Supplement of "Coresets for Wasserstein Distributionally Robust Optimization Problems"

Anonymous Author(s)
Affiliation
Address
`email`

1 Omitted Proofs

1.1 Proof of Theorem 1

Proof. Suppose $W = [w_1, \dots, w_n]$ satisfies Definition 2. By Lemma 1, we have $\lambda_*(\beta), \tilde{\lambda}_*(\beta) \in [\kappa(\beta), \tau(\beta)]$, which implies that

$$\tilde{H}(\beta, \lambda_*(\beta)) \in (1 \pm \epsilon) H(\beta, \lambda_*(\beta)). \tag{1}$$

Then for any fixed $\beta \in \mathcal{H}$, we have both

$$\lambda_* \theta^p + H(\beta, \lambda_*) \geq \frac{1}{1+\epsilon}(\lambda_* \theta^p + \tilde{H}(\beta, \lambda_*)) \geq \frac{1}{1+\epsilon}(\tilde{\lambda}_* \theta^p + \tilde{H}(\beta, \tilde{\lambda}_*)) \tag{2}$$

and

$$\lambda_* \theta^p + H(\beta, \lambda_*) \leq \tilde{\lambda}_* \theta^p + H(\beta, \tilde{\lambda}_*) \leq \frac{1}{1-\epsilon}(\tilde{\lambda}_* \theta^p + \tilde{H}(\beta, \tilde{\lambda}_*)). \tag{3}$$

From proposition 1 we have $R_{\theta,p}(\beta) = H(\beta, \lambda_*) + \lambda_* \theta^p$ and $\tilde{R}_{\theta,p}(\beta) = \tilde{H}(\beta, \tilde{\lambda}_*) + \tilde{\lambda}_* \theta^p$. Together with (2) and (3), they imply

$$\tilde{R}_{\theta,p}(\beta) \in (1 \pm \epsilon) R_{\theta,p}(\beta). \tag{4}$$

Therefore, W is a qualified coreset satisfying Definition 1. □

1.2 Proof of Claim 1

Proof. Denote $z_1 = \arg\max_{z \in \Xi}\{\ell(\beta_1, z) - \lambda \mathtt{d}^p(z, \xi)\}$ and $z_2 = \arg\max_{z \in \Xi}\{\ell(\beta_2, z) - \lambda \mathtt{d}^p(z, \xi)\}$. Then $h(\beta, \lambda_1, \xi) = \ell(\beta_1, z_1) - \lambda \mathtt{d}^p(z_1, \xi)$ and $h(\beta, \lambda_2, \xi) = \ell(\beta_2, z_2) - \lambda \mathtt{d}^p(z_2, \xi)$.

By the definitions of z_1 and z_2, we have

$$\left.\begin{array}{l} \ell(\beta_1, z_1) - \lambda \mathtt{d}^p(z_1, \xi) \geq \ell(\beta_1, z_2) - \lambda \mathtt{d}^p(z_2, \xi); \\ \ell(\beta_2, z_1) - \lambda \mathtt{d}^p(z_1, \xi) \leq \ell(\beta_2, z_2) - \lambda \mathtt{d}^p(z_2, \xi). \end{array}\right\} \tag{5}$$

By Assumption 1 ii, we have

$$\left.\begin{array}{l} |\ell(\beta_1, z_1) - \ell(\beta_2, z_1)| \leq L\|\beta_1 - \beta_2\|_2; \\ |\ell(\beta_1, z_2) - \ell(\beta_2, z_2)| \leq L\|\beta_1 - \beta_2\|_2. \end{array}\right\} \tag{6}$$

Combining (6) and (5), we have

$$\left.\begin{array}{l} h(\beta, \lambda_1, \xi) - L\|\beta_1 - \beta_2\|_2 \leq h(\beta, \lambda_2, \xi); \\ h(\beta, \lambda_2, \xi) - L\|\beta_1 - \beta_2\|_2 \leq h(\beta, \lambda_1, \xi). \end{array}\right\} \tag{7}$$

That is ,

$$|h(\beta, \lambda_1, \xi) - h(\beta, \lambda_2, \xi)| \leq L\|\beta_1 - \beta_2\|_2$$

□

1.3 Proof of Claim 2

Proof. From [4, lemma 3 (ii)], we know that $h(\beta, \lambda, \xi)$ is convex and non-increasing in λ. Define

$$\underline{D}(\beta, \lambda, \xi) := \liminf_{\delta \downarrow 0} \left\{ \mathtt{d}(\xi, \zeta) : \ell(\beta, \zeta) - \lambda \mathtt{d}^p(\xi, \zeta) \geq h(\beta, \lambda, \xi) - \delta \right\}.$$

Further, Gao and Kleywegt [4, lemma 3 (iv)] showed that $\underline{D}^p(\beta, \lambda, \xi)$ is a subderivative on λ for $h(\beta, \lambda, \xi)$. Therefore, from the convexity of $h_i(\beta, \cdot)$, we know

$$|h_i(\beta, \lambda) - h_i(\beta, \lambda')| \leq \max\{\underline{D}^p(\beta, \lambda, \xi_i), \underline{D}^p(\beta, \lambda', \xi_i)\}|\lambda - \lambda'|, \forall \lambda, \lambda' \geq \kappa(\beta). \tag{8}$$

From Assumption 2 (i), we know that $\ell(\beta, \zeta) - \lambda \mathtt{d}^p(\xi, \zeta)$ is continuous in ζ.

If $\{\zeta \in \Xi \colon \ell(\beta, \zeta) - \lambda \mathtt{d}^p(\zeta, \xi_i) = h_i(\beta, \lambda)\} = \emptyset$, by the continuity of $\ell(\beta, \cdot) - \lambda \mathtt{d}^p(\xi, \cdot)$ and the mathematical analysis, we have $\underline{D}(\beta, \lambda, \xi_i) = \infty$. We set $r_i(\beta, \lambda) = \infty$ in this case.

If the set $\{\zeta \in \Xi \colon \ell(\beta, \zeta) - \lambda \mathtt{d}^p(\zeta, \xi_i) = h_i(\beta, \lambda)\}$ is non-empty, by the definition of $r_i(\beta, \lambda)$ and $\underline{D}(\beta, \lambda, \xi_i)$, we know that $r_i(\beta, \lambda) \geq \underline{D}(\beta, \lambda, \xi_i)$, which completes the proof.

□

1.4 Proof of Lemma 2

Proof. For any fixed $0 \leq i, j \leq N$, we regard $h_k(\beta, \lambda)$ as an independent random variable for each $\xi_k \in Q_{ij}$. We consider the following two cases: (i) $\kappa(\beta_{\mathtt{anc}}) \leq \kappa(\beta)$ and (ii) $\kappa(\beta_{\mathtt{anc}}) > \kappa(\beta)$.

For case (i), we have $\lambda \geq \kappa(\beta) \geq \kappa(\beta_{\mathtt{anc}})$ and $\lambda_{\mathtt{anc}} \geq \kappa(\beta_{\mathtt{anc}})$. Together with Claim 2 and (20), we have

$$\mu_i \cdot 2^{i-1} A - R l_{\mathtt{d}} \leq h_k(\beta_{\mathtt{anc}}, \lambda) \leq 2^j B + R l_{\mathtt{d}}. \tag{9}$$

By using Claim 1, we further obtain the following upper and lower bounds for $h_k(\beta, \lambda)$:

$$\left.\begin{aligned} h_k(\beta, \lambda) &\leq \quad 2^j B + R l_{\mathtt{d}} + L l_{\mathtt{p}}; \\ h_k(\beta, \lambda) &\geq \mu_i 2^{i-1} A - R l_{\mathtt{d}} - L l_{\mathtt{p}}. \end{aligned}\right\} \tag{10}$$

For case (ii), we have $\lambda_{\mathtt{anc}} \geq \kappa(\beta) > \kappa(\beta)$ and $\lambda \geq \kappa(\beta)$. We apply Claim 1 and have

$$\mu_i \cdot 2^{i-1} A - L l_{\mathtt{p}} \leq h_k(\beta_{\mathtt{anc}}, \lambda) \leq 2^j B + L l_{\mathtt{p}}. \tag{11}$$

Together with Claim 2 we can achieve the same lower and upper bounds as (10).

Let the sample size $|Q_{ij}| = O((2^j B - \mu_i 2^{i-1} A + 2 L l_{\mathtt{p}} + 2 R l_{\mathtt{d}})^2 \sigma^{-2} \log \frac{1}{\eta})$. Through the Hoeffding's inequality [5], we have

$$\mathbb{P}\left[\left|\frac{1}{|Q_{ij}|} \sum_{\xi_k \in Q_{ij}} h_k(\beta, \lambda) - \frac{1}{|C_{ij}|} \sum_{\xi_k \in C_{ij}} h_k(\beta, \lambda)\right| \geq \sigma\right] \leq \eta.$$

□

1.5 Proof of Lemma 3

Proof. Based on Lemma 2, we have

$$\left|\frac{|C_{ij}|}{|Q_{ij}|} \sum_{\xi_k \in Q_{ij}} h_k(\beta, \lambda) - \sum_{\xi_k \in C_{ij}} h_k(\beta, \lambda)\right| \leq |C_{ij}| \cdot \epsilon_1 (2^{j-1} + 2^{i-1}) A \tag{12}$$

with probability at least $1 - \eta$. Through taking a union bound over all the cells, with probability at least $1 - (N+1)^2 \eta$, we have

$$\begin{aligned} n|\tilde{H}(\beta, \lambda) - H(\beta, \lambda)| &= |\sum_{i,j} \sum_{\xi_k \in Q_{ij}} \frac{|C_{ij}|}{|Q_{ij}|} h_k(\beta, \lambda) - \sum_{i,j} \sum_{\xi_k \in C_{ij}} h_k(\beta, \lambda)| \\ &\leq \sum_{i,j} |C_{ij}| \epsilon_1 (2^{j-1} + 2^{i-1}) A \\ &\leq \sum_{i,j} |C_{ij}| \epsilon_1 (2^{j-1} + 2^{i-1}) H(\beta_{\mathtt{anc}}, \lambda_{\mathtt{anc}}). \end{aligned} \tag{13}$$

We also need the following claim to proceed our proof.

Claim 1 $\sum\limits_{i,j} |C_{ij}| 2^i \leq 3n$ *and* $\sum\limits_{i,j} |C_{ij}| 2^j \leq 3n$

Based on Claim 1, we can rewrite (13) as

$$n|\tilde{H}(\beta, \lambda) - H(\beta, \lambda)| \leq 3n\epsilon_1 H(\beta_{\texttt{anc}}, \lambda_{\texttt{anc}}).$$

So the statement of Lemma 3 is true. □

*Proof.***(of Claim 1)** By the definition of C_{ij}, we have

$$\begin{array}{ll} 2^i A = A, & \text{if } i = 0; \\ 2^i A \leq 2a_k\left(\beta_{\texttt{anc}}\right), \forall \xi_k \in C_{ij}, & \text{if } i \geq 1. \end{array}$$

So we have $2^i A \leq A + 2a_k(\beta_{\texttt{anc}}, \lambda_{\texttt{anc}})$ for all $0 \leq i \leq N$ and $\xi_k \in C_{ij}$. Overall, we have

$$\begin{aligned} \sum_{i,j=0}^{N} \left|C_{ij}\right| 2^i A &= \sum_{i,j=0}^{N} \sum_{\xi_k \in C_{ij}} 2^i A \\ &\leq \sum_{i,j=0}^{N} \sum_{\xi_k \in C_{ij}} \left(2a_k\left(\beta_{\texttt{anc}}, \lambda_{\texttt{anc}}\right) + A\right) \\ &= 2nA + nA = 3nA. \end{aligned}$$

Thus $\sum\limits_{i,j} |C_{ij}| 2^i \leq 3n$, and we can prove $\sum\limits_{i,j} |C_{ij}| 2^j \leq 3n$ via the same manner. □

2 Omitted Details for Applications

We discuss more details for **SVM in the hypercube**. Suppose $\mathbb{X} = [0, l]^d$ is a d-dimensional hypercube and $p = 1$, then by [8] and the strong duality of the linear programming, we know that the WDRO of SVM is equivalent to

$$\begin{array}{lll} \inf\limits_{\substack{\beta, \lambda, s_i, p_i^+, p_i^-, \\ z_i^+, z_i^-}} & \lambda\theta + \frac{1}{n}\sum_{i=1}^{n} s_i & \\ \text{s.t.} & 1 + l \cdot e^\top z_i^+ + x_i^\top p_i^+ \leq s_i & \\ & 1 + l \cdot e^\top z_i^- + x_i^\top p_i^- - \gamma\lambda \leq s_i & \\ & -y_i\beta - p_i^+ \leq z_i^+, \vec{0} \leq z_i^+ & \\ & y_i\beta - p_i^- \leq z_i^-, \vec{0} \leq z_i^- & \\ & \left\|p_i^+\right\|_* \leq \lambda, \left\|p_i^-\right\|_* \leq \lambda, 0 \leq s_i & i \in [n] \end{array} \tag{14}$$

where $e = [1, \ldots, 1] \in \mathbb{R}^d$. Hence $h_i(\beta, \lambda)$ is equivalent to

$$\begin{array}{lll} \inf\limits_{\substack{p_i^+, p_i^-, \\ z_i^+, z_i^-}} & \max\{0, 1 + l \cdot e^\top z_i^+ + x_i^\top p_i^+, 1 + l \cdot e^\top z_i^- + x_i^\top p_i^- - \gamma\lambda\} & \\ \text{s.t.} & -y_i\beta - p_i^+ \leq z_i^+, \vec{0} \leq z_i^+ & \\ & y_i\beta - p_i^- \leq z_i^-, \vec{0} \leq z_i^- & \\ & \left\|p_i^+\right\|_* \leq \lambda, \left\|p_i^-\right\|_* \leq \lambda & i \in [n]. \end{array} \tag{15}$$

In this task we have

- $\kappa \equiv 0$, $\mathtt{C}(\beta) = \|\beta\|_*$ and $R \leq \gamma + l \cdot d^{\frac{1}{p}}$;
- $h_i(\beta, \lambda)$ is the optimal value of a constrained convex programming in (15);
- For any $z_i^+, z_i^-, p_i^+, p_i^-$ satisfying the constraints in (15), $\max\{0, 1 + l \cdot e^\top z_i^+ + x_i^\top p_i^+, 1 + l \cdot e^\top z_i^- + x_i^\top p_i^- - \gamma\lambda\}$ is an upper bound for $h_i(\beta, \lambda)$ and thus can be viewed as $b_i(\beta, \lambda)$. Here we propose a simple strategy for determining the values for these variables.

If $\|\beta\|_* > \lambda$, set

$$p_i^+ = -\tfrac{\lambda y_i \beta}{\|\beta\|_*}, p_i^- = \tfrac{\lambda y_i \beta}{\|\beta\|_*},$$
$$z_i^+ = \max\{(-y_i + \tfrac{\lambda y_i}{\|\beta\|_*})\beta, \vec{0}\}, z_i^- = \max\{(y_i - \tfrac{\lambda y_i}{\|\beta\|_*})\beta, \vec{0}\};$$

otherwise, set

$$p_i^+ = -y_i\beta, p_i^- = y_i\beta,$$
$$z_i^+ = \vec{0}, z_i^- = \vec{0}.$$

- $a_i(\beta, \lambda) = \ell(\beta, \xi_i)$.

3 Experiments

Our experiments were conducted on a server equipped with 2.4GHZ Intel CPUs and 256GB main memory. The algorithms are implemented in Python. We use the MOSEK [1] to solve the tractable reformulations of WDROs.

Compared methods We compare our dual coreset method DualCore with the uniform sampling approach (UniSamp) and the approach that directly runs on whole dataset (Whole).

Datasets We test the algorithms for the SVM and logistic regression problems on two real datasets: Mnist[7] and Letter[3]. To simulate the scenarios where the datasets are contaminated, we perform poisoning attacks to the training set of Letter. Specifically, we use the Min-Max attack from [6] and Alfa attack from [9]. We add the standard Gaussian noise $\mathcal{N}(0, 1)$ to the training set of Mnist and randomly flip 10% of the labels. The dual coreset algorithm for the robust regression problem is evaluated on the real dataset Appliances Energy[2].

Results Let m and n be the coreset size and the training set size, respectively. We set $\sigma := \frac{m}{n}$ to indicate the compression rate and fix the parameter $\gamma = 7$ for all the instances (recall that γ is used for defining the distance $\mathtt{d}(\xi_i, \xi_j) = \|x_i - x_j\| + \frac{\gamma}{2}|y_i - y_j|$). We vary the radius θ of the Wasserstein ball for different tasks. The experiment of each instance were repeated by 50 independent trials. For the WDRO logistic regression and SVM problems, we report the averaged test accuracy and the standard deviation in table 1, 3, 5, 7 and 9, where the higher accuracy of UniSamp and DualCore is written in bold for each instance. The results suggest that our dual coreset method outperforms the uniform sampling method with a higher accuracy in most cases. For the WDRO robust regression task, we report the averaged test Huber loss and the standard deviation in table 11, where the lower loss of UniSamp and DualCore is written in bold for each instance. The results suggest that our dual coreset method outperforms the uniform sampling method with a lower Huber loss in most cases. We also record the normalized CPU time (over the CPU time of Whole) in table 2, 4, 6, 8, 10 and 12.

References

[1] M. ApS. *MOSEK Optimizer API for Python 9.3.20*, 2019. URL `https://docs.mosek.com/latest/pythonapi/index.html`.

[2] L. M. Candanedo, V. Feldheim, and D. Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.

[3] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[4] R. Gao and A. J. Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *arXiv preprint arXiv:1604.02199*, 2016.

[5] W. Hoeffding. Probability inequalities for sums of bounded random variables. In *The collected works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

[6] P. W. Koh, J. Steinhardt, and P. Liang. Stronger data poisoning attacks break data sanitization defenses. *CoRR*, abs/1811.00741, 2018.

[7] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database, 2010.

Table 1: Experimental results of the WDRO logistic regression on Mnist with $\sigma = 0.5\%, \theta = 0.3$

	Whole	UniSamp	DualCore
0 vs 1	99.76%	94.14±6.85%	**94.37±8.05%**
0 vs 2	98.13%	**87.1±10.1%**	84.98±11.28%
0 vs 3	98.78%	84.03±12.88%	**84.38±12.57%**
0 vs 4	99.1%	87.82±12.79%	**87.91±12.11%**
0 vs 5	97.52%	**76.57±13.48%**	75.45±13.76%
0 vs 6	97.94%	**83.17±12.55%**	81.05±12.84%
0 vs 7	99.27%	86.43±13.33%	**90.78±8.8%**
0 vs 8	98.28%	85.57±12.99%	**86.07±11.94%**
0 vs 9	98.53%	**88.35±10.58%**	87.14±9.9%
1 vs 2	96.87%	79.85±13.47%	**86.26±10.54%**
1 vs 3	97.58%	**83.75±14.73%**	83.17±14.87%
1 vs 4	98.97%	87.73±12.73%	**87.85±12%**
1 vs 5	98.03%	76.22±13.88%	**81.33±13.31%**
1 vs 6	99%	85.71±12.17%	**89.32±8.2%**
1 vs 7	97.73%	84.2±14.22%	**87.05±12.5%**
1 vs 8	95.86%	78.95±14.21%	**79.13±12.93%**
1 vs 9	98.65%	87.38±13.33%	**87.44±11.97%**
2 vs 3	95.91%	75.71±12.87%	**76.85±12.23%**
2 vs 4	97.7%	78.12±14.26%	**79.52±14.36%**
2 vs 5	96.8%	**75.53±14.6%**	73.93±14.32%
2 vs 6	96.19%	72.44±12.3%	**75.23±11.11%**
2 vs 7	96.39%	78.54±12.56%	**86.15±10.48%**
2 vs 8	95.96%	69.08±12.62%	**69.18±13.9%**
2 vs 9	97.25%	**82.43±12.68%**	81.92±12.8%
3 vs 4	98.54%	77.33±16.52%	**85.86±13.62%**
3 vs 5	93.4%	62.31±10.81%	**66.12±10.17%**
3 vs 6	98.27%	83.06±14.52%	**88.39±9%**
3 vs 7	97.41%	81.94±10.87%	**81.99±14.57%**
3 vs 8	93.84%	65.56±11.81%	**69.62±11.83%**
3 vs 9	96.95%	78.27±15.1%	**79.13±14.22%**
4 vs 5	97.67%	70.83±13.86%	**75.95±13.76%**
4 vs 6	98.2%	72.75±14.89%	**72.98±14.89%**
4 vs 7	97.44%	72.55±14.45%	**75.63±14.79%**
4 vs 8	98.17%	74.95±14.93%	**77.88±14.33%**
4 vs 9	93.88%	59.1±9.01%	**62.03±8.48%**
5 vs 6	96.88%	73.08±14.19%	**77.26±13.35%**
5 vs 7	98.65%	74.25±15.56%	**78.55±12.74%**
5 vs 8	93.8%	**66.4±11.22%**	65.91±11.5%
5 vs 9	97.39%	70.1±14.81%	**72.16±13.33%**
6 vs 7	99.49%	84.12±12.65%	**87.87±11.64%**
6 vs 8	97.91%	**78.85±13.94%**	76.58±14.58%
6 vs 9	99.48%	**79.76±16.29%**	78.97±15.41%
7 vs 8	97.75%	79.98±14.22%	**80.97±13.48%**
7 vs 9	93.14%	66.09±11.99%	**67.68±12.43%**
8 vs 9	96.13%	72.12±13.57%	**75.52±13.5%**

Table 2: Normalized CPU time of the WDRO logistic regression on MNIST with $\sigma = 0.5\%, \theta = 0.3$

	UNISAMP	DUALCORE
0 vs 1	0.01	0.021
0 vs 2	0.011	0.024
0 vs 3	0.013	0.029
0 vs 4	0.011	0.024
0 vs 5	0.008	0.017
0 vs 6	0.007	0.017
0 vs 7	0.008	0.018
0 vs 8	0.009	0.019
0 vs 9	0.011	0.024
1 vs 2	0.009	0.021
1 vs 3	0.008	0.018
1 vs 4	0.008	0.017
1 vs 5	0.007	0.017
1 vs 6	0.009	0.019
1 vs 7	0.011	0.023
1 vs 8	0.007	0.016
1 vs 9	0.008	0.019
2 vs 3	0.01	0.021
2 vs 4	0.007	0.017
2 vs 5	0.008	0.021
2 vs 6	0.01	0.023
2 vs 7	0.011	0.024
2 vs 8	0.011	0.024
2 vs 9	0.008	0.019
3 vs 4	0.008	0.018
3 vs 5	0.005	0.013
3 vs 6	0.007	0.016
3 vs 7	0.009	0.02
3 vs 8	0.01	0.023
3 vs 9	0.01	0.021
4 vs 5	0.01	0.021
4 vs 6	0.009	0.02
4 vs 7	0.007	0.016
4 vs 8	0.008	0.018
4 vs 9	0.007	0.017
5 vs 6	0.01	0.021
5 vs 7	0.011	0.025
5 vs 8	0.01	0.023
5 vs 9	0.012	0.025
6 vs 7	0.008	0.018
6 vs 8	0.009	0.021
6 vs 9	0.007	0.015
7 vs 8	0.01	0.021
7 vs 9	0.009	0.019
8 vs 9	0.008	0.018

Table 3: Experimental results of the WDRO logistic regression on Letter under Min-Max attack with $\theta = 0.3$

σ	UniSamp	DualCore
1%	79.15±14.32%	**83.86±9.67%**
2%	87.66±8.74%	**87.81±7.03%**
3%	89.32±4.44%	**89.54±7.89%**
4%	89.71±5.28%	**90.1±5.06%**
5%	90.52±4.29%	**91.49±4.1%**
6%	91.55±3.63%	**92.36±2.56%**
7%	91.19±3.68%	**91.67±2.92%**
8%	**92.51±2.82%**	91.59±3.01%
9%	**92.33±2.75%**	91.57±2.56%
10%	91.86±2.79%	**92.57±2.08%**

Table 4: Normalized CPU time of the WDRO logistic regression on the Letter under Min-Max attack with $\theta = 0.3$

σ	UniSamp	DualCore
1%	0.04	0.103
2%	0.053	0.13
3%	0.062	0.151
4%	0.084	0.185
5%	0.105	0.237
6%	0.118	0.257
7%	0.143	0.329
8%	0.164	0.346
9%	0.132	0.278
10%	0.121	0.275

Table 5: Experimental results of the WDRO logistic regression on Letter under Alfa attack with $\theta = 0.3$

σ	UniSamp	DualCore
1%	78.28±12.22%	**79.86±13.56%**
2%	79.69±11.74%	**83.17±10.4%**
3%	81.98±13.56%	**84.89±10.4%**
4%	87.06±8.89%	**87.63±6.38%**
5%	86.14±9.29%	**87.16±8.53%**
6%	86.9±7.44%	**88.59±6.45%**
7%	**87.9±7.08%**	86.86±6.85%
8%	88.23±5.22%	**88.39±4.52%**
9%	88.18±5.67%	**88.63±4.43%**
10%	**89.33±6.46%**	87.44±5.05%

Table 6: Normalized CPU time of the WDRO logistic regression on Letter under Alfa attack with $\theta = 0.3$

σ	UniSamp	DualCore
1%	0.03	0.067
2%	0.033	0.076
3%	0.041	0.095
4%	0.045	0.109
5%	0.058	0.122
6%	0.06	0.137
7%	0.072	0.152
8%	0.092	0.201
9%	0.125	0.25
10%	0.099	0.217

Table 7: Experimental results of the WDRO SVM on LETTER under ALFA attack with $\theta = 0.1$

σ	UNISAMP	DUALCORE
1%	**80.29±13.98%**	79.23±13%
2%	83.86±13.7%	**87.8±11.74%**
3%	89.95±11.23%	**92.78±7.61%**
4%	91.47±9.37%	**92.46±6.54%**
5%	90.89±9.12%	**92.36±9.05%**
6%	**95.5±4.35%**	94.94±5.49%
7%	94.01±6.85%	**95.99±2.65%**
8%	95.61±5.7%	**96.1±2.43%**
9%	94.91±6.09%	**96.43±2.23%**
10%	95.27±5.72%	**95.97±3.76%**

Table 8: Normalized CPU time of the WDRO SVM on LETTER under ALFA attack with $\theta = 0.1$

σ	UNISAMP	DUALCORE
1%	0.137	0.554
2%	0.137	0.561
3%	0.121	0.546
4%	0.125	0.501
5%	0.133	0.503
6%	0.168	0.515
7%	0.228	0.695
8%	0.349	0.955
9%	0.327	0.984
10%	0.163	0.556

Table 9: Experimental results of the WDRO SVM on LETTER under MIN-MAX attack with $\theta = 0.2$

σ	UNISAMP	DUALCORE
1%	82.02±15.31%	**85.17±13.88%**
2%	90.44±9.08%	**93.13±2.26%**
3%	90.29±10.71%	**92.17±6.62%**
4%	91.29±9.01%	**93.7±2.26%**
5%	93.55±2.43%	**93.91±1.72%**
6%	**94.17±2.18%**	93.13±6.47%
7%	92.68±7.33%	**94.39±1.47%**
8%	94.15±2.05%	**94.2±1.36%**
9%	**94.26±1.61%**	94.04±1.28%
10%	93.99±1.53%	**94.2±1.67%**

Table 10: Normalized CPU time of the WDRO SVM on LETTER under MIN-MAX attack with $\theta = 0.2$

σ	UNISAMP	DUALCORE
1%	0.118	0.517
2%	0.122	0.516
3%	0.11	0.445
4%	0.118	0.458
5%	0.159	0.537
6%	0.245	0.804
7%	0.33	0.965
8%	0.297	0.871
9%	0.159	0.519
10%	0.218	0.647

Table 11: Experimental results of the WDRO robust regression on Appliances Energy with $\theta = 100$

σ	UniSamp	DualCore
1%	33.0933±1.8918	**32.3245±1.937**
2%	31.4399±1.5614	**30.7886±1.2459**
3%	31.3852±0.6885	**30.5185±0.4625**
4%	31.5143±0.4824	**31.0308±0.3113**
5%	31.036±0.507	**30.401±0.2476**
6%	31.5388±0.3296	**31.0017±0.1913**
7%	32.2394±0.311	**31.7412±0.1504**
8%	30.225±0.2345	**29.8135±0.1503**
9%	30.0463±0.2292	**29.6167±0.1098**
10%	31.1906±0.2257	**30.8201±0.107**

Table 12: Normalized CPU time of the WDRO robust regression on Appliances Energy with $\theta = 100$

σ	UniSamp	DualCore
1%	0.039	0.145
2%	0.076	0.224
3%	0.068	0.192
4%	0.072	0.207
5%	0.088	0.236
6%	0.097	0.244
7%	0.103	0.259
8%	0.157	0.348
9%	0.147	0.299
10%	0.184	0.374

[8] S. Shafieezadeh-Abadeh, D. Kuhn, and P. M. Esfahani. Regularization via mass transportation. *Journal of Machine Learning Research*, 20(103):1–68, 2019.

[9] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, 2015. doi: 10.1016/j.neucom.2014.08.081.