ABSTRACT
In this paper, we consider the Byzantine-robust stochastic optimization problem defined over a decentralized network, where the agents collaboratively minimize the summation of expectations of stochastic local cost functions, but some of the agents are unreliable. Due to data corruptions, equipment failures or cyber-attacks, these Byzantine agents can send faulty values to their neighbors and bias the optimization process. Our key idea to handle the Byzantine attacks is to formulate a total variation (TV) norm-penalized approximation of the Byzantine-free problem, where the penalty term forces the local models of regular agents to be close, but also allows the existence of outliers from the Byzantine agents. A stochastic subgradient method is applied to solve the penalized problem. We prove that the proposed method converges to a near-optimal solution of the Byzantine-free problem under mild assumptions, and the gap is determined by the number of Byzantine agents and the network topology. Numerical experiments corroborate the theoretical analysis, as well as demonstrate the robustness of proposed method to Byzantine attacks and its superior performance over existing methods.

Index Terms— Decentralized stochastic optimization, Byzantine attacks, robustness

1. INTRODUCTION
In recent years, decentralized stochastic optimization has become a popular research topic in the signal processing and machine learning communities. With the rapidly increasing number of distributed devices and volume of generated data, traditional signal processing and machine learning approaches, which rely on a central controller to collect the data samples or coordinate the optimization process, suffer from privacy and scalability issues [1]. In decentralized stochastic optimization, every device (called as agent thereafter) learns its own model using its local data samples, and regularly exchanges its model with neighboring agents so as to achieve consensus. This scheme is favorable in privacy preservation since the data samples are kept local, and does not rely on any central controller that could be a system bottleneck. Existing decentralized stochastic optimization methods include decentralized stochastic gradient descent (DPSGD) [2], stochastic subgradient projection [3], dual averaging [4], mirror descent [5], etc. Asynchronous algorithms are developed in [6, 7] to reduce the idle time, and variance reduction techniques are proposed in [8–10] to improve the convergence rate. Decentralized stochastic optimization methods are shown to be superior than their centralized counterparts on the highly nonconvex problems of training large-scale neural networks [11] when the communication links are subject to high latency and limited bandwidth.

However, the lack of centralized coordination in decentralized stochastic optimization also raises concerns on robustness. Some of the agents might be malfunctioning or even malicious. Due to data corruptions, equipment failures or cyber-attacks, they can send faulty values to their neighbors and bias the optimization process. We consider a general Byzantine attack model [12], in which the Byzantine agents are omniscient and can arbitrarily modify the values sent to other agents. Such a model imposes no restrictions on the attacks and is worse-case. The purpose of this paper is to develop a Byzantine-robust decentralized stochastic optimization method.

Most of the existing decentralized stochastic optimization methods are vulnerable to Byzantine attacks. Take DPSGD as an example. At every iteration, every agent averages the models received from its neighbors, followed by a stochastic gradient step on the cost function constructed from one local data sample (or a batch of them), to update its local model [2]. When the Byzantine agents send well-designed faulty values instead of the current models, they are able to lead the regular agents to end up with incorrect results.

Byzantine-robust decentralized deterministic optimization methods have been developed in [13, 14], where at every iteration every regular agent uses all of its local data samples, instead one or a batch. The work of [13] proposes ByRDiE, in which every regular agent utilizes coordinate-wise trimmed mean to screen outliers in the received models, and then applies coordinate gradient descent to update its local model. The one-coordinate-at-a-time update of ByRDiE is inefficient for high-dimensional problems [15]. To address this issue, the work of [14] proposes BRIDGE, which allows every regular agent to update all the coordinates of its local model at every iteration. Although these two algorithms are originally developed for decentralized deterministic optimization, they can also be applied to the decentralized stochastic setting according to our numerical experiments. However, to the best of our knowledge, most of the existing works do not explicitly consider the Byzantine-robust decentralized stochastic optimization problem; see for reference the recent survey paper [15].

There are some works that consider the Byzantine-robust centralized stochastic optimization problem, where a central controller aggregates the information from the agents and coordinates the optimization process. The main idea of these works is to modify the stochastic gradient method with robust aggregation rules. To be specific, at every iteration, the central controller sends the current model to all the agents, the regular agents send back their local stochastic gradients, while the Byzantine agents may send back faulty values. When the local data samples are independently and identically distributed (i.i.d.), the local stochastic gradients are also i.i.d. and the central controller can obtain a reliable approximation to the average of the local stochastic gradients through aggregating all the received values with trimmed mean, geometric median, or other robust aggregation rules [16,17]. However, this idea is not directly applicable to decentralized stochastic optimization. Since there is no central controller to maintain a common model, the regular agents have to evaluate their local stochastic gradients at different points. Therefore, even though the local data samples are i.i.d. the local stochastic gradients are not necessarily so, and thus the robust aggregation rules have no theoretical guarantee in this case.
This paper develops a Byzantine-robust decentralized stochastic optimization method, where the network is fully decentralized and contains an unknown number of Byzantine agents, the local data samples at the regular agents are not necessarily i.i.d. and only one data sample (or a batch of them) is available for every regular agent at every iteration. The key idea is to formulate a total variation (TV) norm-penalized approximation of the Byzantine-free problem, where the penalty term forces the local models of regular agents to be close, but also allows the existence of outliers from the Byzantine agents. A stochastic subgradient method is applied to solve the TV norm-penalized approximation of the Byzantine-free problem, where \( \|x_i - x_j\|_{1} \geq \lambda \) for every pair of regular neighbors \((i, j)\), \(x_i\) and \(x_j\) are forced to be close through introducing the TV norm penalty \(\sum_{i \in \mathcal{R}} \sum_{j \in \mathcal{R}_i} \|x_i - x_j\|_{1}\). The larger \(\lambda\) is, the closer \(x_i\) and \(x_j\) are. On the other hand, the TV norm penalty also allows some pairs of \(x_i\) and \(x_j\) to be different, which is important when the Byzantine agents are present as we will discuss later.

Since calculating the full gradients is time-consuming or even impossible, we solve (3) with the stochastic subgradient method. At time \(k + 1\), every regular agent \(i\) updates its local model \(x_i^{k+1}\) as

\[
x_i^{k+1} = x_i^k - \alpha k \left( \nabla F(x_i^k, \xi_i^k) + \sum_{j \in \mathcal{R}_i} \text{sign}(x_i^k - x_j^k) + \nabla f_0(x_i^k) \right),
\]

where \(\xi_i^k\) corresponds to the random data sample chosen by agent \(i\) at time \(k\), \(\text{sign}(\cdot)\) is the element-wise sign function, and \(\alpha k\) is the step size. Given \(\beta \in \mathbb{R}\), \(\text{sign}(\beta)\) equals to 1 when \(\beta > 0\), -1 when \(\beta < 0\), and an arbitrary value within \([-1, 1]\) when \(\beta = 0\). The step size is nonnegative, diminishing, square-summable but not summable, i.e.,

\[
0 \leq \alpha_{k+1} \leq \alpha_k \sum_{k=0}^{\infty} \alpha^k \rightarrow \infty \quad \text{and} \quad \sum_{k=0}^{\infty} (\alpha^k)^2 < \infty.
\]

Observe that (4) is fully decentralized. To update \(x_i^{k+1}\), a regular agent \(i\) needs to evaluate its own local stochastic gradient \(\nabla F(x_i^k, \xi_i^k)\) and gradient \(\nabla f_0(x_i^k)\), as well as combine the models \(\{x_j^k, j \in \mathcal{R}_i\}\) received from its regular neighbors.

Now we consider how (4) performs when the Byzantine agents are present. A Byzantine agent \(j\) will not send its true model to its neighbors at time \(k\). Instead, it sends an arbitrary value \(z_j^k\). In this case, (4) becomes

\[
x_i^{k+1} = x_i^k - \alpha k \left( \nabla F(x_i^k, \xi_i^k) + \lambda \sum_{j \in \mathcal{R}_i} \text{sign}(x_i^k - z_j^k) + \nabla f_0(x_i^k) \right),
\]

The resulting Byzantine-robust decentralized stochastic optimization algorithm is outlined in Algorithm 1. Observe in (5) that the elements of \(\text{sign}(x_i^k - z_j^k)\) are in the range of \([-1, 1]\), such that the influence of the faulty value \(z_j^k\) is limited, although \(z_j^k\) can be arbitrary. We will theoretically justify the robustness of the proposed method to Byzantine attacks in the subsequent section.

**Algorithm 1**

**Input:** \(x_i^0 \in \mathbb{R}^p\) for \(i \in \mathcal{R}\), \(\lambda > 0\) and \(\{\alpha_k, k = 0, 1, \cdots\}\).

1: \(for\ k = 0, 1, \cdots, \) every regular agent \(i \in \mathcal{R}\) do
2: Broadcast its current model \(x_i^k\) to all the neighbors.
3: Receive \(x_j^k\) from regular neighbors \(j \in \mathcal{R}_i\) and \(z_j^k\) from Byzantine neighbors \(j \in \mathcal{B}_i\).
4: Update local iterate \(x_i^{k+1}\) according to (5).
5: end for
In this section, we theoretically analyze the performance of our proposed method under Byzantine attacks. We make the following assumptions, which are common for convergence analysis of decentralized stochastic gradient methods.

**Assumption 1. (Strong Convexity)** Local cost functions $\mathbb{E}[F(\tilde{x}, \xi_i)]$ and regularization term $f_0(\tilde{x})$ are strongly convex with constants $u_i$ and $u_0$, respectively.

**Assumption 2. (Lipschitz Continuous Gradients)** Local cost functions $\mathbb{E}[F(\tilde{x}, \xi_i)]$ and regularization term $f_0(\tilde{x})$ are differentiable and have Lipschitz continuous gradients with constants $L_i$ and $L_0$, respectively.

**Assumption 3. (Bounded Variance)** Every worker $i \in \mathcal{V}$ samples i.i.d. data across time with random variables $\xi_i^k \sim D_i$. The variance of $\nabla F(\tilde{x}, \xi_i^k)$ is upper bounded by $\delta_i^2$, i.e., $\mathbb{E}[\|
abla F(\tilde{x}, \xi_i^k) - \nabla F(\tilde{x}, \xi_i^0)\|^2] \leq \delta_i^2$, for all $i$.

To attain a reasonable performance bound, it is necessary to assume that the network of regular agents is bidirectionally connected [19]. For instance, if a regular agent is surrounded by Byzantine neighbors, it is unable to communicate with any regular agents. Therefore, the best model it can learn is solely based on its local data. The idea of our analysis follows that in [20], which considers the TV norm-penalized problem $\min \{\frac{\lambda^2}{\eta^2} - 1, (k_0 + 1)\mathbb{E}[\|x^{k_0} - x^*\|^2] + \frac{\pi^2}{k_0 + 1}\}$.

Theorem 1. Suppose that Assumptions 1 and 2 hold true. If $\lambda \geq \lambda_0 := \max_{i \in \mathcal{R}} \mathbb{E}[\|
abla F(\tilde{x}^*, \xi_i) + \nabla f_0(\tilde{x}^*)\|_\infty]$, then for the optimal solution $x^*$ of (3) and the optimal solution $\tilde{x}^*$ of (1), we have $x^* = [\tilde{x}^*]$.

No matter how large $\lambda$ is, with a proper step size the proposed stochastic gradient method can converge to the optimal solution of (3) when the Byzantine agents are absent. However, the Byzantine agents bring disturbance to the optimization process, and their influence is illustrated in the second theorem.

**Theorem 2.** Suppose that Assumptions 1–4 hold true. Set the step size of our proposed method as $\alpha^k = \min\{\alpha, \frac{\pi}{k + \pi}\}$, where $\alpha = \min\{\min_{i \in \mathcal{R}} \frac{1}{4\sigma^2} \eta^2, \frac{1}{4\sigma^2} \eta^2\}$, and $\pi > \frac{1}{\eta}$ with $\eta = \min\{\frac{2u_iL_i}{\eta^2 u_i + L_i}, \frac{2u_0L_0}{\eta^2 u_0 + L_0}\}$. Then, there exists a smallest integer $k_0$ satisfying $\alpha \geq \frac{\pi}{k_0 + \pi}$, such that

$$\mathbb{E}[\|x^{k+1} - x^*\|^2] \leq (1 - \eta\alpha)^k \|x^0 - x^*\|^2 + \frac{1}{\eta} (\alpha\Delta_0 + \Delta_2), \quad \forall k < k_0,$$

and

$$\mathbb{E}[\|x^{k+1} - x^*\|^2] \leq \frac{\Delta_1}{k + 1} + \pi\Delta_2, \quad \forall k \geq k_0.$$

Here we define

$$\Delta_1 = \max\{\frac{\pi^2\Delta_0}{\eta^2}, (k_0 + 1)\mathbb{E}[\|x^{k_0} - x^*\|^2] + \frac{\pi^2\Delta_0}{k_0 + 1}\},$$

$$\Delta_0 = \sum_{i \in \mathcal{B}} (4\lambda^2 \mathbb{E}[\|B_i^2p + 4\lambda^2 |B_i|^2p + 2\delta_i^2\|]) \Delta_2 = \sum_{i \in \mathcal{B}} \lambda^2 |B_i|^2 \frac{\rho}{\epsilon}.$$

Theorem 2 asserts that our proposed algorithm can converge to a neighborhood of the optimal solution $x^*$ of (3). At the first stage, the convergence rate is linear. At the second stage, the convergence rate is sublinear, and the size of neighborhood is proportional to $p$ (the dimension of model), $\lambda^2$ (squared penalty parameter), and $\sum_{i \in \mathcal{B}} |B_i|^2$ that is determined by the number of Byzantine agents and the network topology. Combining Theorems 1 and 2, we derive the main Theorem as follows.

**Theorem 3.** Under the same conditions of Theorem 2, if choosing $\lambda \geq \lambda_0$, then for a sufficiently large $k \geq k_0$, we have

$$\mathbb{E}[\|x^{k+1} - [\tilde{x}^*]\|^2] \leq \frac{\Delta_1}{k + 1} + \pi\Delta_2.$$

If choosing $0 < \lambda < \lambda_0$ and supposing that the difference between the optimizers of (3) and (1) is bounded by $\|x^* - [\tilde{x}^*]\|^2 \leq \Delta_3$, then for a sufficiently large $k \geq k_0$, we have

$$\mathbb{E}[\|x^{k+1} - [\tilde{x}^*]\|^2] \leq \frac{2\Delta_1}{k + 1} + 2\pi\Delta_2 + 2\Delta_3.$$

When $\lambda$ is large enough, according to Theorem 1, (3) is equivalent to (1). Therefore, the gap between $x^k$ and $x^*$ directly translates to the gap between $x^k$ and $[\tilde{x}^*]$ as in (8). However, if $\lambda$ is too large, the gap will also be large because $\Delta_2$ is proportional to $\lambda^2$. When $\lambda$ is small, (3) cannot guarantee to have a consensus solution. In this case, the gap between $[\tilde{x}^*]$ and $x^*$ is unclear, but we assume that it is bounded by $\Delta_1$. Therefore, we are also able to characterize the gap between $x^k$ and $[\tilde{x}^*]$ as in (9).

4. NUMERICAL EXPERIMENTS

In this section, we conduct a set of numerical experiments to demonstrate the robustness of our proposed method to Byzantine attacks. The benchmark methods are DPSGD [2], as well as the stochastic versions of ByRDiE [13] and BRIDGE [14] (denoted by ByRDiE-S and BRIDGE-S, respectively). In DPSGD, the mixing matrix is set following the equal neighbor weights rule [21]. In ByRDiE-S, the coordinates of the model are updated sequentially, and the number of inner-loop iterations to update every coordinate is set to be 1, as suggested by [13]. For fair comparison, in ByRDiE-S one iteration refers to that all the coordinates have been updated once. Step sizes of the benchmark methods are hand-tuned to the best.

Consider a Erdos-Renyi graph of $n = 30$ agents. We randomly choose $b$ agents to be Byzantine, but guarantee that the network of regular agents is connected. The data set is MNIST, which contains 10 handwritten digits from 0 to 9, with 60,000 training images and 10,000 testing images. In the i.i.d. case, we randomly and evenly distribute the training images to all the agents. In the non-i.i.d. case, we let every three agents evenly split the training images.
of one digit. We use softmax regression with regularization term
\( f_0(\tilde{x}) = \frac{0.01}{2} \|\tilde{x}\|^2 \) to learn the model. At the testing stage, we
randomly choose one regular agent and use its local model to calculate
classification accuracy. Also, we calculate the variance of regular agents’ local models to quantify the level of consensus.

As shown in Fig. 3, the results are consistent with those under
the same-value attacks, but the performance gain of our proposed
method in terms of classification accuracy is more obvious. Note
that we choose a relatively small \( \lambda \) such that consensus of regular
agents is slightly worse than those of ByRDiE and BRIDGE.

**Impact of Penalty Parameter \( \lambda \).** To investigate the impact of penalty
parameter \( \lambda \), we choose several different values for \( \lambda \) in the setting
of same-value attacks with \( b = 3 \) Byzantine agents. The step sizes are
hand-tuned to the best. As shown in Fig. 4, larger \( \lambda \) ensures better consensus, which corroborates the theoretical results in Section 3. When \( \lambda = 0 \), the level of consensus is the worst, since the
agents do not communicate and learn with their own local data samples independently. However, larger \( \lambda \) leads to larger gap relative to the
Byzantine-free optimal solution, and hence lower classification
accuracy. This observation also matches the results in Section 3.

**Non-i.i.d. Data.** Let the number of Byzantine agents be \( b = 6 \).
All the Byzantine agents copy the values of one randomly chosen
regular agent, and send to their neighbors. Recall that every three
agents evenly split the training images of one digit and here we de-
liberately let the Byzantine agents share the training images of dig-
its 8 and 9. Therefore, information from digits 8 and 9 totally lose
and the best classification accuracy we can reach is no more than
0.8. Note that under these particularly designed attacks, DPSGD
is able to reach a satisfactory classification accuracy. In our pro-
posed method, the penalty parameter is \( \lambda = 0.02 \) and the step size
\( \alpha_k = 0.4/\sqrt{k + 1} \). As shown in Fig. 5, our proposed method almost
coincides with DPSGD with respect to classification accuracy.
ByRDiE-S and BRIDGE-S do not perform well under such attacks,because nine agents (including six Byzantine agents and three regular
agents) essentially use the training images of one digit, such that
the models trained from this particular digit dominate. Therefore,
the majority voting rule of ByRDiE-S and BRIDGE-S emphasizes
more on this particular digit, while ignores other digits relatively.

**Acknowledgement.** Qing Ling is supported in part by NSF China
Grants 61573331 and 61973324, and Fundamental Research Funds
for the Central Universities.
5. REFERENCES

[1] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi, “Decentralized stochastic optimization and gossip algorithms with compressed communication,” In: Proceedings of ICML, 2019.

[2] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu, “Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent,” In: Proceedings of NeurIPS, 2017.

[3] Srinivasan S. Ram, Angelia Nedic, and Venugopal V. Veeravalli, “Distributed stochastic subgradient projection algorithms for convex optimization,” Journal of Optimization Theory and Application, vol. 147, no. 3, pp. 516–545, 2010.

[4] John Duchi, Alekh Agarwal, and Martin Wainwright, “Dual averaging for distributed optimization: Convergence analysis and network scaling,” IEEE Transactions on Automatic Control, vol. 57, no. 3, pp. 592–606, 2012.

[5] Michael Rabbat, “Multi-agent mirror descent for decentralized stochastic optimization,” In: Proceedings of CAMSAP, 2015.

[6] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu, “Asynchronous decentralized parallel stochastic gradient descent,” In: Proceedings of ICML, 2018.

[7] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié, “Asynchronous accelerated proximal stochastic gradient for strongly convex distributed finite sums,” arXiv preprint arXiv:1901.09865, 2019.

[8] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu, “D2: Decentralized training over decentralized data,” In: Proceedings of ICML, 2018.

[9] Kun Yuan, Bicheng Ying, Jiageng Liu, and Ali H. Sayed, “Variance-reduced stochastic learning by networked agents under random reshuffling,” IEEE Transactions on Signal Processing, vol. 67, no. 2, pp. 351–366, 2018.

[10] Aryan Mokhtari and Alejandro Ribeiro, “DSA: Decentralized double stochastic averaging gradient algorithm,” Journal of Machine Learning Research, vol. 17, no. 1, pp. 2165–2199, 2016.

[11] Qinyi Luo, Jinkun Lin, Youwei Zhuo, and Xuehai Qian, “Hop: Heterogeneity-aware decentralized training,” In: Proceedings of ASPLOS, 2019

[12] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease, “The Byzantine generals problem,” ACM Transactions on Programming Languages and Systems, vol. 4, no. 3, pp. 382–401, 1982.

[13] Zhixiong Yang and Waheed U. Bajwa, “ByRdiE: Byzantine-resilient distributed coordinate descent for decentralized learning,” arXiv preprint arXiv:1708.08155, 2017.

[14] Zhixiong Yang and Waheed U. Bajwa, “BRIDGE: Byzantine-resilient decentralized gradient descent,” arXiv preprint arXiv:1908.08098, 2019.

[15] Zhixiong Yang, Arpita Gang, and Waheed U. Bajwa, “Adversary-resilient inference and machine learning: From distributed to decentralized,” arXiv preprint arXiv:1908.08649, 2019.

[16] Yudong Chen, Lili Su, and Jiaming Xu, “Distributed statistical machine learning in adversarial settings: Byzantine gradient descent,” In: Proceedings of SIGMETRICS, 2019.

[17] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta, “Generalized Byzantine-tolerant SGD,” arXiv preprint arXiv:1802.10116, 2018.

[18] Walid Ben-Ameur, Pascal Bianchi, and Jeremie Jakubowicz, “Robust distributed consensus using total variation,” IEEE Transactions on Automatic Control, vol. 61, no. 6, pp. 1550–1564, 2016.

[19] Wei Xu, Zhengqing Li, and Qing Ling, “Robust decentralized dynamic optimization at presence of malfunctioning agents,” Signal Processing, vol. 153, pp. 24–33, 2018.

[20] Liping Li, Wei Xu, Tianyi Chen, Georgios B. Giannakis, and Qing Ling, “RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets,” In: Proceedings of AAAI, 2019.

[21] Vincent D. Blondel, Julien M. Hendrickx, Alex Olshevsky, and John N. Tsitsiklis, “Convergence in multiagent coordination, consensus, and flocking,” In: Proceedings of CDC, 2006.