Supplementary Materials for "A Comparison of Optimization Methods and Software for Large-scale L1-regularized Linear Classification"

Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh and Chih-Jen Lin
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan (cjlin@csie.ntu.edu.tw)

A Introduction

This document presents some materials not included in the paper. In particular, we provide more details of some methods and give additional experimental results.

B More Details about GLMNET

Let $\bar{w}_j = w_j^k + d_j$ and $\bar{L}_j(z) = \frac{1}{2}Az^2 + Bz$. GLMNET obtains the optimal solution $\bar{z}$ of $g_j(z)$ in (6.2) by the following closed-form solution:

$$\bar{w}_j + \bar{z} = \begin{cases} \frac{(|u|-1)\operatorname{sgn}(u)}{\bar{L}_j''(0)} & \text{if } |u| - 1 \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{I}$$

where $u = \bar{L}_j''(0)\bar{w}_j - \bar{L}_j'(0)$. We prove that (11) is equivalent to (27). From (27),

$$\begin{aligned} \bar{w}_j + \bar{z} &= \begin{cases} \bar{w}_j - \frac{\bar{L}_j'(0)+1}{\bar{L}_j''(0)} & \text{if } \bar{L}_j'(0) + 1 \leq \bar{L}_j''(0)\bar{w}_j, \\ \bar{w}_j - \frac{\bar{L}_j'(0)-1}{\bar{L}_j''(0)} & \text{if } \bar{L}_j'(0) - 1 \geq \bar{L}_j''(0)\bar{w}_j, \\ 0 & \text{otherwise,} \end{cases} \\ &= \begin{cases} \frac{\bar{L}_j''(0)\bar{w}_j - \bar{L}_j'(0) - 1}{\bar{L}_j''(0)} & \text{if } \bar{L}_j'(0) + 1 \leq \bar{L}_j''(0)\bar{w}_j, \\ \frac{\bar{L}_j''(0)\bar{w}_j - \bar{L}_j'(0) + 1}{\bar{L}_j''(0)} & \text{if } \bar{L}_j'(0) - 1 \geq \bar{L}_j''(0)\bar{w}_j, \\ 0 & \text{otherwise,} \end{cases} \\ &= \begin{cases} \frac{u-1}{\bar{L}_j''(0)} & \text{if } u - 1 \geq 0, \\ \frac{u+1}{\bar{L}_j''(0)} & \text{if } u + 1 \leq 0, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The last equality then leads to (I).

Next, we discuss the implementation of evaluating $\bar{L}_j(0)$ as it is the main operation at each inner iteration. As we mention in Section 6.2, GLMNET explicitly normalizes $\boldsymbol{x}_i$ by (6.2). Here we use $\bar{L}'_j(0;\tilde{X})$ to denote $\bar{L}'_j(0)$ on the scaled data. If we define

$$\tilde{y}_i = \begin{cases} 1 & \text{if } y_i = 1, \\ 0 & \text{if } y_i = -1, \end{cases}$$

then $\bar{L}'_j(0;\tilde{X})$ can be computed by

$$\begin{aligned}
\bar{L}'_j(0;\tilde{X}) &= Az + B \,|_{z=0}= B \\
&= \nabla_j L(\boldsymbol{w}^k;\tilde{X}) + \sum_{t=1}^{n} \nabla^2_{jt} L(\boldsymbol{w}^k;\tilde{X}) d_t \\
&= C \sum_{i=1}^{l} \big(\tau(y_i \boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - 1\big)\, y_i \tilde{x}_{ij} + C \sum_{t=1}^{n} (\tilde{X}^T D \tilde{X})_{jt} d_t \\
&= C \sum_{i=1}^{l} \left(\frac{x_{ij} - \bar{x}_j}{\sigma_j} \big(\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i\big) \right) + C \sum_{t=1}^{n} \sum_{i=1}^{l} \left(\frac{x_{ij} - \bar{x}_j}{\sigma_j} D_{ii} \frac{x_{it} - \bar{x}_t}{\sigma_t} d_t \right) \\
&= \frac{C}{\sigma_j} \sum_{i=1}^{l} x_{ij} \left(\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t} - D_{ii} \sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t} \right) - \\
&\quad\; C \frac{\bar{x}_j}{\sigma_j} \sum_{i=1}^{l} \left(\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t} - D_{ii} \sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t} \right).
\end{aligned}$$

To reduce the computational effort, GLMNET maintains a vector $\mathsf{r}[1 \ldots \mathsf{l}]$ and two scalars o and svr:

$$\begin{aligned}
\mathsf{r[i]} &\equiv \tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t}, \\
\mathsf{o} &\equiv \sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t}, \\
\mathsf{svr} &\equiv \sum_{i=1}^{l} \left(\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t} - D_{ii} \sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t} \right),
\end{aligned} \tag{II}$$

and evaluates $\bar{L}'_j(0;\tilde{X})$ by

$$\bar{L}'_j(0;\tilde{X}) = \frac{C}{\sigma_j} \left(\sum_{i=1}^{l} x_{ij} (\mathsf{r[i]} - D_{ii}\mathsf{o}) - \mathsf{svr}\bar{x}_j \right). \tag{III}$$

Notation in the paper	Variables in GLMNET code
l	no
n	ni
$D_{ii}, i = 1 \dots l$	v(1:no)
$\tau(\boldsymbol{w}^T \tilde{x}_i), i = 1 \dots l$	q(1:no)
$w_j, j = 1 \dots n$	b(1:ni)
$\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t}$	r(1:no)
$\sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t}$	o
$\sum_{i=1}^{l} \left(\tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i) - \tilde{y}_i + D_{ii} \sum_{t=1}^{n} \frac{x_{it} d_t}{\sigma_t} - D_{ii} \sum_{t=1}^{n} \frac{\bar{x}_t d_t}{\sigma_t} \right)$	svr
$\bar{L}'_j(0; \tilde{X})$	gk
$\bar{L}''_j(0; \tilde{X}), j = 1 \dots n$	xv(1:ni)
$\sum_{i=1}^{l} D_{ii} x_{ij}, j = 1 \dots n$	xm(1:ni)

To maintain $r[1 \dots l]$, o and svr, at each inner iteration, GLMNET updates

$$\begin{aligned} \mathsf{r[i]} &\leftarrow \mathsf{r[i]} + \frac{D_{ii} x_{ij}}{\sigma_j} \bar{z}, \forall i, \\ \mathsf{o} &\leftarrow \mathsf{o} + \frac{\bar{x}_j}{\sigma_j} \bar{z}, \\ \mathsf{svr} &\leftarrow \mathsf{svr} + \frac{\mathsf{xm[j]}}{\sigma_j} \bar{z}, \end{aligned}$$

where $\bar{z}$ is the optimal solution of $\min_z g_j(z)$ and

$$\mathsf{xm[j]} = \sum_{i=1}^{l} D_{ii} x_{ij}. \tag{IV}$$

At the end of each outer iteration, GLMNET computes $\tau(\boldsymbol{w}^T \tilde{x}_i), \forall i$, by (9), and

$$\mathsf{q[i]} \leftarrow \tau(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_i);$$

and then updates

$$\begin{aligned} D_{ii} &\leftarrow \mathsf{q[i]}(1 - \mathsf{q[i]}), \\ \mathsf{r[i]} &\leftarrow \mathsf{q[i]} - \tilde{y}_i, \\ \mathsf{svr} &\leftarrow \sum_{i=1}^{l} (\mathsf{q[i]} - \tilde{y}_i) \end{aligned}$$

and xm[j] by (IV).

In section 6.2, we discuss the shrinking technique applied in GLMNET. Algorithm 1 shows the details.

Algorithm 1 GLMNET with a shrinking technique

```
1. Given $\boldsymbol{w}^1$.
2. For $k = 1, 2, 3, \ldots$
    • Let $\boldsymbol{d}^k \leftarrow \boldsymbol{0}$.
    • While (TRUE)
        – Let $\Omega = \phi$.
        – for $j = 1, \ldots, n$
            ∗ Solve the following one-variable problem by (27):
                $$\bar{z} = \arg\min_{z} \quad q_k(\boldsymbol{d}^k + z\boldsymbol{e}_j) - q_k(\boldsymbol{d}^k).$$
            ∗ If $\bar{z} \neq 0$, then
                $$\Omega \leftarrow \Omega \cup \{j\}.$$
            ∗ $d^k_j \leftarrow d^k_j + \bar{z}$.
        – If $\boldsymbol{d}^k$ is optimal for minimizing $q_k(\boldsymbol{d})$, then BREAK.
        – While $\boldsymbol{d}^k$ is not optimal for minimizing $q_k(\boldsymbol{d})$ on $\Omega$
            ∗ for $j \in \Omega$
                · Solve the following one-variable problem by (27):
                    $$\bar{z} = \arg\min_{z} \quad q_k(\boldsymbol{d}^k + z\boldsymbol{e}_j) - q_k(\boldsymbol{d}^k).$$
                · $d^k_j \leftarrow d^k_j + \bar{z}$.
    • $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \boldsymbol{d}^k$.
```

C More Details on the Comparison Results

C.1 The Time of Each Solver to Reach a Specified Stopping Criterion

In Table A, we show the time of each solver to reduce the function value to be within 1% and 0.01% of the optimal value. When 0.01% is used, we have almost reached the final sparsity pattern. From the result, CDN is still the fastest for most data sets. To provide more information, in Table B we show the time of each solver to reduce the scaled 2-norm of the projected gradient (83) to be less than 0.01 and 0.0001. Note that Eq. (83) ≤ 0.0001 is a very strict condition, which most of the solvers in our experiments can not meet. In such a strict condition, IPM outperforms CDN.

C.2 Number of Iterations

In Table C, we show the number of iterations of each solver to reduce the norm of the scaled project gradient norm (83) to be less than 0.01.

C.3 Average Cost per Iteration

In Table D, we show the average cost per iteration of CDN, CDN-NS, TRON and BBR. We run the programs until the scaled norm of the projected gradient (83) is below 0.01.

D Comparison of L1 and L2 Regularized Logistic Regression

We show the comparison of L1 and L2 regularized logistic regression in Figure A. We use CDprimal to indicate the primal coordinate descent method in Chang et al. (2008) and CDdual for the dual coordinate descent method in Yu et al. (2011). For document data used here, our results indicate that L1's accuracy is only similar to L2's (or even slightly lower). However, for some other types of data (e.g., KDD Cup 2009), we find that L1 is better. Therefore, whether L1 or L2 gives better accuracy deserves further investigations.

Training time is another issue. From Figure A, L2-regularized logistic regression by CDdual is faster than L1-regularized logistic regression by CDN. For practitioners, we recommend them to try L2 first for classification and L1 for feature selection; see also the description in our LIBLINEAR FAQ.[*]

E Convergence Rate of CDN: Logistic Regression

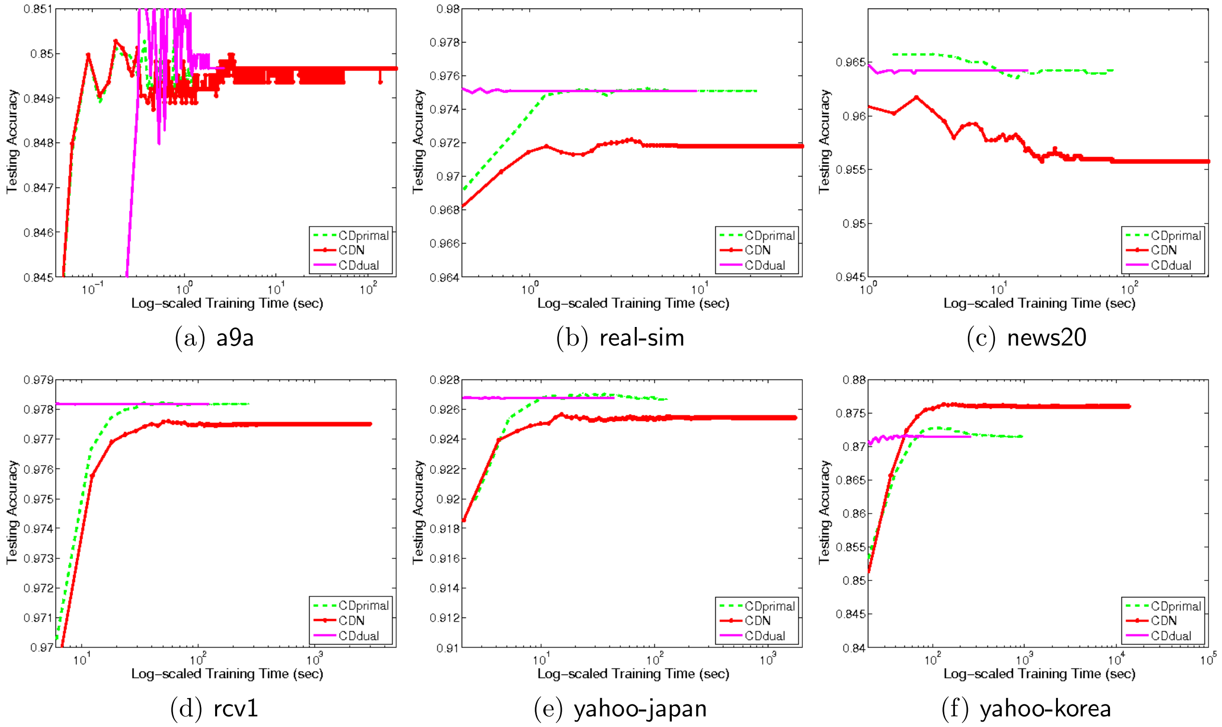
In this section, we show that CDN converges at least linearly if the logistic loss function $L(\boldsymbol{w})$ is strictly convex. We can check whether X has full column rank or not beforehand to ensure the strict convexity. In the following proof, shrinking is not considered.

To apply the linear convergence result in Tseng and Yun (2009), we show that L1-regularized logistic regression satisfies the conditions in their Theorem 2(b) if the loss term $L(\boldsymbol{w})$ is strictly convex (and therefore $\boldsymbol{w}^*$ is unique).

From Appendix D of the paper, we know L1-regularized logistic regression has the following properties.

1. $\nabla L(\boldsymbol{w})$ is Lipschitz continuous; see (92) of the paper.
2. The level set is compact, and hence the optimal solution $\boldsymbol{w}^*$ exists.
3. $\lambda_{\min} \preceq \nabla^2_{jj} L(\boldsymbol{w}^{k,j}) \preceq \lambda_{\max}$, $\forall j = 1, \ldots, n, \forall k$; see (95).

[*] http://www.csie.ntu.edu.tw/~cjlin/liblinear/FAQ.html



(a) a9a

(b) real-sim

(c) news20

(d) rcv1

(e) yahoo-japan

(f) yahoo-korea

Figure A: The testing accuracy versus the training time (log-scaled) A comparison of L1 and L2 regularized Logistic regression without using the bias term.

In addition to the above three conditions, Tseng and Yun (2009, Theorem 2(b)) require that for all $\zeta \geq \min_{\boldsymbol{w}} f(\boldsymbol{w})$, there exists $T > 0, \epsilon > 0$, such that

$$T\|\boldsymbol{d}_{\mathcal{I}}(\boldsymbol{w})\| \geq \|\boldsymbol{w} - \boldsymbol{w}^*\|, \quad \forall \boldsymbol{w} \in \{\boldsymbol{w} \mid f(\boldsymbol{w}) \leq \zeta \text{ and } \|\boldsymbol{d}_{\mathcal{I}}(\boldsymbol{w})\| \leq \epsilon\}, \tag{V}$$

where $\boldsymbol{d}_{\mathcal{I}}(\boldsymbol{w})$ is the solution of CGD-GS's sub-problem (Equation (38) of the paper) at $\boldsymbol{w}$ with $H = \mathcal{I}$ (Tseng and Yun, 2009, Assumption 2). For logistic loss, Yuan et al. (2011, Appendix B) has shown that (V) is satisfied if the loss function is strictly convex.

Therefore, all conditions in Tseng and Yun (2009, Theorem 2(b)) are satisfied, so linear convergence is guaranteed.

F Convergence Rate of CDN: L2-loss SVM

We prove that CDN converges at least linearly if L2-loss function $L(\boldsymbol{w})$ is strictly convex. Similar to the discussion in Section E, we show that all conditions in Tseng and Yun (2009, Theorem 2(b)) are satisfied.

We only need to check condition (V) because Appendix F of the paper has checked all the other conditions. The proof of condition (V) in Yuan et al. (2011, Appendix B) is independent of the loss function and only requires the loss function $L(\boldsymbol{w})$ to be strictly convex, $\nabla L(\boldsymbol{w})$ to be Lipschitz continuous, and the level set to be compact. As a result, the proof in Yuan et al. (2011, AppendixB) can be applied to L2-loss SVM directly. Therefore, linear convergence is guaranteed.

References

Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. Coordinate descent method for large-scale L2-loss linear SVM. *Journal of Machine Learning Research*, 9:1369–1398, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/cdl2.pdf`.

Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117:387–423, 2009.

Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, October 2011. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/maxent_dual.pdf`.

Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin. An improved GLMNET for l1-regularized logistic regression and support vector machines. Technical report, National Taiwan University, 2011. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/long-glmnet.pdf`.

Logistic regression w/o bias $(f - f^*)/f^* \leq 0.01$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
BBR	0.61	1.12	0.14	2.80	17.01	69.72	11.71	157.80
CDN-NS	**0.05**	**0.13**	**0.08**	**1.50**	**8.72**	**34.61**	**8.08**	**78.60**
SCD	11.68	*	5.47	*	*	*	*	*
TRON	0.38	1.00	0.21	4.78	147.15	1419.92	503.75	33656.71
BMRM	3.22	20.32	*	*	*	*	*	*
OWL-QN	1.41	2.65	0.29	2.23	55.26	62.18	36.33	951.20

Logistic regression w/o bias $(f - f^*)/f^* \leq 0.0001$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
BBR	3.08	11.90	*	10.17	261.43	1605.06	503.56	16647.06
CDN-NS	**0.35**	1.14	52.78	**3.66**	**34.45**	**118.21**	**64.44**	**250.99**
SCD	*	*	648.26	*	*	*	*	*
TRON	0.50	**1.13**	**7.37**	12.15	195.97	2851.32	1405.42	147227.69
BMRM	14.11	57.86	*	*	*	*	*	*
OWL-QN	4.01	9.24	11.41	5.85	193.60	168.79	105.81	4305.01

Logistic regression w/ bias $(f - f^*)/f^* \leq 0.01$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	0.27	0.05	**0.08**	**1.55**	**10.02**	**82.58**	**36.52**	**92.97**
CGD-GS	2.48	1.02	0.36	7.73	572.71	485.90	119.49	1833.33
IPM	0.46	0.51	0.18	5.93	169.55	204.54	81.08	1014.91
GLMNET	**0.03**	**0.04**	27.21	*	12302.44	34261.79	*	*
Lassplore	0.29	0.28	0.91	41.71	445.76	1329.36	456.69	7789.72

Logistic regression w/ bias $(f - f^*)/f^* \leq 0.0001$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	0.48	0.58	53.72	**6.94**	**40.79**	524.35	**117.93**	**1432.64**
CGD-GS	2.95	1.94	*	16.93	1433.38	1211.45	*	8444.67
IPM	0.62	0.76	**0.75**	10.35	247.69	**296.23**	218.26	3329.53
GLMNET	**0.04**	**0.06**	34.90	*	*	*	*	*
Lassplore	0.40	0.40	7.54	133.93	1612.33	4444.23	1502.42	28098.40

L2-loss SVM $(f - f^*)/f^* \leq 0.01$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	**1.30**	**1.02**	**0.02**	**0.24**	**48.81**	**6.32**	**3.30**	**14.61**
TRON	6.04	10.00	0.15	3.06	2911.90	124.14	589.02	12651.89
BMRM	34.17	*	*	*	*	*	*	*

L2-loss SVM $(f - f^*)/f^* \leq 0.0001$

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	21.78	*	9.78	**0.88**	**331.13**	**40.51**	**24.62**	**64.84**
TRON	**7.92**	**13.10**	**9.32**	8.83	6331.49	346.89	1854.07	37424.07
BMRM	*	*	*	*	*	*	*	*

Table A: Time in seconds to reduce the relative difference to the optimal function value to be less than 0.01 and 0.0001. We boldface the best approach. The asterisk symbol (*) indicates that a solver is unable to reach the target in a reasonable amount of time.

Logistic regression w/o bias, (83) ≤ 0.01

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
BBR	1.57	0.41	3.38	7.35	132.08	111.74	1027.90	*
CDN-NS	**0.13**	**0.05**	1.46	**2.24**	**9.40**	**39.94**	**94.83**	**378.59**
SCD	27.48	31.05	55.52	*	*	*	*	*
TRON	0.43	0.89	**0.65**	7.46	158.14	621.00	1460.28	123535.11
BMRM	11.36	14.24	*	*	*	*	*	*
OWL-QN	2.45	2.03	3.84	3.81	68.03	79.72	141.51	6725.46

Logistic regression w/o bias, (83) ≤ 0.0001

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
BBR	7.06	12.04	*	29.80	1117.76	5577.84	3563.41	*
CDN-NS	0.95	**0.75**	226.78	**11.67**	**78.11**	336.09	**799.53**	**5600.84**
SCD	*	*	*	*	*	*	*	*
TRON	**0.51**	1.13	**13.10**	24.27	243.74	3217.27	1820.74	164513.18
BMRM	*	78.91	*	*	*	*	*	*
OWL-QN	*	10.16	*	15.21	400.71	**308.78**	*	*

Logistic regression w/ bias, (83) ≤ 0.01

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	0.45	0.40	1.62	5.50	**32.12**	285.85	185.13	2902.26
CGD-GS	2.95	1.02	6.59	15.21	3730.23	580.16	*	*
IPM	0.04	0.08	**0.31**	**5.11**	136.23	**101.42**	**84.74**	**947.67**
GLMNET	**0.04**	**0.06**	34.90	*	16309.71	*	*	*
Lassplore	0.34	0.37	3.17	87.34	726.63	1998.81	1574.16	32368.17

Logistic regression w/ bias, (83) ≤ 0.0001

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	0.77	1.61	126.05	18.55	**105.81**	1188.64	882.12	6524.33
CGD-GS	3.80	3.84	*	55.11	*	*	*	*
IPM	0.04	0.08	**1.06**	**10.67**	205.79	**250.01**	**214.59**	**2992.82**
GLMNET	**0.04**	**0.08**	*	*	*	*	*	*
Lassplore	*	1.04	*	*	*	*	*	*

L2-loss SVM, (83) ≤ 0.01

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	**0.19**	**0.30**	**0.29**	**0.41**	**17.92**	**8.40**	**35.97**	**93.46**
TRON	4.12	5.75	2.92	5.65	666.21	102.44	1886.07	36186.20
BMRM	46.79	*	*	*	*	*	*	*

L2-loss SVM, (83) ≤ 0.0001

	leu	duke	a9a	real-sim	news20	rcv1	yahoo-japan	yahoo-korea
CDN-NS	8.67	**5.72**	26.82	**3.40**	**467.84**	**91.22**	**192.05**	**904.64**
TRON	**8.07**	12.42	**13.12**	10.79	6495.44	534.13	2034.20	49897.53
BMRM	*	*	*	*	*	*	*	*

Table B: Time in seconds to reduce the scaled norm of the projected gradient (83) to be less than 0.01 and 0.0001. We boldface the best approach. The asterisk symbol (*) indicates that a solver is unable to reach the target in a reasonable amount of time.

	Number of iteration			#nHv
Data set	CDN-NS	BBR	TRON	TRON
leu	34	59	41	373
duke	8	11	60	581
a9a	62	81	16	175
real-sim	8	20	23	250
news20	13	147	181	2179
rcv1	7	15	63	739
yahoo-japan	55	440	684	8732
yahoo-korea	26	*	4883	77797

Table C: The number of iterations of each solver and the number of Hessian-vector products (#nHv) of TRON. We run the programs until the scaled norm of the projected gradient is less than 0.01.

Data set	CDN	CDN-NS	TRON	BBR
leu	0.00	0.00	0.01	0.03
duke	0.00	0.01	0.01	0.04
a9a	0.02	0.02	0.04	0.04
real-sim	0.25	0.28	0.32	0.37
news20	0.41	0.72	0.87	0.90
rcv1	5.61	5.71	9.86	7.45
yahoo-japan	1.31	1.72	2.13	2.34
yahoo-korea	12.43	14.56	25.30	*

Table D: The average cost (in seconds) per iteration.