# Sublinear Algorithms for Penalized Logistic Regression in Massive Datasets

Haoruo Peng[1,2], Zhengyu Wang[1,3], Edward Y. Chang[1], Shuchang Zhou[1], and Zhihua Zhang[1,4]

[1] Google Research Beijing, Beijing, China 100084
[2] Department of Computer Science and Technology
Tsinghua University, Beijing, China 100084
[3] Institute for Interdisciplinary Information Sciences
Tsinghua University, Beijing, China 100084
[4] College of Computer Science and Technology
Zhejiang University, Zhejiang, China 310027
penghaoruo@hotmail.com wangsincos@163.com eyuchang@gmail.com
georgezhou@google.com zhzhang@cs.zju.edu.cn

**Abstract.** Penalized logistic regression (PLR) is a widely used supervised learning model. In this paper, we consider its applications in large-scale data problems and resort to a stochastic primal-dual approach for solving PLR. In particular, we employ a random sampling technique in the primal step and a multiplicative weights method in the dual step. This technique leads to an optimization method with sublinear dependency on both the volume and dimensionality of training data. We develop concrete algorithms for PLR with $\ell_2$-norm and $\ell_1$-norm penalties, respectively. Experimental results over several large-scale and high-dimensional datasets demonstrate both efficiency and accuracy of our algorithms.

## 1 Introduction

The penalized logistic regression (PLR) model [9] plays an important role in machine learning and data mining. The model serves for classification problems, and enjoys a substantial body of supporting theories and algorithms. PLR is competitive with the support vector machines (SVMs) [18], because it has both high accuracy and interpretability (PLR can directly estimate a conditional class probability).

Recently, large-scale applications have emerged from many modern massive datasets. A key characteristic of these applications is that the size of their training data is very large and data dimensionality is very high. For example, in medical diagnostic applications [17], both doctors and patients would like to take the advantage of millions of records over hundreds of attributes. More evidently, search engines on texts or multimedia data must handle data volume in the billion scale and each data instance is characterized by a feature space of thousands of dimensions [7]. Large data volume and high data dimensionality pose computational challenges to machine learning problems.

In this paper, we tackle these challenges via stochastic approximation approaches. Stochastic approximation methods, such as stochastic gradient descent [20] and stochastic dual averaging [19], obtain optimal generalization guarantees with only a single pass or a small number of passes over the data. Therefore, they can achieve a desired generalization with runtime linear to the dataset size. We further speed up the runtime, and propose sublinear algorithms for PLR via the use of stochastic approximation idea. Our algorithms work at the same level of performance with traditional learning methods for PLR, but require much shorter running time. Our methods access a single feature of training vectors instead of entire training vectors at each iteration. This *sampling* approach brings much improved computational efficiency by eliminating a large number of vector multiplication operations. By devising clever randomized algorithms, we can also enjoy the benefits of taking less number of iterations and hence accessing less number of features. Such reduction in accessing features can substantially reduce running time as pointed out by [11].

Our algorithms can be easily applied to distributed storage systems [12] with parallel updates on all instances. Compared with other traditional batch algorithms, we do not require any global reduction [14] computation, which is a speedup bottleneck. Thus, our algorithms can achieve significant speedup on massive datasets.

The rest of the paper is organized as follows: Section 2 discusses some related work. In Section 3, we review some preliminaries and explain the setting along with the model. In Section 4, we present the framework of our sublinear algorithms for PLR. In Section 5, we depict detailed algorithms and analysis. Section 6 describes the datasets and the baseline of our experiments and presents the experimental results. Finally, we offer our concluding remarks in Section 7.

## 2 Related Work

There are many existing techniques that address logistic regression with $\ell_1$-penalty in the literature.

The *Reduced Memory Multi-pass* (RMMP) algorithm, proposed by Balakrishnan and Madigan [2], is one of the most accurate and fastest convergent algorithms. RMMP trains sparse linear classifiers on high-dimensional datasets in a multi-pass manner. However, this algorithm has computational complexity and memory requirements that make learning on large-scale datasets infeasible. The central idea of the work is a straightforward quadratic approximation to the likelihood function. When the dimensionality of the data gets large, the cost of many vector-vector multiplication operations increases significantly. Also, the quadratic approximation is added together for all instances in each iteration, and such computation inevitably requires global reduction in a distributed storage system.

The *Hybrid Iterative Shrinkage* (HIS) algorithm, proposed by Shi et al. [15], is also computationally efficient without loss of classification accuracy. This algorithm includes a fixed point continuation phase and an interior point phase. The

first phase is based completely on memory efficient operations such as matrix-vector multiplications, while the second phase is based on a truncated Newton's method. Thus, HIS is in the scope and constraints of traditional way of solving the optimization problem. As RMMP has relatively better scalability and performance, we choose to use RMMP instead of HIS as our baseline for the empirical comparison in this paper.

Recently, Clarkson et al. [3] proposed a new method by taking advantage of randomized algorithms. They presented sublinear-time approximation algorithms for optimization problems arising in machine learning, such as linear classifiers and minimum enclosing balls. The algorithm uses a combination of a novel sampling techniques and a new multiplicative update algorithm. They also proved lower bounds which show the running times to be nearly optimal on the unit-cost RAM model.

Hazan et al. [11] exploited sublinear approximation approach to the linear SVM with $\ell_2$-penalty, from which we were inspired and borrowed some of the ideas (We generally refer to them as the ETN framework in Section 4). Later on, Cotter et al. [4] extended the work to kernelized SVM cases. In [10], Hazan et al. applied the sublinear approximation approach for solving ridge ($\ell_2$-regularized) and lasso ($\ell_1$-regularized) linear regression. Garber and Hazan [6] developed the method in semidenfinite programming (SDP).

## 3 Penalized Logistic Regression Models

Logistic regression is a widely used method for solving classification problems. In this paper, we are mainly concerned with the binary classification problem. Suppose that we are given a set of training data $\mathcal{X} = \{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ are input samples and $y_i \in \{-1, 1\}$ are the corresponding labels. For simplicity, we let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$. In the logistic regression model, the expected value of $y_i$ is given by

$$P(y_i|\mathbf{x}_i) = \frac{1}{1 + \exp(-y_i(\mathbf{x}_i^T\mathbf{w} + b))} \triangleq g_i(y_i),$$

where $\mathbf{w} = (w_1, \ldots, w_d)^T \in \mathbb{R}^d$ is a regression vector and $b \in \mathbb{R}$ is an offset term. The log likelihood function $F(\mathbf{w}, b; \mathcal{X})$ on the training data is given as

$$F(\mathbf{w}, b|\mathcal{X}) = \sum_{i=1}^{n} \log g_i(y_i).$$

Under the penalized framework, one imposes a prior $p(\mathbf{w})$ to $\mathbf{w}$. This allows us to address the maximum a posteriori (MAP) estimation for $\mathbf{w}$ as

$$\max_{\mathbf{w},b} \big\{ \log\, p(\mathbf{w}, b|\mathcal{X}) \propto F(\mathbf{w}, b|\mathcal{X}) + \log\, p(\mathbf{w}) \big\}. \tag{1}$$

In this paper, we consider Gaussian and Laplace priors for $\mathbf{w}$, which in turn induce the $\ell_2$ and $\ell_1$ penalties for $\mathbf{w}$, respectively.

### 3.1 The $\ell_2$-Penalty Logistic Regression

We assume that $\mathbf{w}$ follows a Gaussian distribution with mean $\mathbf{0}$ and covariance matrix $\lambda\mathbf{I}_d$ where $\mathbf{I}_d$ is the $d\times d$ identity matrix, i.e. $\mathbf{w} \sim N(\mathbf{0}, \lambda\mathbf{I}_d)$. In this case, since

$$\log\, p(\mathbf{w}) = \frac{d}{2}\,\log\frac{\lambda}{2\pi} - \frac{\lambda}{2}\|\mathbf{w}\|_2^2,$$

we can equivalently formulate the optimization problem in (1) as

$$\max_{\mathbf{w},b}\ \Big\{F(\mathbf{w}, b|\mathcal{X}) - \frac{\lambda}{2}\|\mathbf{w}\|_2^2\Big\}. \tag{2}$$

(2) shows us that the problem reduces to an optimization problem with an $\ell_2$-penalty.

### 3.2 The $\ell_1$-Penalty Logistic Regression

In the second case, we impose a Laplace prior for $\mathbf{w}$, whose density is given by

$$\log\, p\,(\mathbf{w}) = d\,\log\frac{\gamma}{2} - \gamma\|\mathbf{w}\|_1.$$

With this prior, the optimization problem in (1) is equivalent to the following problem with the $\ell_1$-penalty.

$$\max_{\mathbf{w},b}\ \big\{F(\mathbf{w}, b|\mathcal{X}) - \gamma\|\mathbf{w}\|_1\big\}. \tag{3}$$

The advantage of $\ell_1$-penalty over $\ell_2$-penalty is its utility in sparsity modeling [16] . Thus, $\ell_1$-penalty logistic regression can serve for both classification and feature selection simultaneously.

## 4 Methodology

In this section, we first develop an approach to sublinear learning for $\ell_2$-penalty logistic regression. We then extend the approach to $\ell_1$-penalty case by adding certain conditions to achieve sparseness. Our approach is inspired by the *Elad-Tomer-Nathan* (ETN) framework in [11], a hybrid framework that deals with both hard margin and soft margin. Roughly speaking, our approach consists of three steps: deriving the hard margin and soft margin from the objective function, computing the derivative, and applying the ETN framework.

### 4.1 From $\ell_2$-Penalty to Soft Margin

We treat the objective function (2) as two parts: likelihood and penalty. With this in mind, we introduce the notion of hard margin and soft margin to respectively represent these two parts.

In particular, we consider an alternative optimization problem under an $\varepsilon$-suboptimal solution basis. That is,

$$\max_{\mathbf{w},b,\xi_i\geq 0} \min_{i\in\{1,\cdots,n\}} f_i(\mathbf{w},b)+\xi_i \ \text{ s.t. } \|\mathbf{w}\|_2\leq 1 \ \text{ and } \ \sum_{i=1}^{n}\xi_i\leq n\nu. \tag{4}$$

In (4) $f_i(\mathbf{w},b)=\log g_i(y_i)$ is the hard margin part, while $\xi_i$ is the soft margin part, and we have $\nu=-\frac{\sum_{i=1}^{n} f_i(\mathbf{w},b)}{n\|\mathbf{w}\|_2}$.

The following lemma shows the equivalent relationship between (2) and (4).

**Lemma 1.** *Let $(\mathbf{w}^\varepsilon,b^\varepsilon,\xi^\varepsilon)$ be an $\varepsilon$-suboptimal solution to the optimization problem (4) with optimal value $\kappa$, and consider the rescaled solution $\tilde{\mathbf{w}}=\mathbf{w}^\varepsilon/\kappa,\tilde{b}=b^\varepsilon/\kappa,\tilde{\xi}=\xi^\varepsilon/\kappa$. The the following two inequalities hold.*

$$\|\tilde{\mathbf{w}}\|_2\leq\frac{1}{1-\varepsilon\|\mathbf{w}\|_2}\|\mathbf{w}\|_2 \quad \textit{and} \quad F\left(\tilde{\mathbf{w}},\tilde{b}\right)\leq\frac{1}{1-\varepsilon\|\mathbf{w}\|_2}F(\mathbf{w},b).$$

The proof of Lemma 1 is given in Appendix A. Lemma 1 shows that solving (4) exactly yields Pareto optimal solutions of (2). Moreover, if we solve (4) via approximation, we obtain a suboptimal solution. As for parameters $\nu$ and $\xi_i$, we only need to consider $0\leq\nu\leq 1$ and $0\leq\xi_i\leq 2$.

### 4.2 Derivative of Objective Function

For hard margin, we compute the derivative of $f_i(\mathbf{w},b)$ with respect to $\mathbf{w}$. In this case, we have

$$f_i(\mathbf{w},b)=\log g_i(y_i). \tag{5}$$

The first partial derivative of (5) is as follows

$$\begin{aligned} coef &\triangleq \frac{\partial f_i(\mathbf{w},b)}{\partial\mathbf{w}}=-\frac{\partial\log[1+\exp(-y_i(\mathbf{w}^{\mathbf{T}}\mathbf{x}_i+b))]}{\partial\mathbf{w}} \\ &=\frac{\mathbf{x}_i y_i\exp(-y_i(\mathbf{w}^{\mathbf{T}}\mathbf{x}_i+b))}{1+\exp(-y_i(\mathbf{w}^{\mathbf{T}}\mathbf{x}_i+b))}=y_i g_i(-y_i)\mathbf{x}_i. \end{aligned} \tag{6}$$

In order to extend this result to $\ell_1$-penalty logistic regression, we only need to adjust the derivative of $f_i(\mathbf{w},b)$ with respect to $\mathbf{w}$. In this case, we need to use the sub-differential of $\|\mathbf{w}\|_1$. First, we define a signum multi-function of $t\in\mathbb{R}$ as

$$S(t)\triangleq\partial|t|=\begin{cases}\{+1\} & \text{if } t>0\\ [-1,1] & \text{if } t=0\\ \{-1\} & \text{if } t<0.\end{cases}$$

For $\mathbf{x}\in\mathbb{R}^d$, we define $S(\mathbf{x})\in\mathbb{R}^d$ with $(S(\mathbf{x}))_i=S(x_i)$ for $i=1,\ldots,d$. Then the derivative of (3) is

$$coef=y_i g_i(-y_i)\mathbf{x}_i-\gamma S(\mathbf{w}). \tag{7}$$

Eqn. (7) is the simple and general form for $coef$.

### 4.3 The ETN Framework

The *Elad-Tomer-Nathan* framework [11] is a hybrid method to handle hard margin and soft margin separately and simultaneously. The ETN framework enjoys the property of fast convergence for both hard margin and soft margin.

Each iteration of the method works in two steps. The first one is the *stochastic primal update*:

(1) An instance $i \in \{1, \ldots, n\}$ is chosen according to a probability vector $\mathbf{p}$;
(2) The primal variable $\mathbf{w}$ is updated according to the derivative of $f_i(\mathbf{w}, b)$ and the soft margin, via an online update with regret.

The second one is the *stochastic dual update*:

(1) A stochastic estimate of $f_i(\mathbf{w}, b)$ plus the soft margin is obtained, which can be computed in $O(1)$ time per term;
(2) The probability vector $\mathbf{p}$ is updated based on the above computed terms by using the *Multiplicative Updates* (MW) framework [1] for online optimization over the simplex.

## 5 Algorithms and Analysis

We use the following notations in our algorithms and analysis.

$clip(\cdot)$ is a projection function defined as follows:

$$clip(a, b) \triangleq \max(\min(a, b), -b) \quad a, b \in \mathbb{R}.$$

$\operatorname{sgn}(\cdot)$ is the sign function; namely,

$$\operatorname{sgn}(x) = \begin{cases} +1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0. \end{cases}$$

$g(\cdot)$ is the logistic function; namely,

$$g(x) = \frac{1}{1 + e^{-x}}$$

We let $\Lambda$ be the $\mathbb{R}_n$ Euclidean space which meets the following conditions:

$$\Lambda = \{\xi \in \mathbb{R}_n \,|\, \forall i,\, 0 \le \xi_i \le 2,\, \|\xi\|_1 \le \nu n\}.$$

**Algorithm 1** SLLR-L2

1: Input: $\varepsilon > 0, 0 \leq \nu \leq 1, X \in \mathbb{R}^{n \times d}, Y \in \mathbb{R}^n$
2: Let $T \leftarrow 1000^2 \varepsilon^{-2} \log n, \eta \leftarrow \sqrt{\log(n)/T}$
3: $\mathbf{u}_0 \leftarrow \mathbf{0}_d, \mathbf{w}_1 \leftarrow \mathbf{0}_d, \mathbf{q}_1 \leftarrow \mathbf{1}_n, b_1 \leftarrow 0$
4: **for** $t = 1$ to $T$ **do**
5: $\mathbf{p}_t \leftarrow \mathbf{q}_t / \|\mathbf{q}_t\|_1$
6: Choose $i_t \leftarrow i$ with probability $\mathbf{p}(i)$
7: Let $coef = y_{i_t} g\left(-y_{i_t}\left({\mathbf{w}_t}^T \mathbf{x}_{i_t} + b_t\right)\right)$
8: Let $\mathbf{u}_t \leftarrow \mathbf{u}_{t-1} + \frac{coef}{\sqrt{2T}} \mathbf{x}_{i_t}$
9: $\xi_t \leftarrow \operatorname{argmax}_{\xi \in \Lambda}\left({\mathbf{p}_t}^T \xi\right)$
10: $b_t \leftarrow \operatorname{sgn}\left({\mathbf{p}_t}^T \mathbf{y}\right)$
11: $\mathbf{w}_t \leftarrow \mathbf{u}_t / \max\{1, \|\mathbf{u}_t\|_2\}$
12: Choose $j_t \leftarrow j$ with probability $\mathbf{w}_t(j)^2 / {\|\mathbf{w}_t\|_2}^2$
13: **for** $i = 1$ to $n$ **do**
14: $\sigma \leftarrow \mathbf{x}_i(j_t) {\|\mathbf{w}_t\|_2}^2 / \mathbf{w}_t(j_t) + \xi_t(i) + y_i b_t$
15: $\hat{\sigma} \leftarrow clip(\sigma, 1/\eta)$
16: $\mathbf{q}_{t+1}(i) \leftarrow \mathbf{q}_t(i)\left(1 - \eta\hat{\sigma} + \eta^2 \hat{\sigma}^2\right)$
17: **end for**
18: **end for**
19: Output: $\bar{\mathbf{w}} = \frac{1}{T}\sum_t \mathbf{w}_t, \bar{b} = \frac{1}{T}\sum_t b_t$

### 5.1 The Sublinear Algorithm for $\ell_2$-Penalty Logistic Regression

We give the sublinear algorithm for $\ell_2$-penalty logistic regression in Algorithm 1. In the pseudo-code of Algorithm 1, line 5 to line 11 is the primal part, where $coef$ is the estimator of the derivatives and $\xi$ is the soft margin. Line 12 to line 17 is the dual part, where $\sigma$ serves as an estimator of $f_i(\mathbf{w}, b)$ plus the soft margin. $\sigma$ also serves as the derivative of $\mathbf{p}(i)$. Although the computation of line 15 and 16 makes $\hat{\sigma}$ a biased approximation, it is critical to the stability of the algorithm. The resulting bias is negligible in our approach. This can be shown in the experimental results in [11]. Because of the similarity between our SLLR-L2 and SVM-SIMBA presented in [11], we can naturally invoke the statement here.

Note that the update of $\xi_t$ in line 9 can be accomplished by using a simple greedy algorithm in $O(n)$ time. We can always set $\xi_t(i) = 2$ corresponding to the first $[\frac{\nu n}{2}]$ number of largest entries $\mathbf{p}(i)$ of $\mathbf{p}_t$ with respect to $i$. Then the residue $\nu n - 2[\frac{\nu n}{2}]$ is assigned to $\xi_t(\hat{i})$, where $\hat{i}$ is exactly the index of the $[\frac{\nu n}{2}]+1$ largest one in $\mathbf{p}_t$. Finally, we put $\xi_t(i) = 0$ elsewhere.

### 5.2 The Sublinear Algorithm for $\ell_1$-Penalty Logistic Regression

In Algorithm 2, we give the sublinear approximation procedure for $\ell_1$-penalty logistic regression. Here, we let $\mathbf{uprev}_t$ be $\mathbf{u}_{t-1}$ in the previous iteration. We achieve sparseness by adding pseudo-code from Line 11 to Line 16.

To make use of (7), we introduce some techniques to ensure the numerical convergence and stability. Considering the update computation in the primal step, we should make the following three rules.

**Algorithm 2** SLLR-L1

1: Input: $\varepsilon > 0, \gamma > 0, X \in \mathbb{R}^{n \times d}, Y \in \mathbb{R}^n$
2: Let $T \leftarrow 1000^2 \varepsilon^{-2} \log n, \eta \leftarrow \sqrt{\log(n)/T}$
3: $\mathbf{u}_0 \leftarrow \mathbf{0}_d, \mathbf{wavg}_0 \leftarrow \mathbf{0}_d, \mathbf{q}_1 \leftarrow \mathbf{1}_n, b_1 \leftarrow 0$
4: **for** $t = 1$ to $T$ **do**
5: $\mathbf{p}_t \leftarrow \mathbf{q}_t / \|\mathbf{q}_t\|_1$
6: $\mathbf{uprev}_t \leftarrow \mathbf{u}_{t-1}$
7: Choose $i_t \leftarrow i$ with probability $\mathbf{p}(i)$
8: Let $coef = y_{i_t} g\left(-y_{i_t}\left({\mathbf{wavg}_{t-1}}^T \mathbf{x}_{i_t} + b_t\right)\right)$
9: Let $\mathbf{u}_t \leftarrow \mathbf{u}_{t-1} + \frac{coef}{\sqrt{2T}} \mathbf{x}_{i_t}$
10: $b_t \leftarrow \text{sgn}\left({\mathbf{p}_t}^T \mathbf{y}\right)$
11: **for** $j = 1$ to $d$ **do**
12: **if** $\mathbf{uprev}_t(j) > 0$ **and** $\mathbf{u}_t(j) > 0$
13: $\mathbf{u}_t(j) = \max(\mathbf{u}_t(j) - \gamma, 0)$
14: **if** $\mathbf{uprev}_t(j) < 0$ **and** $\mathbf{u}_t(j) < 0$
15: $\mathbf{u}_t(j) = \min(\mathbf{u}_t(j) + \gamma, 0)$
16: **end for**
17: $\mathbf{w}_t \leftarrow \mathbf{u}_t / \max\{1, \|\mathbf{u}_t\|_2\}$
18: $\mathbf{wavg}_t \leftarrow \frac{t-1}{t} \mathbf{wavg}_{t-1} + \frac{1}{t} \mathbf{w}_t$
19: Choose $j_t \leftarrow j$ with probability $\mathbf{w}_t(j)^2 / {\|\mathbf{w}_t\|_2}^2$
20: **for** $i = 1$ to $n$ **do**
21: $\sigma \leftarrow \mathbf{x}_i(j_t) {\|\mathbf{w}_t\|_2}^2 / \mathbf{w}_t(j_t) + y_i b_t$
22: $\hat{\sigma} \leftarrow clip(\sigma, 1/\eta)$
23: $\mathbf{q}_{t+1}(i) \leftarrow \mathbf{q}_t(i)\left(1 - \eta\hat{\sigma} + \eta^2 \hat{\sigma}^2\right)$
24: **end for**
25: **end for**
26: Output: $\mathbf{wavg}_t, \bar{b} = \frac{1}{T} \sum_t b_t$

(1) When $\mathbf{u}_t(j) = 0$, we do not apply $-\gamma S(\mathbf{w})$ and simply make the value 0 by default.

(2) In order to apply $-\gamma S(\mathbf{w})$ for sparseness, we set $\mathbf{u}_t(j) = 0$, if it changes between positive values and negative values after applying the derivative. This is showed in Line 13 and Line 15.

(3) To avoid a $\mathbf{0}$ vector when $\gamma$ is large, we need to determine the derivative by a trend, not a single point. Thus, we consider two consecutive update steps of $\mathbf{u}_t(j)$. Line 12 and Line 14 ensure that if $\mathbf{u}_t(j)$ and $\mathbf{u}_{t-1}(j)$ are either both positive values or both negative ones, we apply the derivative, otherwise we do not change $\mathbf{u}_t(j)$. This is a logical approximation, and enables the small variance of values changing between positive values and negative ones.

For $\ell_1$-penalty logistic regression, the derivative is much more sensitive with respect to $\mathbf{w}_t$, as it is sparse in the computation. So in line 8, when we compute $coef$, we change $\mathbf{w}_t$ to $\mathbf{wavg}_{t-1}$ in order to make our algorithm more computationally stable.

### 5.3 Running Time Analysis

We now formally describe the MW algorithm and give theorems for running times of our algorithms.

**Definition 1.** *(MW algorithm) [3]. Consider a sequence of vectors* $\mathbf{v}_1, ..., \mathbf{v}_T \in \mathbb{R}^d$ *and a parameter* $\eta > 0$*. The Multiplicative Weights (MW) algorithm is defined as follows: let* $\mathbf{w}_1 \leftarrow \mathbf{1}_n$*, and for* $t \geq 1$*,*

$$\mathbf{p}_t \leftarrow \mathbf{w}_t / \|\mathbf{w}_t\|_1, \quad \textit{and} \quad \mathbf{w}_{t+1}(i) \leftarrow \mathbf{w}_t(i) \left(1 - \eta \mathbf{v}_t(i) + \eta^2 \mathbf{v}_t(i)^2\right).$$

The following lemma establishes a regret bound for the MW algorithm.

**Lemma 2.** *(The Variance MW Lemma) [3]. The MW algorithm satisfies*

$$\sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{v}_t \leq \min_{i \in \{1,\dots,n\}} \sum_{t=1}^{T} \max\{\mathbf{v}_t(i), -\frac{1}{\eta}\} + \frac{\log n}{\eta} + \eta \sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{v}_t^2$$

The following theorems give the running times of Algorithm 1 and Algorithm 2, respectively.

**Theorem 1.** *The SLLR-L2 algorithm returns an* $\varepsilon$*-approximate solution to the optimization problem of (4) with probability at least* $1/2$*. Its running time is* $\tilde{O}\left(\varepsilon^{-2}(n+d)\right)$*.*

We give the proof of Theorem 1 in Appendix B. Because the SLLR-L1 is essentially an extension of SLLR-L2, the running time is the same, and we omit the proof of Theorem 2 in this paper due to length constraint.

**Theorem 2.** *The SLLR-L1 algorithm returns an* $\varepsilon$*-approximate solution to the optimization problem of (3) with probability at least* $1/2$*. Its running time is* $\tilde{O}\left(\varepsilon^{-2}(n+d)\right)$

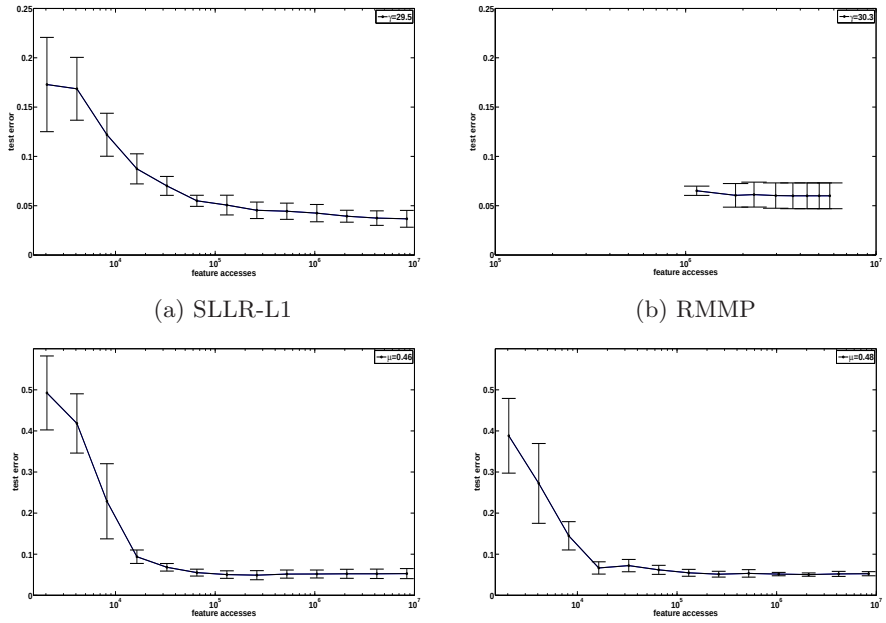
## 6 Experiments

In this section, we conduct an empirical analysis of our algorithms. Particularly, we illustrate test errors in terms of feature accesses and convergence in terms of MAP. As illustrated in Section 1, feature accesses are the main cost in computation. They are good indicators of running time and best demonstrate the efficiency of the proposed algorithms. For SLLR-L2, we choose SVM-SIMBA algorithm [11] as a comparison baseline. For SLLR-L1, we choose the state-of-the-art RMMP [2] algorithm, a popular method for solving logistic regression with $\ell_1$-penalty.

We choose three open datasets to run all four test programs: The **News-Group** dataset (after proper preprocessing) has 893 features and 1985 instances. We split it into a training set of 1390 instances and a test set of 595 instances. The second test dataset is the **Gisette** [8] dataset, which has 5000 features and

7000 instances. We split it into a training set of 6000 instances and a test set of 1000 instances. The third and final test dataset is the **ECUE Spam** [5] dataset, which has 197650 features and 10978 instances (after proper preprocessing). We split it into a training set of 9000 instances and a test set of 1978 instances. We randomly repeat such split 20 times and our analysis is based on the average performance of 20 repetitions.

### 6.1 Analysis of Performance



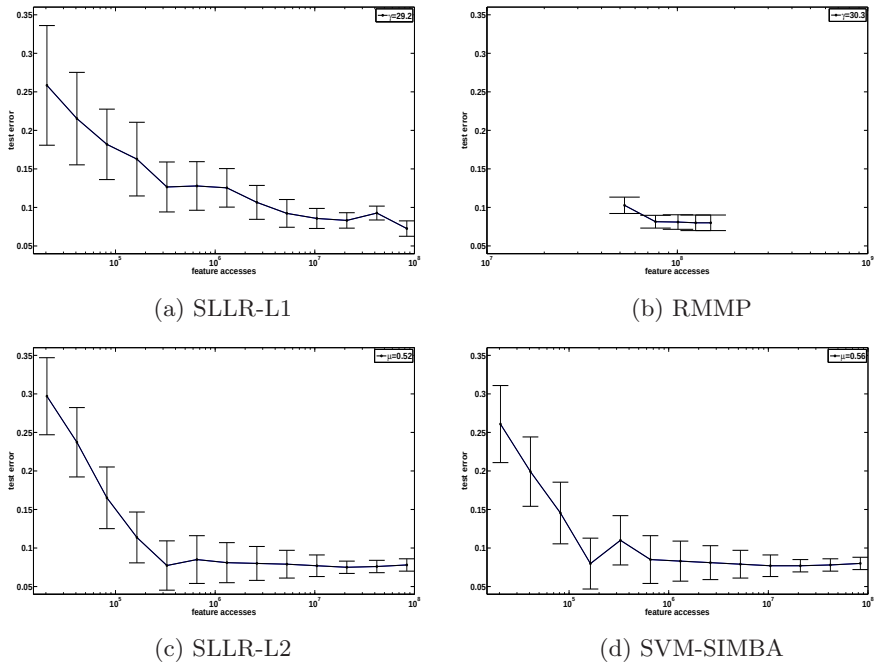

(a) SLLR-L1 (b) RMMP

(c) SLLR-L2 (d) SVM-SIMBA

**Fig. 1.** The test error, as a function of the number of feature accesses, on the **News-Group** dataset. For both SLLR-L1 and SLLR-L2, we set $\varepsilon = 0.5$.

In all three experiments, we tuned parameters $\nu$ and $\gamma$ of each algorithm based on the cross-validation method [13]. Note that our algorithms assume random access to features (as opposed to instances), thus it is not meaningful to compare the test error as a function of the number of iterations of each algorithm. Instead, according to our computational model, we compare the test error as a function of the number of feature accesses of each algorithm. The results, averaged over 20 repetitions, are presented in Figure 1, 2 and 3.

As can be seen from the figures, the performance of our SLLR-L2 algorithm is competitive with that of SIMBA on all the three datasets. With respect to $\ell_2$-penalty, our experiments show that our SLLR-L2 algorithm can achieve a similar




(a) SLLR-L1 (b) RMMP

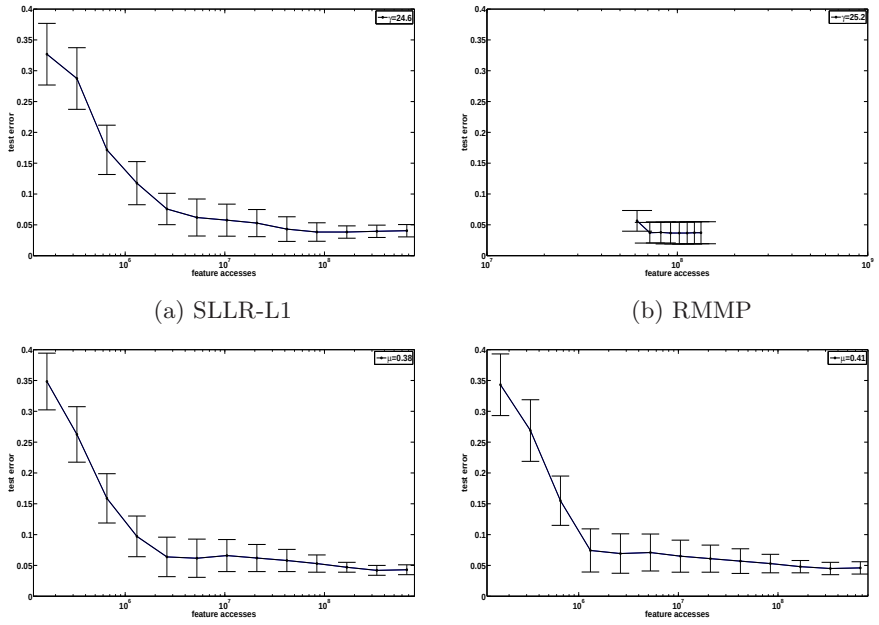
(c) SLLR-L2 (d) SVM-SIMBA

**Fig. 2.** The test error, as a function of the number of feature accesses, on the **Gisette** dataset. For both SLLR-L1 and SLLR-L2, we set $\varepsilon = 0.5$.

performance with SIMBA. With respect to $\ell_1$-penalty, our SLLR-L1 algorithm can achieve a same level of performance as RMMP. Our SLLR-L1 algorithm has a fast convergence rate, which enables us to achieve an acceptable test error with much fewer feature accesses in comparison with RMMP (basically a batch algorithm).

### 6.2 Analysis of Convergence

Figures 4 shows the convergence of our algorithms. With respect to $\ell_2$-penalty, we do not consider SVM-SIMBA, as its optimization objective function is not comparable with that of SLLR-L2. As the variance of MAP in different experiments is so small and does not contain much information, they are not shown in the figure for simplicity. The convergence of SLLR-L2 algorithm is very fast. There is a rapid growing of MAP, and it happens in a very early stage. With respect to $\ell_1$-penalty, the optimum value achieved by our SLLR-L1 and RMMP, a state-of-art algorithm with a remarkable accuracy on MAP, is very close. Our SLLR-L1, though not strictly better than RMMP on accuracy, has a very small gap away from the optimum solution and it is acceptable considering the test error results shown previously. Moreover, our SLLR-L1 has the advantage of achieving its local optimum value much earlier than RMMP. This is because our




(a) SLLR-L1 (b) RMMP

(c) SLLR-L2 (d) SVM-SIMBA

**Fig. 3.** The test error, as a function of the number of feature accesses, on the **ECUE Spam** dataset. For both SLLR-L1 and SLLR-L2, we set $\varepsilon = 0.5$.

approach loosely takes anywhere from 100 to 1000 times fewer feature accesses than RMMP.

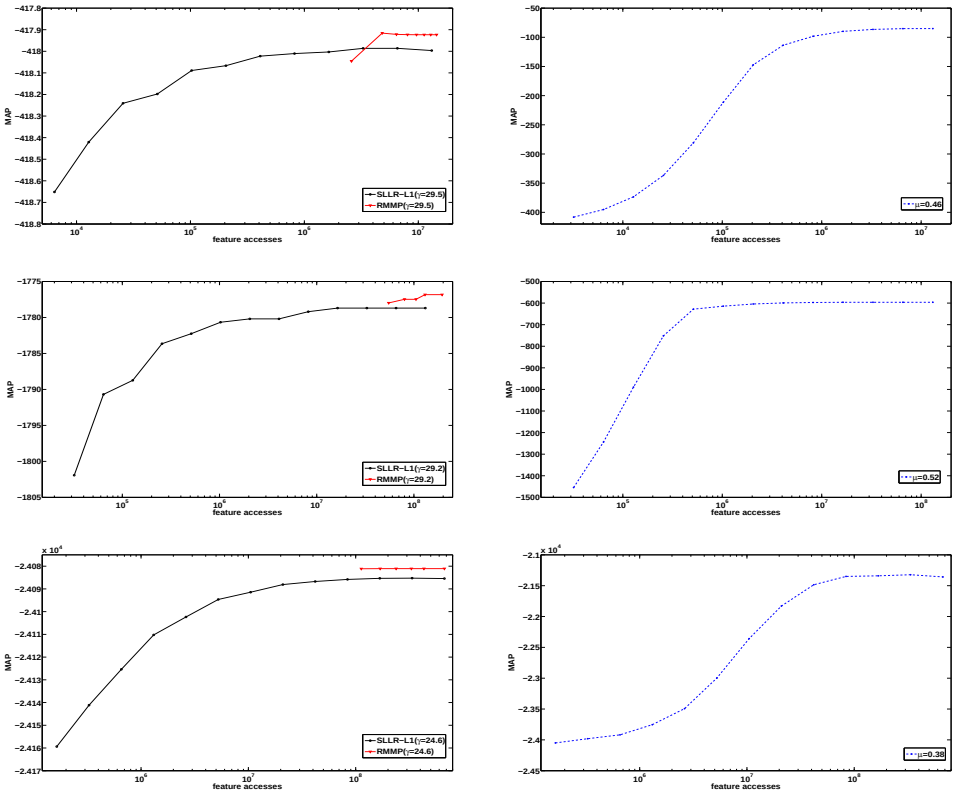
## 7 Conclusion

In this paper, we have presented two efficient algorithms to solve PLR through the use of a stochastic approximation approach. In particular, we have devised two sublinear algorithms for the logistic regression models with $\ell_2$-penalty and $\ell_1$-penalty, respectively. Experimental results have illustrated that our algorithms work well on massive datasets and have significant computational performance over other existing methods for PLR. Our algorithms can also be easily applied to distributed storage systems with parallel update on all instances.

## Appendix

### A. Proof of Lemma 1

The method we use here is similar to that in [11].




(a)SLLR-L1 and RMMP for $\ell_1$-penalty (b)SLLR-L2 for $\ell_2$-penalty

**Fig. 4.** The MAP, averaged over 20 random repetitions, as a function of the number of feature accesses, on the **NewsGroup**(first row), **Gisette**(second row), **ECUE Spam**(third row) datasets. For both SLLR-L1 and SLLR-L2, we set $\varepsilon = 0.5$.

*Proof.* We first consider the solution which is given by $\mathbf{w}^* = \mathbf{w}/\|\mathbf{w}\|_2, b^* = b/\|\mathbf{w}\|_2, \xi^* = \xi/\|\mathbf{w}\|_2$. So we have $\sum_{i=1}^n \xi_i^* = F(\mathbf{w}, b)/\|\mathbf{w}\|_2 = n\nu$. Then the optimal value is given by:

$$\kappa^* = \min_{i\in\{1,\cdots,n\}} \frac{f_i(\mathbf{w}, b) + \xi_i}{\|\mathbf{w}\|_2} = \frac{1}{\|\mathbf{w}\|_2}.$$

By the assumption on the suboptimality of $\mathbf{w}^\varepsilon, b^\varepsilon, \xi^\varepsilon$, we have $\kappa \geq \kappa^* - \varepsilon = \frac{1}{\|\mathbf{w}\|_2} - \varepsilon$, from which we can conclude that:

$$\|\tilde{\mathbf{w}}\|_2 = \frac{\|\mathbf{w}\|_2}{\kappa} \leq \frac{\|\mathbf{w}\|_2}{1/\|\mathbf{w}\|_2 - \varepsilon} \leq \frac{1}{1 - \varepsilon\|\mathbf{w}\|_2}\|\mathbf{w}\|_2.$$

From the form of the objective function, we also have:

$$F\left(\tilde{\mathbf{w}},\tilde{b}\right)\leq\sum_{i=1}^{n}\tilde{\xi}_i\leq\frac{n\nu}{\kappa}\leq\frac{F\left(\mathbf{w},b\right)}{\|\mathbf{w}\|_2}\cdot\frac{1}{1/\|\mathbf{w}\|_2-\varepsilon}=\frac{F\left(\mathbf{w},b\right)}{1-\varepsilon\|\mathbf{w}\|_2}.$$

## B. Proof of Theorem 1

The method we use here is similar to that in [11].
We first introduce some basic lemmas to simplify the proof.

**Lemma 3.** *For $\sqrt{\log(n)/T}\leq\eta\leq 1/6$ with probability at least $1-O(1/n)$, it holds that*

$$\max_{i\in\{1,\cdots,n\}}\sum_{t=1}^{T}\mathbf{v}_t(i)-\sum_{t=1}^{T}\left[\mathbf{x}_i^T\mathbf{w}_t+\xi_t(i)\right]\leq 4\eta T,$$

$$\left|\sum_{t=1}^{T}\mathbf{p}_t^T\mathbf{v}_t-\sum_{t=1}^{T}\mathbf{p}_t^T\left(\mathbf{X}\mathbf{w}_t+\xi_t\right)\right|\leq 4\eta T.$$

**Lemma 4.** *For $\sqrt{\log(n)/T}\leq\eta\leq 1/4$ with probability at least $1-O(1/n)$, it holds that*

$$\left|\sum_{t=1}^{T}\mathbf{x}_{i_t}^T\mathbf{w}_t-\sum_{t=1}^{T}\mathbf{p}_t^T\mathbf{X}\mathbf{w}_t\right|\leq 12\eta T\ \textit{and}\ \left|\sum_{t=1}^{T}\mathbf{x}_{i_t}^T\mathbf{w}^*-\sum_{t=1}^{T}\mathbf{p}_t^T\mathbf{X}\mathbf{w}^*\right|\leq 12\eta T.$$

**Lemma 5.** *With probability at least 3/4, it holds that*

$$\sum_{t=1}^{T}\mathbf{p}_t^T\mathbf{v}_t^2\leq 48T$$

We omit the proofs because they can be immediately obtained from [11] with some minor modifications.

*Proof.* Firstly, we prove the running time. Algorithm 1 makes $T=O(\varepsilon^{-2}\log n)$ iterations. Each iteration consists of two steps: the primal update and the dual update. The primal update contains a $\ell_1$-sampling process for the choice of $i_t$ ($O(n)$ time), and the update of $\mathbf{w}_t$ ($O(d)$ time). The update of $\xi_t$ can be done using a simple greedy algorithm which takes $O(n)$ time. The primal update contains a $\ell_2$-sampling process for the choice of $j_t$ ($O(d)$ time), and an update of $\mathbf{p}$ ($O(n)$ time). Altogether, each iteration takes $O(n+d)$ time and the overall running time is therefore $\tilde{O}\left(\varepsilon^{-2}\left(n+d\right)\right)$.

Next, we analyze the output quality of Algorithm 1. Let $\gamma^*$ be the value of the optimal solution of (4). Then, by the definition we have

$$\sum_{t=1}^{T}\mathbf{p}_t^T\left(\mathbf{X}\mathbf{w}^*+\xi^*\right)\geq T\gamma^*. \tag{8}$$

In the primal part of the algorithm we have

$$\sum_{t=1}^{T} \mathbf{x}_{i_t}^T \mathbf{w}_t \geq \sum_{t=1}^{T} \mathbf{x}_{i_t}^T \mathbf{w}^* - 2\sqrt{2T}.$$

Thus, from Lemma 4 we obtain that with probability $1 - O(1/n)$,

$$\sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{X}\mathbf{w}_t \geq \sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{X}\mathbf{w}^* - 2\sqrt{2T} - 24\eta T.$$

On the other hand,

$$\sum_{t=1}^{T} \mathbf{p}_t^T \xi_t \geq \sum_{t=1}^{T} \mathbf{p}_t^T \xi_t^*,$$

since $\xi_t$ is the maximizer of $\mathbf{p}_t^T \xi$, and recalling (8), we get the following lower bound:

$$\sum_{t=1}^{T} \mathbf{p}_t^T \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) \geq T\gamma^* - 2\sqrt{2T} - 24\eta T. \tag{9}$$

In the dual part of the algorithm, applying Lemma 2 on the clipped vector $\mathbf{v}_t$, we have that

$$\sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{v}_t \leq \min \sum_{t=1}^{T} \mathbf{v}_t + \frac{\log n}{\eta} + \eta \sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{v}_t^2,$$

and together with Lemma 3, we get that with probability $1 - O(1/n)$,

$$\sum_{t=1}^{T} \mathbf{p}_t^T \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) \leq \min \sum_{t=1}^{T} \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) + \frac{\log n}{\eta} + \eta \sum_{t=1}^{T} \mathbf{p}_t^T \mathbf{v}_t^2 + 8\eta T.$$

Hence, from Lemma 5, we obtain the following upper bound, with probability more than $\frac{3}{4} - O(1/n) \geq \frac{1}{2}$

$$\sum_{t=1}^{T} \mathbf{p}_t^T \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) \leq \min \sum_{t=1}^{T} \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) + \frac{\log n}{\eta} + 56\eta T. \tag{10}$$

Finally, combining bounds (9), (10) and dividing by $T$ we have that with probability more than $\frac{1}{2}$,

$$\min \frac{1}{T} \sum_{t=1}^{T} \mathbf{p}_t^T \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) \geq \gamma^* - \frac{2\sqrt{2}}{\sqrt{T}} - \frac{\log n}{\eta T} - 80\eta,$$

and using our choices for $T$ and $\eta$, we conclude that with probability at least $\frac{1}{2}$, it holds that

$$\min \left(\mathbf{X}\mathbf{w}_t + \xi_t\right) \geq \gamma^* - \varepsilon.$$

This implies that the vectors $(\bar{\mathbf{w}}, \bar{\xi})$ form an $\varepsilon$-approximate solution.

## References


1. S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta algorithm and applications. *Manuscript, 2005. Preliminary draft of paper available online at http://www.cs.princeton.edu/ arora/pubs/MWsurvey.pdf*, 2005.
2. S. Balakrishnan and D. Madigan. Algorithms for sparse linear classifiers in the massive data setting. *The Journal of Machine Learning Research*, 9:313–337, 2008.
3. K.L. Clarkson, E. Hazan, and D.P. Woodruff. Sublinear optimization for machine learning. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 449–457. IEEE Computer Society, 2010.
4. A. Cotter, S. Shalev-Shwartz, and N. Srebro. The kernelized stochastic batch perceptron. *Arxiv preprint arXiv:1204.0566*, 2012.
5. S. J. Delany, P. Cunningham, A. Tsymbal, and L. Coyle. A case-based technique for tracking concept drift in spam filtering. *Knowledge-Based Systems*, 18(4–5):187–195, 2005.
6. D. Garber and E. Hazan. Approximating semidefinite programs in sublinear time. In *Advances in Neural Information Processing Systems*, 2011.
7. A. Genkin, D.D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.
8. I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. *Advances in Neural Information Processing Systems*, 17:545–552, 2004.
9. T. Hastie, R. Tishirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, 2001.
10. E. Hazan and T. Koren. Optimal algorithms for ridge and lasso regression with partially observed attributes. *Arxiv preprint arXiv:1108.4559*, 2011.
11. E. Hazan, T. Koren, and N. Srebro. Beating sgd: Learning svms in sublinear time. In *Advances in Neural Information Processing Systems*, 2011.
12. C. Hogan, L. Cassell, J. Foglesong, J. Kordas, M. Nemanic, and G. Richmond. The livermore distributed storage system: Requirements and overview. In *Digest of Papers. Tenth IEEE Symposium on Mass Storage Systems*, pages 6–17. IEEE, 1990.
13. R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International joint Conference on artificial intelligence*, volume 14, pages 1137–1145. LAWRENCE ERLBAUM ASSOCIATES LTD, 1995.
14. D.K. Panda. Global reduction in wormhole k-ary n-cube networks with multidestination exchange worms. In *In IPPS: 9th International Parallel Processing Symposium*, pages 652–659. IEEE Computer Society Press, 1995.
15. J. Shi, W. Yin, S. Osher, and P. Sajda. A fast hybrid algorithm for large scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 1:8888, 2008.
16. R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
17. S. Tsumoto. Mining diagnostic rules from clinical databases using rough sets and medical diagnostic model. *Information sciences*, 162(2):65–80, 2004.
18. V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.
19. L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *The Journal of Machine Learning Research*, 11:2543–2596, 2010.
20. T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116. ACM, 2004.