Robust Distributed Gradient Descent with Arbitrary Number of Byzantine Attackers

Xinyang Cao & Lifeng Lai

Dept. of Elec. and Comp. Engr., University of California, Davis

Contact Information:
Dept. of Elec. and Comp. Engr.
University of California
Davis, CA, USA
Email: {xyacao,lflai}@ucdavis.edu

Abstract

Due to the grow of modern dataset size and the desire to harness computing power of multiple machines, there is a recent surge of interest in the design of distributed machine learning algorithms. However, distributed algorithms are sensitive to Byzantine attackers who can send falsified data to prevent the convergence of algorithms or lead the algorithms to converge to value of the attackers' choice. Our novel algorithm can deal with an arbitrary number of Byzantine attackers.

Introduction

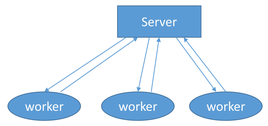

Figure 1: DML

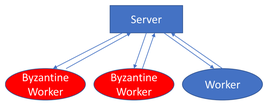

Figure 2: DML with Byzantine attack

- The design of distributed optimization attracted recent research interests [1, 2, 3] due to:
  – dataset is too big to fit into one machine;
  – to harness the computing power of multiple machines.
- These bring security issues:
  – a worker can produce arbitrary output;
  – a worker could be a Byzantine attacker.
- Recent work [4, 5] can deal with up to half of workers that are Byzantine workers.

Goal : Propose a new robust distributed gradient descent algorithm that can converge with **arbitrary number** of Byzantine workers.

Model

- Goal:

$$\theta^* \in \arg\min_{\theta\in\Theta} F(\theta) \triangleq \mathbb{E}[f(X,\theta)]. \tag{1}$$

- Approximate goal:

$$\min_{\theta\in\Theta} \frac{1}{N}\sum_{i=1}^{N} f(X_i,\theta). \tag{2}$$

- Model for typical distributed optimization:
  – m workers, each has a subset of data can compute:

$$\nabla\overline{f}^{(j)}(\theta_{t-1}) = \frac{1}{|\mathcal{S}_j|}\sum_{i\in\mathcal{S}_j}\nabla f(X_i,\theta_{t-1}), \tag{3}$$

  – one server computes:

$$\theta_t = \theta_{t-1} - \eta\frac{1}{m}\sum_{i=1}^{m}\nabla\overline{f}^{(j)}(\theta_{t-1}) \tag{4}$$

- Server receive data with Byzantine attacks:

$$g_t^{(j)}(\theta_{t-1}) = \begin{cases} \nabla\overline{f}^{(j)}(\theta_{t-1}) & j\notin\mathcal{B}_t \\ \star & j\in\mathcal{B}_t \end{cases}, \tag{5}$$

Existing Work

Comparing data received from all workers among themselves:

- [4] computes $s(i)=\sum_j\|g^{(i)}-g^{(j)}\|^2$ for all m workers, where $g^{(j)}$ belongs to $m-p-2$ closest vectors to $g^{(i)}$, then chooses the $g^{(i)}$ which has the smallest $s(i)$;
- [5] uses geometric median of means of the gradients. At iteration $t-1$, the update gradient is

$$A_k = med\left\{\frac{1}{b}\sum_{j=1}^{b}g_t^{(j)}(\theta_{t-1}),\ldots,\frac{1}{b}\sum_{j=n-b+1}^{b}g_t^{(j)}(\theta_{t-1})\right\} \tag{6}$$

Our Proposed Algorithm

Comparing data received from all workers with server:

- Server computes a gradient $\nabla\overline{f}^{(0)}(\theta)$ with a small subset of data;
- Server compares each $g^{(j)}(\theta)$ with $\nabla\overline{f}^{(0)}(\theta)$, accepts the one that is close to $\nabla\overline{f}^{(0)}(\theta)$ and rejects the one that is far away;
- Average all accept gradient and $\nabla\overline{f}^{(0)}(\theta)$ for update.

Algorithm

Algorithm Iteration $t\geq 1$

Parameter server:
Initialize: Randomly selects $\theta_0\in\Theta$;randomly selects $\mathcal{S}_0$;
1: Broadcasts the current model parameter estimator θ_{t-1} to all working machines;
2: Computes $\nabla\overline{f}_t^{(0)}(\theta_{t-1})$ using $\mathcal{S}_0$;
3: Waits to receive all the gradients from the m machines;
Let $g_t^{(j)}(\theta_{t-1})$ denote the value received from machine j;
4: Compares $g_t^{(j)}(\theta_{t-1})$ with $\nabla\overline{f}_t^{(0)}(\theta_{t-1})$;If
$\| g_t^{(j)}(\theta_{t-1})-\nabla\overline{f}_t^{(0)}(\theta_{t-1})\|\leq\xi_1\|\nabla\overline{f}_t^{(0)}(\theta_{t-1})\|$,
the server accepts it and sets it to be $q_t^{(l)}(\theta_{t-1})$;
5: Assume there are k acceptable value, then
$G(\theta_{t-1})\leftarrow\frac{1}{k+1}\left(\sum_{l=1}^{k}q_t^{(l)}(\theta_{t-1})+\nabla\overline{f}_t^{(0)}(\theta_{t-1})\right)$;
6:Updates $\theta_t\leftarrow\theta_{t-1}-\eta G(\theta_{t-1})$;

Working machine j:
1: Computes the gradient $\nabla\overline{f}^{(j)}(\theta_{t-1})$;
2: If machine j is honest,
it sends $\nabla\overline{f}^{(j)}(\theta_{t-1})$ back to the server;
If machine j is compromised,
it sends the value determined by the attacker;

Table 1: Proposed algorithm

Convergence Analysis

Main idea

- $\nabla\overline{f}^{(0)}(\theta)$ is a good estimate of $\nabla F(\theta)$;
- All accepted gradients are close to $\nabla\overline{f}^{(0)}(\theta)$ due to the algorithm;
- $G(\theta)$ is a good estimate of $\nabla F(\theta)$.

Details

Assumption 1. The population risk function $F:\Theta\to\mathbb{R}$ is L-strongly convex, and differentiable over Θ with M-Lipschitz gradient.

Assumption 2. There exist positive constants σ_1 and α_1 such that for any unit vector $v\in B$, $\langle\nabla f(X,\theta^*),v\rangle$ is sub-exponential with σ_1 and α_1, that is,

$$\sup_{v\in B}\mathbb{E}[\exp(\lambda\langle\nabla f(X,\theta^*),v\rangle)]\leq e^{\sigma_1^2\lambda^2/2},\forall|\lambda|\leq 1/\alpha_1,$$

where B denotes the unit sphere $\theta:\|\theta\|_2=1$.

Assumption 3. There exist positive constants σ_2 and α_2 such that for any $\theta\in\Theta$ with $\theta\neq\theta^*$ and any unit vector $v\in B$, $\langle h(X,\theta)-\mathbb{E}[h(X,\theta)],v\rangle/\|\theta-\theta^*\|$ is sub-exponential with σ_2 and α_2, that is,

$$\sup_{\theta\in\Theta,v\in B}\mathbb{E}\left[\exp\left(\frac{\lambda\langle h(X,\theta)-E[h(X,\theta)],v\rangle}{\|\theta-\theta^*\|}\right)\right]\leq e^{\sigma_2^2\lambda^2/2},\forall|\lambda|\leq\frac{1}{\alpha_2}.$$

Assumption 4. For any $\delta\in(0,1)$, there exists an $M'=M'(\delta)$ such that

$$\Pr\left\{\sup_{\theta,\theta'\in\Theta:\theta\neq\theta'}\frac{\|\nabla f(X,\theta)-\nabla f(X,\theta')\|}{\|\theta-\theta'\|}\leq M'\right\}\geq 1-\frac{\delta}{4}.$$

Lemma 1. For an arbitrary number of attackers, the distance between $G(\theta)$ and $\nabla F(\theta)$ is bounded as

$$\begin{aligned}\|G(\theta)-\nabla F(\theta)\|\leq{}&(1+\xi_1)\|\nabla F(\theta)-\nabla\overline{f}^{(0)}(\theta)\|\\&+\xi_1\|\nabla F(\theta)-\nabla F(\theta^*)\|,\forall\theta.\end{aligned} \tag{7}$$

Theorem 1. Under above assumptions, regardless the number of attackers, the probability at least $1-\delta$ that

$$\|\theta_t-\theta^*\|\leq(1-\rho)^t\|\theta_0-\theta^*\|+(2\eta\Delta_1+2\eta\xi_1\Delta_1)/\rho,$$

in which $\rho=1-\left(\sqrt{1-\frac{L^2}{4M^2}}+4\Delta_2\eta+\eta\xi_1(4\Delta_2+M)\right)$, and

$$\Delta_1=\sqrt{2}\sigma_1\sqrt{(d\log 6+\log(3/\delta))/|\mathcal{S}_0|}, \tag{8}$$

$$\Delta_2=\sqrt{2}\sigma_2\sqrt{\frac{d\log 6+d\log(\frac{M\vee M'}{\sigma_2}+0.5d\log(\frac{|\mathcal{S}_0|}{d})+\log(\frac{3}{\delta})+\log(r\sqrt{d})}{|\mathcal{S}_0|}}. \tag{9}$$

Numerical Results

Focus on $Y_i=X_i^T\theta^*+\epsilon_i, i=1,2,\cdots,N$. $d=20$. $N=10000$. $m=100$. $|\mathcal{S}_0|=50$. $\xi_1=0.975$. $\theta^*\overset{i.i.d.}{\sim}\mathcal{N}(0,4)$. $X_i\sim\mathcal{N}(0,16)$

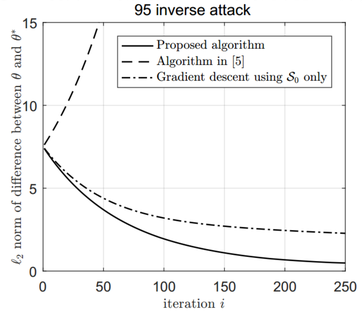

Figure 3: Inverse attack.

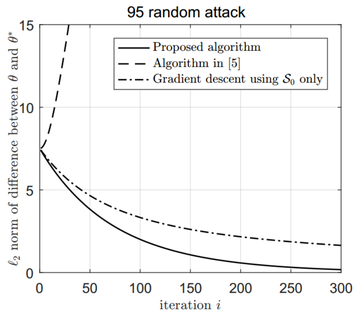

Figure 4: Random attack.

Conclusions

- Proposed a robust gradient descent algorithm that can tolerant an arbitrary number of Byzantine attackers
- Proposed algorithm converges to the true value
- Provided numerical examples to illustrate the performance of the proposed algorithm and compared it with those of other algorithms.

References

[1] M. Jordan, J. Lee, and Y. Yang, "Communication-efficient distributed statistical inference," *arXiv preprint arXiv:1605.07689*, Nov. 2016.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[3] P. Moritz, R. Nishihara, I. Stoica, and M. Jordan, "Sparknet: Training deep networks in spark," *arXiv preprint arXiv:1511.06051*, Feb. 2015.

[4] P. Blanchard, E. Mhamdi, R. Guerraoui, and J. Stainer, "Byzantine-tolerant machine learning," *arXiv preprint arXiv:1703.02757*, Mar. 2017.

[5] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *arXiv preprint arXiv:1705.05491*, May 2017.

Acknowledgements

The work of X. Cao and L. Lai was supported by National Science Foundation under grants CCF-1717943 and ECCS-1711468.