics and for updating classifiers when different data sets are merged.

ACKNOWLEDGMENTS

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA), and administered by the Army Research Office under ESP MURI Award No. DAAD19-01-1-0504. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), and Army Research Office.

REFERENCES

1. V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, 1995.
2. B. Schölkopf and A. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.
3. S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.
4. R. J. Vanderbei, *Linear Programming: Foundations and Extensions*, Kluwer Academic Publishers, 1997.
5. R. Vanderbei, "LOQO: An interior point code for quadratic programming," *Optimization Methods and Software* **11**, pp. 451–484, 1999.
6. *MOSEK v2.0 User's manual*, 2002. Available from `http://www.mosek.com`.
7. E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Proceedings of IEEE NNSP'97*, pp. 276–285, 1997.
8. E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proceedings of CVPR '97*, 1997.
9. S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *Journal of Machine Learning Research* **2**, pp. 243–264, 2001.
10. Y. Censor, "Row-action techniques for huge and sparse systems and their applications," *SIAM Review* **23**, pp. 444–466, 1980.
11. J.-L. Goffin and J.-P. Vial, "Shallow, deep and very deep cuts in the analytic center cutting plane method," *Mathematical Programming* **84**, pp. 89–103, Jan. 1999.
12. Z.-Q. Luo and J. Sun, "An analytic center based column generation algorithm for convex quadratic feasibility problems," *SIAM J. on Optimization* **9**, pp. 217–235, 1998.
13. Y. Ye, *Interior Point Algorithms: Theory and Analysis*, Discrete Mathematics and Optimization, Wiley, New York, 1997.
14. E. Bai, Y. Ye, and R. Tempo, "Bounded error parameter estimation: a sequential analytic center approach," *IEEE Trans. Aut. Control* **44**(6), pp. 1107–1117, 1999.
15. E. Bai, M. Fu, R. Tempo, and Y. Ye, "Convergence results of the analytic center estimator," *IEEE Trans. Aut. Control* **45**(3), pp. 569–572, 2000.
16. Y. Nesterov and A. Nemirovsky, *Interior-point polynomial methods in convex programming*, vol. 13 of *Studies in Applied Mathematics*, SIAM, Philadelphia, PA, 1994.
17. S. Boyd and L. Vandenberghe, *Convex Optimization*, 2003. To be published. Draft available as course reader, Stanford University and UCLA.
18. J. S. Marron and M. Todd, "Distance weighted discrimination," tech. rep., School of Operations Research and Industrial Engineering, Cornell University, 2003.
19. P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, "Methods for modifying matrix factorizations," *Mathematics of Computation* **28**, pp. 505–535, 1974.