A Method for Large-Scale ℓ_1-Regularized Logistic Regression

Kwangmoo Koh and **Seung-Jean Kim** and **Stephen Boyd**
Electrical Engineering Department
Stanford University
Stanford, CA 94305

Abstract

Logistic regression with ℓ_1 regularization has been proposed as a promising method for feature selection in classification problems. Several specialized solution methods have been proposed for ℓ_1-regularized logistic regression problems (LRPs). However, existing methods do not scale well to large problems that arise in many practical settings. In this paper we describe an efficient interior-point method for solving ℓ_1-regularized LRPs. Small problems with up to a thousand or so features and examples can be solved in seconds on a PC. A variation on the basic method, that uses a preconditioned conjugate gradient method to compute the search step, can solve large sparse problems, with a million features and examples (*e.g.*, the 20 Newsgroups data set), in a few tens of minutes, on a PC. Numerical experiments show that our method outperforms standard methods for solving convex optimization problems as well as other methods specifically designed for ℓ_1-regularized LRPs.

Introduction

Logistic regression

Let $x \in \mathbf{R}^n$ denote a vector of feature variables, and $b \in \{-1, +1\}$ denote the associated binary output. In the logistic model, the conditional probability of b, given x, has the form

$$\mathrm{Prob}(b|x) = 1/(1 + \exp\left(-b(w^T x + v)\right)).$$

The parameters of this model are $v \in \mathbf{R}$ (the intercept) and $w \in \mathbf{R}^n$ (the weight vector).

Suppose we are given a set of training or observed examples, $(x_i, b_i) \in \mathbf{R}^n \times \{-1, +1\}$, $i = 1, \ldots, m$, assumed to be independent samples from a distribution. The model parameters w and v can be found by maximum likelihood estimation from the observed examples. The maximum likelihood estimate minimizes the average loss

$$l_{\mathrm{avg}}(v, w) = (1/m) \sum_{i=1}^{m} f(w^T a_i + v b_i),$$

where $a_i = b_i x_i \in \mathbf{R}^n$ and f is the logistic loss function defined by $f(z) = \log(1+\exp(-z))$. Finding the maximum likelihood estimate of v and w is called *logistic regression* (LR).

ℓ_1-regularized logistic regression

Recently, ℓ_1-*regularized logistic regression* has received much attention as a promising method for feature selection. The ℓ_1-regularized logistic regression problem (LRP) can be formulated as

$$\text{minimize} \quad (1/m) \textstyle\sum_{i=1}^{m} f(w^T a_i + v b_i) + \lambda \|w\|_1, \qquad (1)$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm, *i.e.*, $\|w\|_1 = \sum_{i=1}^{n} |w_i|$, and $\lambda > 0$ is the regularization parameter. The problem data are

$$A = [a_1 \ \cdots \ a_m]^T \in \mathbf{R}^{m \times n}, \quad b = [b_1 \ \cdots \ b_m]^T \in \mathbf{R}^m.$$

The main motivation is that ℓ_1-regularized logistic regression typically yields a *sparse* vector w, *i.e.*, w typically has relatively few nonzero coefficients. Recent studies show that ℓ_1-regularized LR can outperform ℓ_2-regularized LR, especially when the number of observations is smaller than the number of features (Ng 2004).

In the literature, ℓ_1-regularized logistic regression has been used in a variety of applications, such as text categorization (Genkin, Lewis, & Madigan 2006), graphical model selection (Wainwright, Ravikumar, & Lafferty 2007), author identification (Madigan *et al.* 2005), and gene selection in cancer classification (Shevade & Keerthi 2003; Cawley & Talbot 2006). In these applications, often the number of features is very large (exceeding one million).

Related Work

The objective function in the ℓ_1-regularized LRP (1) is convex, but not differentiable. Generic methods for nondifferentiable convex problems can be used, such as the ellipsoid method or subgradient methods (Shor 1985). These methods are usually very slow in practice, however.

Faster methods are based on transforming the problem to an equivalent one, with linear inequality constraints,

$$\begin{array}{ll} \text{minimize} & (1/m) \sum_{i=1}^{m} f(w^T a_i + v b_i) + \lambda \mathbf{1}^T u \\ \text{subject to} & -u_i \le w_i \le u_i, \quad i = 1, \ldots, n, \end{array} \qquad (2)$$

where the variables are the original ones $v \in \mathbf{R}$, $w \in \mathbf{R}^n$, along with $u \in \mathbf{R}^n$. Here $\mathbf{1}$ denotes the vector with all components one, so $\mathbf{1}^T u$ is the sum of the components of u. The reformulated problem (2) is a convex optimization problem, with a smooth objective, and linear constraint functions, so it can be solved by standard convex optimization methods such as SQP, augmented Lagrangian, interior-point, and other methods. High quality solvers that can directly handle (2) include for example MOSEK (MOSEK ApS 2002), and NPSOL (Gill *et al.* 1986). These general purpose solvers can solve small and medium scale ℓ_1-regularized LRPs effectively, but do not scale well to large problems.

Recently, several researchers have developed path-following methods for ℓ_1-regularized LRPs (Park & Hastie 2006; Rosset 2005). When the solution of (1) is extremely sparse, path-following methods can be very fast. Otherwise, path-following methods are slow, especially for large-scale problems. Other recent work on computational methods for ℓ_1-regularized logistic regression includes the iteratively reweighted least squares (IRLS) method (Lee *et al.* 2006; Lokhorst 1999), a generalized LASSO method (Roth 2004) that extends the LASSO method proposed by (Osborne, Presnell, & Turlach 2000) to generalized linear models, generalized iterative scaling (Goodman 2004), bound optimization algorithms (Krishnapuram *et al.* 2005), online algorithms (Balakrishnan & Madigan 2005; Perkins & Theiler 2003), coordinate descent methods (Genkin, Lewis, & Madigan 2006), and the Gauss-Seidel method (Shevade & Keerthi 2003).

The purpose

The main purpose of this paper is to describe a specialized interior-point method for solving the ℓ_1-regularized logistic regression problem that is very efficient, for all size problems. The method can solve with high accuracy large sparse problems with a million features and examples, in a few tens of minutes on a PC. Extensive comparison with many existing methods shows that our method is most efficient for large problems, and for small and medium problems as well, reliably providing very accurate solutions. The efficiency of the method comes mainly from the use of an effective preconditioner which has very low computational overhead but accelerates the convergence significantly.

Preliminaries

In this section, we give some preliminaries needed later.

Optimality conditions

A standard result in convex optimization is that a point x minimizes a convex function $f(x)$ (which is not necessarily differentiable at all points) if and only if $0 \in \partial f(x)$, where $\partial f(x)$ is the subdifferential of f at x (Shor 1985). Using subdifferential calculus, we can show that the necessary and sufficient conditions for (v, w) to be optimal for the ℓ_1-regularized LRP (1) are

$$b^T p(v,w) = 0, \qquad (3)$$

$$(1/m)(A^T p(v,w))_i \in \begin{cases} \{+\lambda\} & w_i > 0, \\ \{-\lambda\} & w_i < 0, \\ [-\lambda,\, \lambda] & w_i = 0, \end{cases} \qquad (4)$$

for $i = 1, \ldots, n$. Here, we use the notation

$$p(v,w)_i = \frac{1}{1+\exp(w^T a_i + v b_i)}, \quad i = 1, \ldots, m.$$

Let us analyze when a pair of the form $(v, 0)$ is optimal. By plugging $(v, 0)$ into (3) we get $v = \log(m_+/m_-)$, where m_+ is the number of training examples with outcome 1 (called positive) and m_- is the number of training examples with outcome -1 (called negative). We can see from (4) that if

$$\lambda \geq \lambda_{\max} = \|(1/m) A^T p(\log(m_+/m_-), 0)\|_\infty, \qquad (5)$$

then the logistic model obtained from ℓ_1-regularized LR has weight zero. Here $\|\cdot\|_\infty$ denotes the ℓ_∞-norm, *i.e.*, $\|w\|_\infty = \max(w_1, \ldots, w_n)$. The number $\lambda_{\max}$ gives us an upper bound on the useful range of the regularization parameter λ: For any larger value of λ, the logistic model obtained from ℓ_1-regularized LR has weight zero, which has no ability to classify.

Dual problem

To derive a Lagrange dual of the ℓ_1-regularized LRP (1), we first introduce a new variable $z \in \mathbf{R}^m$, as well as new equality constraints $z_i = w^T a_i + v b_i$, $i = 1, \ldots, m$, to obtain the equivalent problem

$$\begin{array}{ll} \text{minimize} & (1/m)\sum_{i=1}^m f(z_i) + \lambda \|w\|_1 \\ \text{subject to} & z_i = w^T a_i + v b_i, \quad i = 1, \ldots, m. \end{array}$$

Associating dual variables $\nu_i \in \mathbf{R}$ with the equality constraints, we have the following Lagrange dual of the ℓ_1-regularized LRP (1):

$$\begin{array}{ll} \text{maximize} & G(\nu) = -(1/m)\sum_{i=1}^m f^*(-m\nu_i) \\ \text{subject to} & \|A^T \nu\|_\infty \leq \lambda, \quad b^T \nu = 0, \end{array} \qquad (6)$$

where f^* is the *conjugate* of the logistic loss function f:

$$f^*(y) = \begin{cases} -y\log(-y) + (1+y)\log(1+y), & -1 \leq y \leq 0 \\ \infty, & \text{otherwise}, \end{cases}$$

with the interpretation $0 \log 0 = 0$.

The dual problem (6) is a convex optimization problem with variable $\nu \in \mathbf{R}^m$, and has the form of an ℓ_∞-norm constrained maximum generalized entropy problem. We say that $\nu \in \mathbf{R}^m$ is *dual feasible* if it satisfies $\|A^T \nu\|_\infty \leq \lambda$, $b^T \nu = 0$. Any dual feasible point ν gives a lower bound on the optimal value $p^\star$ of the primal ℓ_1-regularized LRP (1): $G(\nu) \leq p^\star$. Furthermore, $G(\nu^\star) = p^\star$, where $\nu^\star$ is the optimal solution of (6).

We can relate a primal optimal point $(v^\star, w^\star)$ and a dual optimal point $\nu^\star$ to the optimality condition (3) and (4). They are related by

$$\nu^\star = (1/m) p(v^\star, w^\star).$$

Suboptimality bound

We now derive an easily computed bound on the suboptimality of a pair (v, w), by constructing a dual feasible point $\bar{\nu}$ from an arbitrary w. This bound will be incorporated in the stopping rule of the interior-point method described in the next section.

Define $\bar{v}$ as

$$\bar{v} = \arg\min_{v} l_{\text{avg}}(v, w), \qquad (7)$$

i.e., $\bar{v}$ is the optimal intercept for the weight vector w, characterized by $b^T p(\bar{v}, w) = 0$. Note that $\bar{v}$ is the solution of an one-dimensional smooth convex optimization problem, which can be solved very efficiently. Now, we define $\bar{\nu}$ as

$$\bar{\nu} = (s/m)p(\bar{v}, w), \qquad (8)$$

where $s = \min\left\{m\lambda/\|A^T p(\bar{v}, w)\|_\infty, 1\right\}$ is a scaling constant. Evidently $\bar{\nu}$ is dual feasible, so $G(\bar{\nu})$ is a lower bound on $p^\star$, the optimal value of the ℓ_1-regularized LRP (1).

The difference between the primal objective value of (v, w), and the associated lower bound $G(\bar{\nu})$, is called the *duality gap*, and denoted $\eta(v, w)$:

$$\eta(v, w) \quad = \quad l_{\text{avg}}(v, w) + \lambda\|w\|_1 - G(\bar{\nu}) \qquad (9)$$

We always have $\eta(v, w) \geq 0$, and the point (v, w) is no more than $\eta(v, w)$-suboptimal. At the optimal point $(v^\star, w^\star)$, the duality gap is zero.

Logarithmic barrier and central path

The *logarithmic barrier* for the bound constraints $-u_i \leq w_i \leq u_i$ of the equivalent problem (2) is

$$\Phi(w, u) = -\sum_{i=1}^{n} \log(u_i + w_i) - \sum_{i=1}^{n} \log(u_i - w_i),$$

defined on $\{(w, u) \in \mathbf{R}^n \times \mathbf{R}^n \mid |w_i| < u_i,\ i = 1, \ldots, n\}$. The logarithmic barrier function is smooth and convex. We augment the weighted objective function of (2) by the logarithmic barrier, to obtain

$$\phi_t(v, w, u) = t l_{\text{avg}}(v, w) + t\lambda \mathbf{1}^T u + \Phi(w, u), \qquad (10)$$

where $t > 0$ is a parameter. This function is smooth, strictly convex, and bounded below, and so has a unique minimizer which we denote $(v^\star(t), w^\star(t), u^\star(t))$. This defines a curve in $\mathbf{R} \times \mathbf{R}^n \times \mathbf{R}^n$, parameterized by t, called the *central path*.

With the point $(v^\star(t), w^\star(t), u^\star(t))$ we associate

$$\nu^\star(t) = (1/m)p(v^\star(t), w^\star(t)), \qquad (11)$$

which can be shown to be dual feasible. Indeed, it coincides with the dual feasible point $\bar{\nu}$ constructed from $w^\star(t)$ using (8). As a standard result in convex optimization, $(v^\star(t), w^\star(t))$ is no more than $2n/t$-suboptimal (Boyd & Vandenberghe 2004, §11).

Our Method

In a primal interior-point method, we compute a sequence of points on the central path, for an increasing sequence of values of t, using Newton's method to minimize $\phi_t(v, w, u)$, starting from the previously computed central point. Using our method for cheaply computing a dual feasible point and associated duality gap for *any* (v, w), we can construct a custom interior-point method that updates t at each iteration.

```
INTERIOR-POINT METHOD FOR $\ell_1$-REGULARIZED LR.
given tolerance $\epsilon > 0$, parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$
Initialize. $t := 1/\lambda$, $v := \log(m_+/m_-)$, $w := 0$, $u := \mathbf{1}$.
repeat
  1. Compute search direction. Solve the Newton system
     $\nabla^2 \phi_t(v, w, u) \begin{bmatrix} \Delta v \\ \Delta w \\ \Delta u \end{bmatrix} = -\nabla \phi_t(v, w, u).$
  2. Backtracking line search.
     Find the smallest integer $k \geq 0$ that satisfies
     $\phi_t(v + \beta^k \Delta v, w + \beta^k \Delta w, u + \beta^k \Delta u)$
     $\leq \phi_t(v, w, u) + \alpha\beta^k \nabla \phi_t(v, w, u)^T \begin{bmatrix} \Delta v \\ \Delta w \\ \Delta u \end{bmatrix}.$
  3. Update. $(v, w, u) := (v, w, u) + \beta^k(\Delta v, \Delta w, \Delta u)$.
  4. Set $v := \bar{v}$, the optimal value of the intercept, as in (7).
  5. Construct dual feasible point $\nu$ from (8).
  6. Evaluate duality gap $\eta$ from (9).
  7. quit if $\eta \leq \epsilon$.
  8. Update $t$.
```

The choice $v = \log(m_+/m_-)$ is the optimal value of v when $w = 0$ and $u = \mathbf{1}$, and the choice $t = 1/\lambda$ minimizes $\|(1/t)\nabla\phi_t(\log(m_+/m_-), 0, \mathbf{1})\|_2$. Typical values for the line search parameters are $\alpha = 0.01$, $\beta = 0.5$. The choice of the initial values and the parameters does not greatly affect performance.

Update rule

The update rule we propose is

$$t := \begin{cases} \max\left\{\mu \min\{\hat{t}, t\}, t\right\}, & s \geq s_{\min} \\ t, & s < s_{\min} \end{cases} \qquad (12)$$

where $\hat{t} = 2n/\eta$ and $s = \beta^k$ is the step length chosen in the line search. Here $\mu > 1$ and $s_{\min} \in (0, 1]$ are algorithm parameters; we have found good performance with $\mu = 2$ and $s_{\min} = 0.5$.

To explain the update rule (12), we first give an interpretation of $\hat{t}$. If (v, w, u) is on the central path, the duality gap is $\eta = 2n/t$. Thus $\hat{t}$ is the value of t for which the associated central point has the same duality gap as the current point. We use the step length s as a crude measure of proximity to the central path, so we increase t by a factor μ when the current point is near the central path ($s \geq s_{\min}$ and $\hat{t} \approx t$). (See the longer version of the paper (Koh, Kim, & Boyd 2006) for more on the update rule and convergence of the resulting interior-point method.) Compared with standard update rules, the update rule is quite robust and works well when combined with the PCG algorithm we will describe soon.

Computing the search direction

The Newton system can be written as

$$\begin{bmatrix} t b^T D_0 b & t b^T D_0 A & 0 \\ t A^T D_0 b & t A^T D_0 A + D_1 & D_2 \\ 0 & D_2 & D_1 \end{bmatrix} \begin{bmatrix} \Delta v \\ \Delta w \\ \Delta u \end{bmatrix} = - \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix}, \qquad (13)$$

where

$$
\begin{aligned}
D_0 &= \operatorname{diag}\left(\frac{f''(w^Ta_1+vb_1)}{m},\ldots,\frac{f''(w^Ta_m+vb_m)}{m}\right),\\
D_1 &= \operatorname{diag}\left(\frac{2(u_1^2+w_1^2)}{(u_1^2-w_1^2)^2},\ldots,\frac{2(u_n^2+w_n^2)}{(u_n^2-w_n^2)^2}\right),\\
D_2 &= \operatorname{diag}\left(\frac{-4u_1w_1}{(u_1^2-w_1^2)^2},\ldots,\frac{-4u_nw_n}{(u_n^2-w_n^2)^2}\right),\\
g_1 &= -(t/m)b^T(\mathbf{1}-p_{\log}(v,w)) \in \mathbf{R},\\
g_2 &= -(t/m)A^T(\mathbf{1}-p_{\log}(v,w)) + q(w,u) \in \mathbf{R}^n,\\
g_3 &= t\lambda\mathbf{1} - q(u,w) \in \mathbf{R}^n.
\end{aligned}
$$

Here, we use $\operatorname{diag}(z_1,\ldots,z_m)$ to denote the diagonal matrix with diagonal entries $z_1,\ldots,z_m$, where $z_i \in \mathbf{R}$, $i = 1,\ldots,m$. We also use the notation

$$q(w,u)_i = 2w_i/(u_i^2-w_i^2),\ i=1,\ldots,n.$$

The computational effort per iteration is dominated by step 1, the search direction computation. The Newton system can be solved by direct methods (*e.g.*, the Cholesky factorization) and iterative methods (*e.g.*, conjugate gradients). Direct methods are effective for small and medium dense problems. For large problems, however, direct methods are not computationally practical, and iterative methods are far more efficient. In using an iterative method to compute the search direction, we need to find a search direction which is good enough in terms of the trade-off of the computational complexity versus the convergence rate it provides.

Direct methods We first eliminate Δu from (13) to obtain the reduced Newton system

$$\begin{bmatrix} tb^TD_0b & tb^TD_0A \\ tA^TD_0b & tA^TD_0A+D_3 \end{bmatrix}\begin{bmatrix}\Delta v\\ \Delta w\end{bmatrix} = -\begin{bmatrix} g_1\\ g_4\end{bmatrix}, \tag{14}$$

where $D_3 = D_1 - D_2D_1^{-1}D_2$ and $g_4 = g_2 - D_2D_1^{-1}g_3$. Once this reduced system is solved, Δu can be recovered as $\Delta u = -D_1^{-1}(g_3 + D_2\Delta w)$. When $m < n$, *i.e.*, there are fewer examples than features, the matrix in (14) is a diagonal matrix plus a rank $m+1$ matrix, so we can use the Sherman-Morrison-Woodbury (SMW) formula to solve the reduced Newton system (14). We start by eliminating Δw from (14) to obtain

$$
\begin{aligned}
&(tb^TD_0b - t^2b^TD_0AS^{-1}A^TD_0b)\Delta v\\
&\quad = \ -g_1 + tb^TD_0AS^{-1}g_4,
\end{aligned}
$$

where $S = tA^TD_0A + D_3$. By the SMW formula, the inverse of S is given by

$$S^{-1} = D_3^{-1} - D_3^{-1}A^T\left((1/t)D_0^{-1} + AD_3^{-1}A^T\right)^{-1}AD_3^{-1}.$$

We can now calculate Δv via Cholesky factorization of the matrix $\left((1/t)D_0^{-1} + AD_3^{-1}A^T\right)$ and two backsubstitutions (Boyd & Vandenberghe 2004, App. C). Once we compute Δv, we can compute the other components of the search direction as

$$\Delta w = -S^{-1}(g_4 + tA^TD_0b\Delta v),\ \Delta u = -D_1^{-1}(g_3 + D_2\Delta w).$$

The number of flops needed to compute the search direction using direct methods is $O(\min(n,m)^2\max(n,m))$.

Computing search direction via PCG For large problems, we compute the search direction approximately, using a preconditioned conjugate gradients (PCG) algorithm, which uses a symmetric positive definite preconditioner $P \in \mathbf{R}^{2n+1\times 2n+1}$ (Demmel 1997, §6.6). To describe the preconditioner, we note that the Hessian can be written as $H = t\nabla^2 l_{\text{avg}}(v,w) + \nabla^2\Phi(w,u)$. The preconditioner approximates the Hessian of tl_{avg} at (v,w) with its diagonal entries, while retaining the Hessian of the logarithmic barrier:

$$
\begin{aligned}
P &= \operatorname{diag}\left(t\nabla^2 l_{\text{avg}}(v,w)\right) + \nabla^2\Phi(w,u)\\
&= \begin{bmatrix} d_0 & 0 & 0\\ 0 & D_3 & D_2\\ 0 & D_2 & D_1\end{bmatrix},
\end{aligned}
$$

where $d_0 = tb^TD_0b$ and $D_3 = \operatorname{diag}(tA^TD_0A) + D_1$.

The PCG algorithm needs a good initial search direction and an effective termination rule.

- *Initial point.* There are many choices for the initial search direction, such as negative gradient, 0, or the search direction found in the previous iteration of the method. The previous search direction appears to have a small advantage over the negative gradient and 0.
- *Termination rule.* The PCG algorithm terminates when the cumulative number of PCG iterations exceeds the given limit N_{pcg} or we compute a point with relative tolerance less than ϵ_{pcg}. We propose to change the relative tolerance adaptively as

  $$\epsilon_{\text{pcg}} = \min\{0.1, \xi\eta/\|g\|_2\},$$

  where g is the gradient and η is the duality gap at the current iterate. Here, ξ is an algorithm parameter. We have found that $\xi = 0.3$ works well for a wide range of problems. That is, we solve the Newton system with low accuracy (but never worse than 10%) at early iterations, and solve it more accurately as the duality gap decreases. Since the convergence of the PCG algorithm is usually very fast, there is no significant effect of N_{pcg} on the overall performance, as long as the limit is set to a large value.

Each iteration of the PCG algorithm involves a handful of inner products, the matrix-vector product Hp and a solve step with P in computing $P^{-1}r$, where $p \in \mathbf{R}^{2n+1}$ and $r \in \mathbf{R}^{2n+1}$. We can compute Hp in the PCG algorithm using

$$Hp = \begin{bmatrix} b^Tu\\ A^Tu + D_1p_2\\ D_2p_2 + D_1p_3\end{bmatrix},$$

where $u = tD_0(bp_1 + Ap_2) \in \mathbf{R}^m$ and $p = (p_1,p_2,p_3)$. The cost of computing Hp is $O(s)$ flops when A is sparse with s nonzero elements. (We assume $s \geq n$, which holds if each example has at least one nonzero feature.) We can compute $P^{-1}r$ with $r = (r_1,r_2,r_3)$ as

$$P^{-1}r = \begin{bmatrix} r_1/d_0\\ (D_1D_3 - D_2^2)^{-1}(D_1r_2 - D_2r_3)\\ (D_1D_3 - D_2^2)^{-1}(-D_2r_2 + D_3r_3)\end{bmatrix}.$$

The computational cost is $O(n)$ flops. In short, the computational effort of each iteration of the PCG algorithm is dominated by the matrix-vector product Hp.

The computational effort of the interior-point method that uses the PCG algorithm to compute the search direction is the product of the total number of PCG steps required over all iterations and the cost of a PCG step. In extensive testing, we found the method to be very efficient, requiring a total number of PCG steps ranging between a few hundred (for medium size problems) and several thousand (for large problems).

Numerical Experiments

In this section we compare the performance of our method and three existing methods for ℓ_1-regularized LR, on various types of data sets. Our method is implemented in Matlab and C, and the C implementation, which is more efficient than Matlab implementation (especially for sparse problems), is available online (`www.stanford.edu/~boyd/l1_logreg`).

Experimental setup

Existing methods In our comparison study, we considered three existing methods: `MOSEK` (MOSEK ApS 2002), `BBR` (Genkin, Lewis, & Madigan 2006), and `IRLS` (Lee *et al.* 2006). `MOSEK` is a high quality implementation of interior-point method for convex optimization, whose efficiency over other standard solvers has been confirmed by a recent benchmark study available at `http://plato.asu.edu/ftp/dimacs_sdp.html`. `BBR` is a Gauss-Seidel type first-order method that scales to large problems and has been used in practical applications such as gene classification and text classification (Setakis, Stirnadel, & Balding 2006). `IRLS` is a recently developed method that outperforms many other existing methods including the generalized Lasso (Roth 2004) and iterative scaling methods (Goodman 2004; Perkins & Theiler 2003). For these reasons, we believe that our comparison results shown below are fairly extensive.

Data sets For comparison, various types of data including small, medium and large, dense and sparse data sets were taken from the UCI benchmark repository and other sources: leukemia cancer gene expression data (Golub *et al.* 1999), colon tumor gene expression data (Alon *et al.* 1999), ionosphere data and spambase data (Newman *et al.* 1998), and 20 Newsgroup data (Lang 1995). We processed the 20 Newsgroup data in a way similar to (Keerthi & DeCoste 2005). The positive class consists of the 10 groups with names of form sci.*, comp.*, and misc.forsale, and the negative class consists of the other 10 groups. We used McCallum's Rainbow program (McCallum 1996) to tokenize the (text) data set with options specifying trigrams, skip message headers, no stoplist, and drop terms occurring fewer than two times.

We also generated 21 data sets, with the number of features n varying from one hundred to ten millions, and $m = 0.1n$ examples. Each example has an equal number of positive and negative examples. Features of positive (negative) examples are independent and identically distributed, drawn from a normal distribution $\mathcal{N}(1,1)$ ($\mathcal{N}(-1,1)$), with an average number of nonzero features per example 30. The data sets were standardized to make each column of A have zero mean and unit variance. Table 1 summarizes the statistics of the data sets used in comparison.

Data	n	m	Type
Leukemia	7129	38	dense
Colon	2000	62	dense
Ionosphere	34	351	dense
Spambase	57	4061	dense
Internet Ads.	1430	2359	sparse
20 Newsgroups	777811	11314	sparse
Rand 1	100	10	sparse
Rand 2	180	18	sparse
Rand 3	320	32	sparse
· · ·	· · ·	· · ·	· · ·
Rand 19	3162280	316228	sparse
Rand 20	5623420	562342	sparse
Rand 21	10000000	1000000	sparse

Table 1: Dense and sparse data sets used in comparison.

Stopping criteria `MOSEK` uses the duality gap as the stopping criterion like our method. The original implementation of `IRLS` does not have a stopping criterion, so we modified the code to use the suboptimality bound described above as the stopping criterion. For a given tolerance value, `MOSEK`, (modified) `IRLS`, and our method terminate with the same provable bound on the suboptimality (duality gap $< 10^{-8}$), so the three methods give solutions with a similar accuracy. The stopping criterion of `BBR` is based on the fractional improvement. Since we could not modify the C implementation of `BBR`, we used the original stopping criterion with tolerance 10^{-10}. For each data set, the solution computed by `BBR` is far less accurate than those by the other existing methods and our method with modest tolerance (duality gap).

Regularization parameters The choice of the regularization parameter greatly affects the runtime of the methods. For each data set, we considered two values of the regularization parameter λ: $0.1\lambda_{\max}$ and $0.001\lambda_{\max}$. (For most standardized problems, the interval $[0.001\lambda_{\max}, 0.1\lambda_{\max}]$ covers the range of practical interest.)

Comparison details To compute the search direction, our method uses direct methods for dense problems and the PCG algorithm for sparse problems. `MOSEK` and `BBR` are implemented in C or Fortran, and `IRLS` is implemented in Matlab. To ensure fair comparison with `IRLS`, we use the Matlab implementation of our method denoted IPM (M), and we compare the C implementation denoted IPM (C) with the other two methods.

Computational environment For small and medium problems, the existing methods and our method were run on a 3.2GHz Pentium IV, 1GB RAM under Linux. For large problems, the methods were run on AMD Opteron 254, 8GB RAM under Linux.

Data	λ	IPM (C)	MOSEK	BBR	IPM (M)	IRLS
Leuk-	λ_1	0.62	9.63	36.2	1.19	1.64
emia	λ_2	0.57	22.41	64.5	1.10	4.72
Colon	λ_1	0.25	33.6	9.23	0.46	0.80
	λ_2	0.28	139.7	57.3	0.53	4.18
Iono-	λ_1	0.02	0.19	0.25	0.06	0.06
sphere	λ_2	0.03	0.47	1.73	0.07	0.32
Spam-	λ_1	0.66	6.64	3.05	1.11	1.28
base	λ_2	0.72	20.4	264	1.20	4.53
Internt	λ_1	0.30	85.3	165	3.08	40.2
Ads.	λ_2	1.46	103.1	4310	14.5	-

Table 2: Runtime (in seconds) of the C and Matlab implementation of our method, MOSEK, BBR, IRLS on benchmark data sets with two regularization parameters: $\lambda_1 = 10^{-1}\lambda_{\max}$ and $\lambda_2 = 10^{-3}\lambda_{\max}$.

$\lambda/\lambda_{\max}$	PCG iterations	Time (sec)
0.5	558	134
0.1	1036	256
0.05	2090	501

Table 3: Performance of our method on the 20 Newsgroups data set for 3 values of λ.

Experimental results

Benchmark problems Table 2 summarizes the comparison results for the five benchmark problems. (Here '$-$' means that the tolerance is not achieved in five hours.) The C implementation of our method is far superior to BBR and MOSEK for the problems. The Matlab implementation is a few times faster than IRLS for the smaller value of λ. For $\lambda = 10^{-1}\lambda_{\max}$, there was no significant difference in the performance of the Matlab implementation and IRLS. The comparison results clearly show that our method is more efficient than the existing methods for the benchmark problems. Moreover, its performance is not sensitive to the regularization parameter, unlike the existing methods.

A large sparse problem Our method could solve the ℓ_1-regularized LRP for the 20 Newsgroups data set efficiently. The runtime depends on the value of the regularization parameter. Table 3 summarizes the performance. No existing method could solve the problems in 10 hours.

Randomly generated problems To examine the effect of problem size on the performance of our method and the three existing methods (MOSEK, IRLS, and BBR), we used the 21 randomly generated data sets. with $\lambda = 0.1\lambda_{\max}$.

Figure 1 summarizes the scalability comparison results. Our method (IPM) is more efficient than the existing methods for small problems, and far more efficient for medium and large problems. By fitting an exponent to the data over the range from $n = 320$ to the largest problem successfully solved by each method, we find that the interior-point method scales almost linearly as $O(n^{1.3})$, *i.e.*, the runtime increases almost linearly with problem size. The empirical complexities of BBR, MOSEK and IRLS are $O(n^2)$, $O(n^3)$ and $O(n^{3.4})$ respectively.

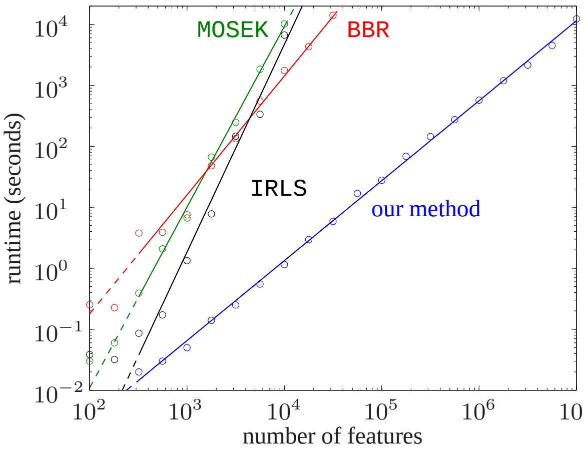

Figure 1: Runtime of our method and the three existing methods, for (standardized) randomly generated sparse problems, with the regularization parameter $\lambda = 0.1\lambda_{\max}$.

Comparison summary

In summary, the comparison results above show that our method is far more efficient than existing specialized methods for ℓ_1-regularized logistic regression, not only for small and medium problems but also for large problems, providing very accurate solutions reliably. For sparse problems, the empirical complexity of our method is almost linear in the problem size (*i.e.*, the number of features). The complexity of the existing methods is more than quadratic, and so the exiting methods become quickly inefficient as the problem size grows.

Conclusions

In this paper we have described a specialized method for solving ℓ_1-regularized LRP, which scales well to large problems that arise in practical applications. The method can be extended to other problems that involve ℓ_1 regularization, such as ℓ_1-regularized least squares problems. The most important part in the generalization is to find a preconditioner that gives a good trade-off between the computational complexity and the accelerated convergence it provides.

Acknowledgments

The authors thank Michael Grant, Trevor Hastie, Honglak Lee, Suin Lee, Mee Young Park, Yinyu Ye, and Sungroh Yoon for helpful comments and suggestions. This work was funded in part by the Focus Center Research Program Center for Circuit & System Solutions (www.c2s2.org), under contract 2003-CT-888, by AFOSR grant AF F49620-01-1-0365, by NSF grant ECS-0423905, by NSF grant 0529426, by DARPA/MIT grant 5710001848, by AFOSR grant FA9550-06-1-0514, DARPA/Lockheed contract N66001-06-C-2021, and by AFOSR/Vanderbilt grant FA9550-06-1-0312.

References

Alon, U.; Barkai, N.; Notterman, D.; Gish, K.; Ybarra, S.; Mack, D.; and Levine, A. 1999. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences* 96:6745–6750.

Balakrishnan, S., and Madigan, D. 2005. Algorithms for sparse linear classifiers in the massive data setting. Manuscript. Available from `www.stat.rutgers.edu/~madigan/papers/`.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Cawley, G., and Talbot, N. 2006. Gene selection in cancer classification using sparse logistic regression with bayesian regularization. *Bioinformatics* 22(19):2348–2355.

Demmel, J. 1997. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics.

Genkin, A.; Lewis, D.; and Madigan, D. 2006. Large-scale Bayesian logistic regression for text categorization. To appear in *Technometrics*. Available from `www.stat.rutgers.edu/~madigan/papers/`.

Gill, P.; Murray, W.; Saunders, M.; and Wright, M. 1986. User's guide for NPSOL (Version 4.0): A FORTRAN package for nonlinear programming. Technical Report SOL 86-2, Operations Research Dept., Stanford University, Stanford, California 94305.

Golub, T.; Slonim, D.; Tamayo, P.; Gaasenbeek, C.; Mesirov, J.; Coller, H.; Loh, M.; Downing, J.; Caligiuri, M.; Bloomfield, C.; and Lander, E. 1999. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* 286:531–537.

Goodman, J. 2004. Exponential priors for maximum entropy models. In *Proceedings of the Annual Meetings of the Association for Computational Linguistics*.

Keerthi, S., and DeCoste, D. 2005. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research* 6:341–361.

Koh, K.; Kim, S.-J.; and Boyd, S. 2006. An interior-point method for ℓ_1-regularized logistic regression. Manuscript. Available from `www.stanford.edu/~boyd/l1_logistic_reg.html`.

Krishnapuram, B.; Carin, L.; Figueiredo, M.; and Hartemink, A. 2005. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(6):957–968.

Lang, K. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*, 331–339.

Lee, S.; Lee, H.; Abeel, P.; and Ng, A. 2006. Efficient l_1-regularized logistic regression. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*.

Lokhorst, J. 1999. The LASSO and generalised linear models. Honors Project, Department of Statistics, The University of Adelaide, South Australia, Australia.

Madigan, D.; Genkin, A.; Lewis, D.; and Fradkin, D. 2005. Bayesian multinomial logistic regression for author identification.

McCallum, A. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. Available from `www.cs.cmu.edu/~mccallum/bow`.

MOSEK ApS. 2002. *The MOSEK Optimization Tools Version 2.5. User's Manual and Reference*. Available from `www.mosek.com`.

Newman, D.; Hettich, S.; Blake, C.; and Merz, C. 1998. UCI repository of machine learning databases. Available from `www.ics.uci.edu/~mlearn/MLRepository.html`.

Ng, A. 2004. Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*, 78–85. New York, NY, USA: ACM Press.

Osborne, M.; Presnell, B.; and Turlach, B. 2000. A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis* 20(3):389–403.

Park, M., and Hastie, T. 2006. An ℓ_1 regularization-path algorithm for generalized linear models. To appear in *Journal of the Royal Statistical Society, Series B*.

Perkins, S., and Theiler, J. 2003. Online feature selection using grafting. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*, 592–599. ACM Press.

Rosset, S. 2005. Tracking curved regularized optimization solution paths. In Saul, L.; Weiss, Y.; and Bottou, L., eds., *Advances in Neural Information Processing Systems 17*. Cambridge, MA: MIT Press.

Roth, V. 2004. The generalized LASSO. *IEEE Transactions on Neural Networks* 15(1):16–28.

Setakis, E.; Stirnadel, H.; and Balding, D. 2006. Logistic regression protects against population structure in genetic association studies. *Genome Research* 16:290–296.

Shevade, S., and Keerthi, S. 2003. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics* 19(17):2246–2253.

Shor, N. Z. 1985. *Minimization Methods for Non-differentiable Functions*. Springer Series in Computational Mathematics. Springer.

Wainwright, M.; Ravikumar, P.; and Lafferty, J. 2007. High-dimensional graphical model selection using ℓ_1-regularized logistic regression. To appear in *Advances in Neural Information Processing Systems (NIPS) 19*. Available from `http://www.eecs.berkeley.edu/~wainwrig/Pubs/publist.html#High-dimensional`.