Byzantine-Resilient Stochastic Gradient Descent for Distributed Learning: A Lipschitz-Inspired Coordinate-wise Median Approach

Haibo Yang, Xin Zhang, Minghong Fang, and Jia Liu

***Abstract*— In this work, we consider the resilience of distributed algorithms based on stochastic gradient descent (SGD) in distributed learning with potentially Byzantine attackers, who could send arbitrary information to the parameter server to disrupt the training process. Toward this end, we propose a new Lipschitz-inspired coordinate-wise median approach (LICM-SGD) to mitigate Byzantine attacks. We show that our LICM-SGD algorithm can resist up to half of the workers being Byzantine attackers, while still converging almost surely to a stationary region in non-convex settings. Also, our LICM-SGD method does not require any information about the number of attackers and the Lipschitz constant, which makes it attractive for practical implementations. Moreover, our LICM-SGD method enjoys the optimal $O(md)$ computational time-complexity in the sense that the time-complexity is the same as that of the standard SGD under no attacks. We conduct extensive experiments to show that our LICM-SGD algorithm consistently outperforms existing methods in training multi-class logistic regression and convolutional neural networks with MNIST and CIFAR-10 datasets. In our experiments, LICM-SGD also achieves a much faster running time thanks to its low computational time-complexity.**

I. Introduction

Fueled by the rise of machine learning and big data analytics, recent years have witnessed an ever-increasing interest in solving large-scale empirical risk minimization problems (ERM) – a fundamental optimization problem that underpins a wide range of machine learning applications. In the post-Moore's-Law era, however, to sustain the rapidly growing computational power needs for solving large-scale ERM, the only viable solution is to exploit *parallelism* at and across different spatial scales. Indeed, the recent success of machine learning applications is due in large part to the use of distributed machine learning frameworks (e.g., TensorFlow [1], MXNet [2], Cognitive ToolKit (CNTK) [3], Caffe [4], etc.), which exploit the abundance of distributed CPU/GPU resources in large-scale computing clusters. Furthermore, in many large-scale learning systems, data are sampled and stored at different geo-locations. As a result, it is often infeasible to move all the data to a centralized location because of prohibitively high costs or privacy concerns. Due to these factors, first-order stochastic gradient descent (SGD) based methods have been the workhorse algorithms in most distributed machine learning frameworks thanks to their low complexity and simplicity in distributed implementations.

Haibo Yang, Minghong Fang, and Jia Liu are with the Department of Computer Science, Iowa State University, Ames, IA 50011, USA yanghb@iastate.edu, myfang@iastate.edu, jialiu@iastate.edu

Xin Zhang is with the Department of Statistics, Iowa State University, Ames, IA 50011, USA xinzhang@iastate.edu

Unfortunately, the proliferation of distributed learning systems also introduces many new cybersecurity challenges in the design of SGD-based distributed optimization algorithms. Besides the conventional computation/communication errors or stalled processes seen in traditional distributed computing systems, a serious problem in SGD-based distributed optimization methods is that they are prone to the so-called Byzantine attacks, where a malicious worker machine returns arbitrary information to the parameter server. It has been shown in [5] that, under Byzantine attacks, even a single erroneous gradient can fail the whole learning system and causing the classical distributed SGD algorithm to diverge. In light of the vulnerability of the traditional SGD-based optimization algorithms, there have been strong interests in designing robust SGD-type algorithms that are resilient to Byzantine attacks in distributed learning.

However, developing Byzantine-resilient optimization algorithms for distributed learning is highly non-trivial. Despite a significant amount of efforts spent over the years, most existing work in the literature on Byzantine-resilient distributed algorithms (see, e.g., [5]–[11]) have two main limitations: i) requiring the knowledge of the number of malicious workers, which is often infeasible in practice; and ii) suffering from high computational complexity in stochastic subgradient screening and aggregation mechanisms (see Section II for detailed discussions). Moreover, the classification accuracy performance of these existing works are far from satisfactory in practice (see our numerical experiments in Section V). The limitations of these existing works motivate us to develop a new Byzantine-resilient SGD-based optimization algorithm for distributed learning.

The main contribution of this paper is that we propose a new Byzantine-resilient SGD-based optimization algorithm based on a low-complexity Lipschitz-inspired coordinate-wise median approach (LICM-SGD), which overcomes the aforementioned limitations in mitigating Byzantine attacks. Our main technical results are summarized as follows:

- Inspired by the rationale that "benign workers should generate stochastic gradients closely following the Lipschitz characteristics of the true gradients," we develop a Lipschitz-inspired coordinate-wise stochastic gradient screening and aggregation mechanism. We show that our proposed LICM-SGD can resist up to one-half of the workers being Byzantine, and yet still achieve the same convergence performance compared to the no-attack scenario. We note that these nice performance gains under LICM-SGD are achieved *without* requiring any knowledge of the number of Byzantine attackers, which is assumed in

most existing works (see, e.g., [5]–[7], [9]–[12]). Hence, our proposed LICM-SGD is more advantageous for practical implementations.

- Another salient feature of our proposed LICM-SGD approach is that it has low computational complexity in stochastic gradient screening and aggregation. Specifically, our LICM-SGD method only requires $O(md)$ time-complexity, where m is the number of worker machines and d is the dimensionality of the ERM problem. We note that this time-complexity is *optimal* in the sense that it is the same as the most basic distributed SGD algorithm, which has the lowest computational time-complexity. In contrast, most of the existing algorithms with Byzantine-resilience performance similar to ours (i.e., being able to resist up to one half of workers being Byzantine) have time complexity $O(m^2d)$, which is problematic in data centers where the number of workers is typically large.
- Last but not least, to verify the real-world performance of our proposed LICM-SGD approach, we conduct extensive experiments by training multi-class regression (MLR) and convolutional neural networks (CNN) based on the MNIST and CIFAR-10 datasets. Our experimental results show that the classification accuracy under LICM-SGD is only 5% lower than the standard distributed SGD method with no attacks, and consistently outperforms the state of the art in the literature. For MLR and CNN training on the MNIST datasets, LICM-SGD can reach up to 85% classification accuracy, which is three times as high as other coordinate-wise median-based methods. These good numerical results corroborate our theoretical results.

Collectively, our results advance the design of Byzantine-resilient SGD-type optimization algorithms for distributed machine learning. The remainder of this paper is organized as follows: In Section II, we review the literature to put our work in comparative perspectives. Section III introduces the system and problem formulation. Section IV focuses on the LICM-SGD algorithmic design and performance analysis. Section V presents numerical results and Section VI concludes this paper.

II. Related Work

Although the SGD algorithm traces its roots to the seminal work by Robbins and Monro in early 50s [13], the design of Byzantine-resilient SGD methods for distributed learning remains relatively under-explored. Generally speaking, the basic idea behind most Byzantine-resilient SGD methods is to take the median of a batch of stochastic gradients, which is statistically more stable than taking the average as in the basic SGD method. Under this basic idea, existing works can be roughly classified into two main categories as follows.

The first category is based on the geometric median. Specifically, this approach aims to find a point in the parameter vector space that minimizes the sum distance to the current batch of stochastic gradients in some ℓ_p norm sense. Two notable approaches in this category are [8] and [5], both of which can be shown to converge with up to half of the workers being Byzantine. We note that these methods have the same Byzantine resilience performance compared to ours. However, the computational complexity of these approaches is $O(m^2d)$, where m is the number of workers and d is the dimensionality of the problem. We note that this computational complexity is significantly higher than our $O(md)$ result and could be problematic in data center settings, where the number of machines is on the order of thousands or even higher. Further, it has been recently shown that Byzantine attackers can utilize the high dimensionality and the highly non-convex landscape of the loss function in ERM to make geometric-median-based methods ineffective [7]. The reason is that since geometric median minimizes the sum distance of all dimensions, it may not be able to discriminate how much two stochastic gradients disagree in each dimension. To address this problem, the authors of [7] proposed a hybrid strategy: the first step is to recursively use a geometric-median-based method to pick $m-2q$ gradients, and then utilize a coordinate-wise trimmed mean approach (to be discussed shortly) to determine the update direction. However, their computational time-complexity remains $O(m^2d)$ and the maximum number of Byzantine attackers cannot exceed $1/4$ of total workers.

The second category is based on using the coordinate-wise median (or related methods) as an aggregation rule for the stochastic gradient updates. Specifically, coordinate-wise median methods simply take the median in each dimension rather than the entire vector as in geometric median. For example, in [6], the authors gave a sharp analysis on the statistical error rates for two Byzantine-resilient algorithms, namely coordinate-wise median and trimmed mean, respectively. The convergence of these existing coordinate median based methods can tolerate up to half of the workers being Byzantine, and with computational time-complexity $O(md)$ (same as ours). However, in practice, the classification accuracy performance of these coordinate-wise median based methods are usually worse than those of the geometric median based methods. We note that our LICM-SGD method also falls into the category of coordinate-wise median methods. However, our work differs from the existing coordinate median approaches in that we develop a Lipschitz-inspired selection rule for screening stochastic gradients. Our LICM-SGD method not only retains the low $O(md)$ time-complexity, but also achieves a classification accuracy significantly higher than the existing work.

We note that there are other related works that consider different settings under Byzantine attacks, which are not directly comparable to our work algorithmically. For example, the authors in [12] considered asynchronous SGD, while the authors in [14] used historical information to remove suspicious workers from future considerations. In [15], the authors proposed to let the parameter server keep a small set of data to compute an estimate of the true gradient, which is used as a benchmark to filter out suspicious gradients. In [10], the authors developed a suspicion-based aggregation rule, but with a weaker attack model that the Byzantine attackers have no information about the loss function.

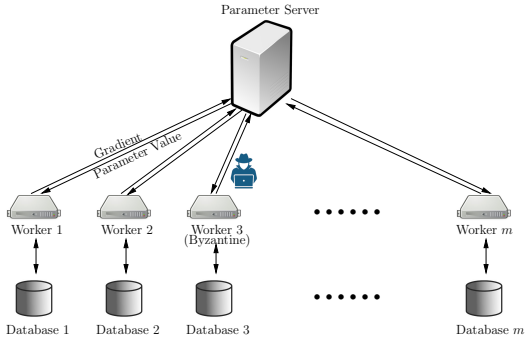

Fig. 1. A distributed learning system with Byzantine attacks.

III. System Model and Problem Statement

Notation: In this paper, we use boldface to denote matrices/vectors. We let $[\mathbf{v}]_i$ represent the i-th entry of $\mathbf{v}$. We use $\|\cdot\|$ to denote the ℓ^2-norm.

In this paper, we consider a distributed machine learning system with potential adversarial attacks. As shown in Fig. 1, there are one parameter server (PS) and m distributed worker machines in the system. Some of the workers could be malicious and launch Byzantine attacks. We consider a standard empirical risk minimization (ERM) setting, where the goal is to find an optimal parameter vector $\mathbf{w}^*$ that minimizes the risk function $F(\mathbf{w})$, i.e.,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) \triangleq \frac{1}{N}\sum_{j=1}^{N} f(\mathbf{w},\xi_j),$$

where $f:\mathbb{R}^d\to\mathbb{R}$ represents a loss function of some learning models (e.g., regression, deep neural networks, etc.) and $\{\xi_j\}_{j=1}^N$ denotes the set of random samples with size N. In this system, the PS and workers employ a stochastic gradient descent (SGD) based optimization algorithm to solve the ERM problem in a distributed fashion as follows: In iteration k, each worker i first retrieves the current parameter value $\mathbf{w}_k$ from the PS. It then draws a mini-batch of independently and identically distributed (i.i.d.) random data samples from its local database to compute a stochastic gradient $g_i(\mathbf{w}_k)\in\mathbb{R}^d$ of the loss function and return $g_i(\mathbf{w}_k)$ to the PS. Upon the reception of all m stochastic gradients from the workers, the PS updates the parameter vector as follows:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta_k G[g_i(\mathbf{w}_k),\ldots,g_m(\mathbf{w}_k)], \qquad (1)$$

where η_k is the step-size in iteration k and $G[\cdot]$ denotes some aggregation rule. As an example, for the standard SGD in normal systems without attacks, G corresponds to taking the average of the stochastic gradients.

In Byzantine attacks, malicious workers have the full knowledge of the system and they can collaborate with each other. Each Byzantine worker could return arbitrary values to the PS. Hence, the Byzantine attack model is viewed as the most difficult class of attacks to defend in distributed systems. In this paper, we aim to *develop a robust aggregation function $G(\cdot)$ (instead of taking the average) to mitigate Byzantine attacks*. Toward this end, we make the following assumptions:

Assumption 1 (Unbiased Gradient Estimator): The stochastic gradient $g_i(\mathbf{w}_k)$ returned by a correct (non-Byzantine) worker i is an unbiased estimator of the true gradient of $F(\cdot)$ evaluated at $\mathbf{w}_k$, i.e., $\mathbb{E}g_i(\mathbf{w}_k)=\nabla F(\mathbf{w}_k)$, $i\in\{1,\ldots,m\}$.

Assumption 2 (Bounded Variance): The stochastic gradient $g_i(\mathbf{w}_k)$ from a correct (non-Byzantine) worker $i\in\{1,\ldots,m\}$ has a bounded variance, i.e., $\mathbb{E}\big|[g_i(\mathbf{w}_k)]_j - [\nabla F(\mathbf{w}_k)]_j\big|^2 \le \sigma^2$, $\forall j=1,\ldots,d$, where $\sigma^2>0$ is a constant.

Assumption 3 (Lipschitz Continuous Gradient): There exists a constant $L>0$ such that for all $\mathbf{w}_1,\mathbf{w}_2\in\mathbb{R}^d$, it holds that $\|\nabla F(\mathbf{w}_1)-\nabla F(\mathbf{w}_2)\|\le L\|\mathbf{w}_1-\mathbf{w}_2\|$.

Assumption 4 (Step-size): The step-sizes $\{\eta_k\}$ in (1) satisfy $\sum_{k=1}^{\infty}\eta_k=\infty$ and $\sum_{k=1}^{\infty}\eta_k^2<\infty$.

Assumption 5 (Differentiability): The objective function $F(\cdot)$ is three times differentiable with continuous derivatives.

Assumption 6 (Linear Growth of the r-th Moment): For a stochastic gradient $g_i(\mathbf{w}_k)$ returned by a correct (non-Byzantine) worker i, there exist positive constants $C_r, D_r>0$, $r=2,3,4$, such that $\mathbb{E}\|g_i(\mathbf{w}_k)\|^r \le C_r + D_r\|\mathbf{w}_k\|^r, \forall \mathbf{w}_k\in\mathbb{R}^d$.

Several remarks for Assumptions 1–6 are in order: Assumptions 1–4 are standard in the SGD convergence analysis literature. Assumption 5 can usually be satisfied by most learning problems in practice. Assumption 6 implies that the r-th moment of a non-Byzantine stochastic gradient does not grow faster than linearly with respect to the norm of $\mathbf{w}_k$, which is a necessary condition in order to bound the distribution tails for convergence (cf. [16]). We note that Assumption 6 is not restrictive and has appeared in non-convex stochastic approximations (e.g., [16]) as well as several recent Byzantine tolerant gradient descent algorithms [5] [12]. Note that we do not assume any convexity property about $F(\cdot)$, i.e., $F(\cdot)$ could potentially be non-convex.

With the above modeling and assumptions, we are now in a position to present our Byzantine-resilient SGD method, which constitutes the main subject in the next section.

IV. A Lipschitz-Inspired Coordinate-wise Median Approach to Mitigate Byzantine Attacks

In this section, we will first introduce our LICM-SGD algorithm in Section IV-A. Then, we will present the main theoretical results and their intuitions in Section IV-B. The proofs for the main results are provided in Section IV-C.

A. The LICM-SGD Algorithm

Before presenting our LICM-SGD algorithm, we first formally define the notion of coordinate-wise median, which serves as a key building block for our aggregation rule.

Definition 1 (Coordinate-wise Median): For a set of vectors $\mathbf{v}_i\in\mathbb{R}^d, i=1,\ldots,m$, the coordinate-wise median, denoted as $\texttt{CoorMed}\{\mathbf{v}_i, i=1,\ldots,m\}$, is a vector with its j-th coordinate being $\texttt{Med}\{[\mathbf{v}_i]_j, i=1,\ldots,m\}, j=1,\ldots,d$, where $\texttt{Med}\{\cdot\}$ denotes the median of a set of scalars.

Our proposed LICM-SGD algorithm is stated in Algorithm 1 as follows:

Algorithm 1: The LICM-SGD Algorithm.

Initialization:

1. Let $k = 0$. Choose an initial parameter vector $\mathbf{w}_0$ and an initial step-size η_0.

Main Loop:

2. In the k-th iteration, each worker i retrieves the current parameter vector $\mathbf{w}_k$ from the PS, $\forall i = 1, \ldots, m$.
3. For each worker $i \in \{1, \ldots, m\}$, if it is non-Byzantine, it computes a stochastic gradient $g_i(\mathbf{w}_k)$ based on a mini-batch and returns $g_i(\mathbf{w}_k)$ to the PS; otherwise, it returns an arbitrary value to the PS.
4. Upon receiving stochastic gradients from all m workers, the PS computes the coordinate-wise median $\mathbf{u}_k$:

$$\mathbf{u}_k = \texttt{CoorMed}\{g_i(\mathbf{w}_k), i = 1, \ldots, m\}. \quad (2)$$

Then, the PS selects all stochastic gradient vectors $g_i(\mathbf{w}_k)$, $i \in \{1, \ldots, m\}$, that satisfy:

$$\big|[g_i(\mathbf{w}_k)]_j - [\mathbf{u}_{k-1}]_j\big| \leq \gamma \big|[\mathbf{u}_k]_j - [\mathbf{u}_{k-1}]_j\big|, \quad (3)$$

$\forall j \in \{1, \ldots, d\}$, to form a set $\mathcal{T}_k^*$, where $\gamma \geq 1$ is some system parameter. If $k = 0$, let $\tilde{g}(\mathbf{w}_0) = \mathbf{u}_0$. Otherwise, let

$$\tilde{g}(\mathbf{w}_k) = \frac{1}{|\mathcal{T}_k^*|} \sum_{i \in \mathcal{T}_k^*} g_i(\mathbf{w}_k). \quad (4)$$

5. Update the parameter vector as:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta_k \tilde{g}(\mathbf{w}_k). \quad (5)$$

6. Let $k \leftarrow k + 1$ and go to Step 2.

Remark 1: Several important remarks on Algorithm 1 are in order: i) Algorithm 1 is inspired by the intuition that "benign workers should generate stochastic gradients closely following the Lipschitz characteristics of the true gradients." To see this, note that if the $\texttt{CoorMed}\{\cdot\}$ operation removes the outliers and so $\mathbf{u}_k$ is "close" to the average (i.e., an unbiased estimator of $\nabla F(\cdot)$), then we approximately have:

$$\frac{\big|[\mathbf{u}_k]_j - [\mathbf{u}_{k-1}]_j\big|}{\big|[\mathbf{w}_k]_j - [\mathbf{w}_{k-1}]_j\big|} \approx \frac{\big|[\nabla F(\mathbf{w}_k)]_j - [\nabla F(\mathbf{w}_{k-1})]_j\big|}{\big|[\mathbf{w}_k]_j - [\mathbf{w}_{k-1}]_j\big|} \leq L,$$

where the last inequality follows from the Lipschitz gradient assumption (cf. Assumption 3). Hence, if a stochastic gradient $g_i(\mathbf{w}_k)$ satisfies the following relationship:

$$\begin{aligned}\frac{|[g_i(\mathbf{w}_k)]_j - [\mathbf{u}_{k-1}]_j|}{|[\mathbf{w}_k]_j - [\mathbf{w}_{k-1}]_j|} &\leq \frac{\gamma|[\mathbf{u}_k]_j - [\mathbf{u}_{k-1}]_j|}{|[\mathbf{w}_k]_j - [\mathbf{w}_{k-1}]_j|} \\ &\approx \frac{\gamma|[\nabla F(\mathbf{w}_k)]_j - [\nabla F(\mathbf{w}_{k-1})]_j|}{|[\mathbf{w}_k]_j - [\mathbf{w}_{k-1}]_j|} \lessapprox \gamma L, \quad \forall j, \quad (6)\end{aligned}$$

for some appropriately chosen parameter $\gamma \geq 1$, then it is likely that worker i is a benign worker. Lastly, extracting the numerator relationship from (6) yields:

$$\big|[g_i(\mathbf{w}_k)]_j - [\mathbf{u}_{k-1}]_j\big| \leq \gamma \big|[\mathbf{u}_k]_j - [\mathbf{u}_{k-1}]_j\big|,$$

which is the stochastic gradient selection rule (3) in Algorithm 1. It is also worth pointing out that even though Eq. (3) is inspired by the Lipschitz characteristics of $\nabla F(\cdot)$, we do *not* need to know the value of the Lipschitz constant L. ii) The parameter $\gamma \geq 1$ is used to balance the trade-off between Byzantine-resilience and variance. This is because, on one hand, increasing γ helps increase the size of $\mathcal{T}_k^*$, leading to smaller variance (if all included stochastic gradients are benign) but at a higher risk to include Byzantine workers. On the other hand, decreasing γ could lead to too few stochastic gradients being eligible, which results in a larger variance in the final stochastic gradient computation (4). In practical implementations, γ can either be preset or data-driven (i.e., adaptively choosing γ based on the sequentially arriving random samples). iii) We note that, unlike most existing methods in the literature, Algorithm 1 does *not* require the knowledge of the number of Byzantine workers, which is usually difficult to estimate in practice. Hence, our proposed LICM-SGD method is more advantageous for practical implementations.

B. Main Theoretical Results

For better readability, we summarize the main theoretical results in this subsection and relegate their proofs to Section IV-C. Our first key result in this paper suggests that the coordinate-wise median of a set of stochastic gradients under Byzantine attacks shares the *same* statistical characteristics as that of a non-Byzantine stochastic gradient if the number of Byzantine workers is less than half of the total.

Lemma 1 (Statistical Properties of $\texttt{CoorMed}(\cdot)$*):* Let q be the number of Byzantine workers and if $2q + 1 < m$, then $\mathbf{u}_k = \texttt{CoorMed}\{g_i(\mathbf{w}_k), i = 1, \ldots, m\}$ share the same statistical characteristics as a stochastic gradient returned by a non-Byzantine worker:

$$\begin{aligned}&\mathbb{E}\big|[\mathbf{u}_k]_j - [\nabla F(\mathbf{w}_k)]_j\big|^2 \leq \sigma^2, \quad j = 1, \ldots, d, \\ &\mathbb{E}\|\mathbf{u}_k\|^r \leq C_r + D_r \|\mathbf{w}_k\|^r, \quad \forall \mathbf{w}_k \in \mathbb{R}^d, r = 2, 3, 4.\end{aligned}$$

Lemma 1 indicates that, statistically, $\mathbf{u}_k$ can be viewed as a non-Byzantine stochastic gradient if $2q+1 < m$. This insight is our fundamental rationale to use $\mathbf{u}_k$ as a benchmark for our aggregation criterion in Algorithm 1. Thanks to this nice statistical property of coordinate-wise median, we can further prove that, based on the screening rule in (3), the aggregated vector $\tilde{g}(\mathbf{w}_k)$ obtained from (4) has linear growths of the r-th moments, $r = 2, 3, 4$. We formally stated this result in the following lemma:

Lemma 2 (Linear Growth of r*-th Moment):* There exists constants $A_r, B_r > 0$, $r = 2, 3, 4$, such that $\forall k \geq 0, \mathbb{E}\|\tilde{g}(\mathbf{w}_k)\|^r \leq A_r + B_r \|\mathbf{w}_k\|^r$ if the step-sizes η_k, $\forall k$, satisfy $\eta_k \leq \min\{h_r(A_r, B_r, C_r, D_r), r = 2, 3, 4\}$, where the functions $h_r(\cdot)$, $r = 2, 3, 4$, will be specified in the proof.

Lemma 2 indicates that, under the aggregation rule in Algorithm 1, the obtained $\tilde{g}(\mathbf{w}_k)$ remains satisfying the linear growth of the r-th moment assumption (cf. Assumption 6). As will be discussed next, the boundedness of the moments implies the global confinement property of the weight parameter vector $\mathbf{w}_k$, which in turn will play an important role in establishing convergence of SGD-type

methods in non-convex optimization. We formally state the global confinement property as follows:

Lemma 3 (Global Confinement of $\{\mathbf{w}_k\}$): Let $\{\mathbf{w}_k\}$ be the sequence of weight parameter vectors generated by Algorithm 1. There exist a constant $D > 0$ such that the sequence $\{\mathbf{w}_k\}$ almost surely satisfies $\|\mathbf{w}_k\| \leq D$ as $k \to \infty$.

In what follows, we will characterize the Byzantine-resilience performance of our proposed LICM-SGD approach. Toward this end, we first introduce a useful performance metric for measuring Byzantine resilience:

Definition 2 ((α, q)-Byzantine Resilience [5]): Let $\alpha \in [0, \pi/2]$ and let $q \in [0, m]$ be an integer. Let $\mathbf{w}_1, \ldots, \mathbf{w}_m$ be i.i.d. random vectors distributed as $\mathbf{w} \in \mathbb{R}^d$ with $\mathbb{E}\mathbf{w} = \mathbf{g}$, where $\mathbf{g} \in \mathbb{R}^d$ is deterministic. Let $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_q \in \mathbb{R}^d$ be any random vectors possibly dependent on $\{\mathbf{w}_i\}$. An aggregation rule G is said to be (α, q)-Byzantine resilient, if for any $1 \leq i_1 < ... < i_q \leq m$, the aggregated vector

$$G(\mathbf{w}_1, \ldots, \underbrace{\mathbf{b}_1}_{i_1}, \ldots, \underbrace{\mathbf{b}_q}_{i_q}, \ldots, \mathbf{w}_m)$$

satisfies: i) $\langle \mathbb{E}G, \mathbf{g} \rangle \geq (1 - \sin\alpha)\|\mathbf{g}\|^2 > 0$, and ii) for $r = 2, 3, 4$, $\mathbb{E}\|G\|^r$ is bounded by a linear combination of terms $\mathbb{E}\|\mathbf{w}\|^{r_i}$, $i = 1, \ldots, n-1$, with $r_1 + ... + r_{n-1} = r$.

Geometrically speaking, Condition i) in Definition 2 means that, with q Byzantine attackers, the average of stochastic aggregation outcome of $G(\cdot)$ is contained in an error ball centered at the end point of $\mathbf{g}$, such that the angle between $\mathbb{E}G$ and $\mathbf{g}$ is bounded by α. Condition ii) is a technical condition that is useful in our subsequent convergence analysis. With Lemmas 1–2, and Definition 2, we are now in a position to characterize the Byzantine-resilience of our proposed LICM-SGD method as follows:

Theorem 1 (Byzantine Resilience): Under Assumptions 1–6, if i) the number of Byzantine worker q satisfies $2q + 1 < m$ (i.e., less than one half) and ii) $(3 + 2\gamma + \epsilon)\sqrt{d}\sigma_k < ||\nabla F(\mathbf{w}_k)||$, $\forall k$, where $\epsilon > 0$ can be arbitrarily small, then Algorithm 1 is $(\alpha, q)-$Byzantine resilient, where $\alpha \in [0, \pi/2]$ is defined by

$$\sin\alpha = \sup_{\forall k} \left\{ \frac{(3 + 2\gamma + \epsilon)\sqrt{d}\sigma_k}{\|\nabla F(\mathbf{w}_k)\|} \right\},$$

where σ_k denotes the deviation in iteration k.

Based on the Byzantine resilience result in Theorem 1, we establish the convergence of LICM-SGD as follows:

Theorem 2 (Almost Sure Convergence of LICM-SGD): Under Assumptions 1–6, the sequence of true gradients $\{\nabla F(\mathbf{w}_k)\}$ generated by Algorithm 1 converges almost surely to a flat region defined by $\left\{\mathbf{w} \in \mathbb{R}^d : \|\nabla F(\mathbf{w})\| \leq (3 + 2\gamma + \epsilon)\sqrt{d}\sigma_k\right\}$, where $\epsilon > 0$ can be arbitrarily small.

Finally, the time-complexity of the aggregation scheme in Algorithm 1 is stated in the following proposition:

Proposition 3 (Aggregation Time-Complexity): The aggregation rule (3)–(4) in Algorithm 1 has a computational time-complexity $O(md)$.

C. Proofs of the Main Results

In this subsection, we provide detailed proofs for all theoretical results stated in Section IV-B.

Proof of Lemma 1. If the number of Byzantine workers q satisfies $2q + 1 < m$, then there must at least exist two non-Byzantine workers p and q, such that

$$[g_p(\mathbf{w}_k)]_j \leq [\mathbf{u}_k]_j \leq [g_q(\mathbf{w}_k)]_j.$$

Since p and q are non-Byzantine, we have that $g_p(\mathbf{w}_k)$ and $g_q(\mathbf{w}_k)$ satisfy the assumptions of bounded variance and linear growth of r-th moment (cf. Assumption 2 and 6). Hence, it follows that $\mathbb{E}\|[\mathbf{u}_k]_j - [\nabla F(\mathbf{w}_k)]_j\|^2 \leq \sigma^2$, and $\mathbb{E}\|\mathbf{u}_k\|^r \leq C_r + D_r\|\mathbf{w}_k\|^r, \forall \mathbf{w}_k \in \mathbb{R}^d, r = 2, 3, 4$. This completes the proof of Lemma 1. □

Proof of Lemma 2. We prove Lemma 2 by induction. Let A_r and B_r, $r = 2, 3, 4$, be some constants such that $A_r > C_r$ and $B_r > D_r$, respectively. In the base case, $\tilde{g}(\mathbf{w}_0) = \mathbf{u}_0$, which is the coordinate-wise median of all stochastic gradients in the first iteration. As long as the Byzantine attackers are less than half of the total workers, Lemma 1 guarantees that $\mathbb{E}\|\tilde{g}(\mathbf{w}_0)\|^r \leq C_r + D_r\|\mathbf{w}_0\|^r \leq A_r + B_r\|\mathbf{w}_0\|^r, r = 2, 3, 4$.

Now, assume that in iteration k, we have $\mathbb{E}\|\tilde{g}(\mathbf{w}_k)\|^r \leq A_r + B_r\|\mathbf{w}_k\|^r$ holding true, we want to show that $\mathbb{E}\|\tilde{g}(\mathbf{w}_{k+1})\|^r \leq A_r + B_r\|\mathbf{w}_{k+1}\|^r$ continues to hold in iteration $k + 1$. Toward this end, note that

$$\begin{aligned}
\|\tilde{g}(\mathbf{w}_{k+1})\| &= \left\| \tfrac{1}{|\mathcal{T}_k^*|} \textstyle\sum_{i \in \mathcal{T}_k^*} g_i(\mathbf{w}_{k+1}) \right\| \\
&\overset{(a)}{\leq} \left\| \tfrac{1}{|\mathcal{T}_k^*|} \textstyle\sum_{i \in \mathcal{T}_k^*} g_i(\mathbf{w}_{k+1}) - \mathbf{u}_k \right\| + \|\mathbf{u}_k - \mathbf{u}_{k+1}\| + \|\mathbf{u}_{k+1}\| \\
&\overset{(b)}{\leq} (1 + \gamma)\|\mathbf{u}_{k+1} - \mathbf{u}_k\| + \|\mathbf{u}_{k+1}\|
\end{aligned}$$

where (a) follows from the triangle inequality and (b) is due to the selection process in Step 4 in Algorithm 1. It then follows that, for $r = 2, 3, 4$, we have:

$$\|\tilde{g}(\mathbf{w}_{k+1})\|^r \leq \hat{A}_r\|\mathbf{u}_{k+1} - \mathbf{u}_k\|^r + \hat{B}_r\|\mathbf{u}_{k+1}\|^r, \quad (7)$$

where $\hat{A}_r$ and $\hat{B}_r$ are constants that depend on r. For example, if $r = 2$, then after expanding (7) and collecting terms, we have $\hat{A}_r = 2(1+\gamma)^2$ and $\hat{B}_r = 2$. The derivations for $r = 3, 4$ follow similar processes but are more tedious, and so we omit the details here for brevity. From (7), we further have that:

$$\mathbb{E}\|\tilde{g}(\mathbf{w}_{k+1})\|^r \leq \hat{A}_r \underbrace{\mathbb{E}\|\mathbf{u}_{k+1} - \mathbf{u}_k\|^r}_{\text{(P1)}} + \hat{B}_r \underbrace{\mathbb{E}\|\mathbf{u}_{k+1}\|^r}_{\text{(P2)}}, \quad (8)$$

For $r = 2$, to bound (P1) in (8), we have from Lemma 1 that:

$$\begin{aligned}
\mathbb{E}\|\mathbf{u}_{k+1} - \mathbf{u}_k\|^2 &\leq 2\mathbb{E}\|\mathbf{u}_{k+1}\|^2 + 2\mathbb{E}\|\mathbf{u}_k\|^2 \\
&\leq 2(C_2 + D_2\|\mathbf{w}_{k+1}\|^2) + 2(C_2 + D_2\|\mathbf{w}_k\|^2) \\
&= 4C_2 + 2D_2\|\mathbf{w}_{k+1}\|^2 + 2D_2\|\mathbf{w}_k\|^2.
\end{aligned}$$

Recall that $\mathbf{w}_{k+1} = \mathbf{w}_k - \eta_k\tilde{g}(\mathbf{w}_k)$, which implies that $\|\mathbf{w}_k\| \leq \|\mathbf{w}_{k+1}\| + \eta_k\|\tilde{g}(\mathbf{w}_k)\|$. It then follows that:

$$\begin{aligned}
\|\mathbf{w}_k\|^2 &\leq 2\|\mathbf{w}_{k+1}\|^2 + 2\eta_k^2\|\tilde{g}(\mathbf{w}_k)\|^2 \\
&\leq 2\|\mathbf{w}_{k+1}\|^2 + 2\eta_k^2(A_2 + B_2\|\mathbf{w}_k\|^2). \quad (9)
\end{aligned}$$

After rearranging (9), we have

$$(1-2B_2\eta_k^2)\|\mathbf{w}_k\|^2 \leq 2\|\mathbf{w}_{k+1}\|^2 + 2A_2\eta_k^2$$

Since $\lim_{t\to\infty}\eta_t = 0$, there exists some k_0 such that $\forall k > k_0$, we have $1-2B_2\eta_k^2 > 0$. For notational convenience, we let $R_1^{(2)} \triangleq \frac{1}{2B_2}$. Then, we have:

$$\|\mathbf{w}_k\|^2 \leq \frac{2}{(1-2B_2\eta_k^2)}\|\mathbf{w}_{k+1}\|^2 + \frac{2A_2\eta_k^2}{(1-2B_2\eta_k^2)}. \quad (10)$$

Plugging (10) into (9) yields:

$$\mathbb{E}\|\mathbf{u}_{k+1}-\mathbf{u}_k\|^2 \leq 4C_2 + 2D_2\|\mathbf{w}_{k+1}\|^2 + 2D_2\|\mathbf{w}_k\|^2$$
$$= \Big(4C_2 + \frac{4A_2D_2\eta_k^2}{1-2B_2\eta_k^2}\Big) + \Big(2D_2 + \frac{4D_2}{1-2B_2\eta_t^2}\Big)\|\mathbf{w}_{k+1}\|^2. \quad (11)$$

Using Lemma 1, we have:

$$\mathbb{E}\|\mathbf{u}_{k+1}\|^2 \leq C_2 + D_2\|\mathbf{w}_{k+1}\|^2 \quad (12)$$

Combining (11) and (12), we obtain:

$$\mathbb{E}\|\tilde{g}(\mathbf{w}_{k+1})\|^2 \leq A_2' + B_2'\|\mathbf{w}_{k+1}\|^2,$$

where constants A_2' and B_2' are defined as:

$$A_2' \triangleq 2(1+\gamma)^2(4C_2 + \frac{4A_2D_2\eta_k^2}{1-2B_2\eta_k^2}) + 2C_2,$$
$$B_2' \triangleq 2(1+\gamma)^2(4D_2 + \frac{4D_2}{1-2B_2\eta_k^2}) + 2D_2.$$

To guarantee $A_2' \leq A_2$ and $B_2' \leq B_2$, it suffices to have:

$$\eta_k^2 \leq \frac{4A_2 - [8(1+\gamma)^2+2]C_2}{4A_2D_2 - 2B_2C_2[8(1+\gamma)^2+2] + 2A_2B_2} \triangleq R_2^{(2)}$$
$$\eta_k^2 \leq \frac{B_2 - 2[6(1+\gamma)^2-1]D_2}{2B_2^2 - 8B_2D_2[(1+\gamma)^2] - 4B_2D_2} \triangleq R_3^{(2)}$$

Letting $h_2(A_2,B_2,C_2,D_2) \triangleq [\min(R_1^{(2)},R_2^{(2)},R_3^{(2)})]^{\frac{1}{2}}$ completes the proof for $r=2$. For $r=3,4$, the proofs follow similar processes and we omit them in here for brevity. □

Proof of Lemma 3. With Lemma 2 and Assumption 6, the global confinement of $\{\mathbf{w}_k\}$ follows from [16, Sections 5.1–5.2] and [12, Lemma 2]. □

Proof of Theorem 1. To show that LICM-SGD is (α,q)–Byzantine resilient, we first verify Condition i) in Definition 2, i.e. $\langle \mathbb{E}\tilde{g}(\mathbf{w}_k), \nabla F(\mathbf{w}_k)\rangle \geq (1-\sin\alpha)\|\nabla F(\mathbf{w}_k)\|^2$. Note that

$$\|\tilde{g}(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_{k+1})\| \overset{(a)}{\leq} \|\tilde{g}(\mathbf{w}_{k+1}) - \mathbf{u}_k\| + \|\mathbf{u}_{k+1} - \mathbf{u}_k\| + \|\mathbf{u}_{k+1} - \nabla F(\mathbf{w}_{k+1})\|$$
$$\overset{(b)}{\leq} (1+\gamma)\|\mathbf{u}_{k+1} - \mathbf{u}_k\| + \|\mathbf{u}_{k+1} - \nabla F(\mathbf{w}_{k+1})\|,$$

where (a) follows from the triangle inequality and (b) is due to the selection process in Step 4 in Algorithm 1. Following the same token, we have

$$\|\mathbf{u}_{k+1} - \mathbf{u}_k\| \leq \|\mathbf{u}_{k+1} - \nabla F(\mathbf{w}_{k+1})\| + \|\mathbf{u}_k - \nabla F(\mathbf{w}_k)\| + \|\nabla F(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_k)\|,$$

which implies:

$$\mathbb{E}\|\mathbf{u}_{k+1} - \mathbf{u}_k\| \leq \mathbb{E}\|\mathbf{u}_{k+1} - \nabla F(\mathbf{w}_{k+1})\| + \mathbb{E}\|\mathbf{u}_k - \nabla F(\mathbf{w}_k)\| + \mathbb{E}\|\nabla F(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_k)\| \overset{(a)}{\leq} 2\sqrt{d}\sigma + L\eta_k\mathbb{E}\|\tilde{g}(\mathbf{w}_k)\|,$$

where (a) follows from Lemma 1, Assumption 2 and the update in (5). It then follows that

$$\mathbb{E}\|\tilde{g}(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_{k+1})\| \leq (3+2\gamma)\sqrt{d}\sigma + (1+\gamma)L\eta_k\mathbb{E}\tilde{g}(\mathbf{w}_k).$$

By Lemma 3, $\mathbf{w}_k$ is global confined. Hence, all continuous functions of $\mathbf{w}_k$ are bounded, including $\|\mathbf{w}_k\|^2$, $\mathbb{E}\tilde{g}(\mathbf{w}_k)$, and all derivatives of the cost function $F(\mathbf{w}_k)$. Since $\lim_{k\to\infty}\eta_k = 0$, for any $\epsilon > 0$, there exists a k_ϵ, such that $\forall k \geq k_\epsilon$, $(1+\gamma)L\eta_k\mathbb{E}\tilde{g}(\mathbf{w}_k) \leq \epsilon\mathbb{E}(\mathbf{u}_k - \nabla F(\mathbf{w}_k)) \leq \epsilon\sqrt{d}\sigma_k$. Hence,

$$\mathbb{E}\|\tilde{g}(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_{k+1})\| \leq (3+2\gamma+\epsilon)\sqrt{d}\sigma_k.$$

Thus, if $\|\nabla F(\mathbf{w}_{k+1})\| > \|\tilde{g}(\mathbf{w}_{k+1}) - \nabla F(\mathbf{w}_{k+1})\|$, we have

$$\sin\alpha = \sup_{\forall k}\Big\{\frac{(3+2\gamma+\epsilon)\sqrt{d}\sigma_k}{\|\nabla F(\mathbf{w}_k)\|}\Big\}.$$

This completes the proof of verifying Condition i).

Next, we verify Condition ii), i.e., the r-th moment of $\tilde{g}(\mathbf{w})$ is bounded by a linear growth with respect to $\|\mathbf{w}\|^r$, $r=2,3,4$. Note that this is exactly what has been proved in Lemma 2, which implies that Condition ii) is satisfied. This completes the proof. □

Proof of Theorem 2. The proof of Theorem 2 follows a similar framework to that used in [16] to prove stochastic approximation without attacks. We show that, using our aggregation rule in Algorithm 1, the same convergence result can still be obtained under Byzantine attacks. Let $\mathcal{P}_k$ denote the σ–algebra (i.e., the information available) available up to iteration k. Applying the first-order Taylor expansion on $F(\cdot)$ and using the update step (5), we have:

$$F(\mathbf{w}_{k+1}) - F(\mathbf{w}_k) \leq -2\eta_k\langle\tilde{g}(\mathbf{w}_k), \nabla F(\mathbf{w}_k)\rangle + \eta_k^2\tilde{g}^2(\mathbf{w}_k)K_1,$$

where K_1 is a constant that upper bounds the second derivative (cf. Assumption 5). Then, we have

$$\mathbb{E}[F(\mathbf{w}_{k+1}) - F(\mathbf{w}_k)|\mathcal{P}_k] \overset{(a)}{\leq} -2\eta_k(1-\sin\alpha)(\nabla F(\mathbf{w}_k))^2 + \eta_k^2\mathbb{E}[\tilde{g}^2(\mathbf{w}_k)]K_1 \overset{(b)}{\leq} \eta_k^2K_2K_1, \quad (13)$$

where (a) is due to $\tilde{g}(\mathbf{w}_k)$ is $(\alpha,q)-$Byzantine resilient and (b) follows from the global confinement of $\mathbf{w}_k$. Let $\delta_k(x) = 1$ if $x > 0$ or 0 otherwise. Then, taking expectation on both sides of (13) yields $\mathbb{E}[\delta_k(F(\mathbf{w}_{k+1}) - F(\mathbf{w}_k))] \leq \eta_k^2K_2K_1$. Note that the right-hand side is a convergent infinite sum due to step-sizes Assumption 4. By quasi-martingale convergence theorem [16], the sequence $F(\mathbf{w}_k)$ converges almost surely, i.e. $F(\mathbf{w}_k) \xrightarrow{a.s.} F_\infty$. Summing (13) over $k = 1,2,...,\infty$, the convergence of $F(\mathbf{w}_k)$ implies that:

$$\sum_{k=1}^{\infty}\eta_k(\nabla F(\mathbf{w}_k))^2 < \infty \quad a.s. \quad (14)$$

Now define $\hat{g}_k = \|\nabla F(\mathbf{w}_k)\|^2$, following the same process, we can obtain: $\mathbb{E}[\delta_k(\hat{g}_{k+1} - \hat{g}_k)] = \mathbb{E}[\delta_k \mathbb{E}[\hat{g}(\mathbf{w}_{k+1}) - \hat{g}(\mathbf{w}_k)|\mathcal{P}_k]] \leq \eta_k(\nabla F(\mathbf{w}_k))^2 + \eta_k^2 K_2 K_3$. The two terms on right-hand side are summands of convergent infinite sums by Assumption 4. Again, by quasi-martingale convergence theorem, we have $\hat{g}_k \xrightarrow{a.s.} 0, \nabla F(\mathbf{w}_k) \xrightarrow{a.s.} 0$. □

Proof of Proposition 3. To obtain the median of an m-sized array, the average time complexity is $O(m)$. Repeating this process d times in our aggregation rule yields a time-complexity $O(md)$. Plus, the time-complexity of comparing the median and each returned gradient us $O(md)$. Thus, the total time-complexity is $O(md)$. □

V. Numerical Results

In this section, we conduct experiments to show the convergence and robustness of our LICM-SGD algorithm and compare it with state-of-art Byzantine-resilient algorithms.

Experimental Setup: We consider training multi-class logistic regression (MLR) and convolutional neural network (CNN) classifiers based on the MNIST and CIFAR-10 datasets. The details of CNN classifier are shown in Table I. The batch size is 32 and 64 for MLR and CNN classifiers on MNIST, respectively. The batch size is 128 for both MLR and CNN classifiers on CIFAR-10. γ is set to 10 in all classify tasks. In each experiment, 40 workers are launched to form the distributed learning process. The number of Byzantine attackers are set to 0, 8, 12, and 18, respectively.

TABLE I
CNN architecture used for two datasets.

Layer Type	Size
Convolution + ReLU	$3 \times 3 \times 16$
Max Pooling	2×2
Convolution + ReLU	$3 \times 3 \times 16$
Max Pooling	2×2
Fully Connected + ReLU	10
Softmax	10

We use the normal averaging method without any attacks as the ground truth baseline, denoted by "Mean (no attack)." In addition, we compare our method with state-of-art Byzantine tolerant algorithms, which can be categorized into two classes. The first class is geometric-median-based algorithms, including Krum [5] and Bulyan [7]. The second class is coordinate-wise median-based algorithms, including Median and Trimmed mean [6].

Byzantine Attack Models: We consider three popular Byzantine attack models in this paper, namely, Gaussian Attacks, Label Flipping Attacks, and Omniscient Attacks.

1) Gaussian Attacks: Byzantine attackers draw fake gradients from a Gaussian distribution with mean zero and isotropic covariance matrix with standard deviation 200 to replace the true gradient estimations.

2) Label Flipping Attacks: Byzantine attackers do not require any knowledge of the training data distribution. In our experiment, we generate label flipping attacks as follows: On each Byzantine machine, we replace every training label l with $9 - l$, e.g, 2 is replaced with 7, etc.

3) Omniscient Attacks: Byzantine attackers know all the benign gradients of the total m workers. Then, they replace the benign gradients with the opposite value multiplied by an arbitrarily large factor.

In Fig. 2, we evaluate the effects of three different attacks on Mean method and found that even the Mean method can tolerance small fraction of Gaussian and Label Flipping attacks. From Fig. 2 and prior results [5], [9], it can be seen that the Omniscient Attacks model is much stronger than the other two. Thus, in the rest of the experiments, we only show the results of Omniscient Attacks.

Results: *1) Resilience to Byzantine attacks:* In Fig. 3, we test MLR task on MNIST with a small fraction of attackers $q = 8, m = 40$. Not surprisingly, all Byzantine-resilient algorithms can reach reasonable accuracy. Our LICM-SGD algorithm has the same accuracy as Bulyan, outperforming others. In Fig. 4, we illustrate the CNN results for MNIST with $q = 18, m = 40$, which is the maximum defendable number of attackers $(2q + 1 < m)$. All other methods fail completely with accuracy of less than 30%. Our LICM-SGD algorithm achieves 85% accuracy, only slightly lower than the ground truth baseline. Here we do not compare with Bulyan since it requires $4q+3 < m$, hence at most $1/4$ being Byzantine asymptotically. Hence, Bulyan cannot be directly compared to LICM-SGD, which allows up to $1/2$ of workers being Byzantine. We can obtain similar results on dataset CIFAR-10 for both MLR and CNN tasks under a small and large number of Byzantine attackers in Figs. 5 and 6. We note that in all experiments, the performance of Krum [5] is not stable and its performance could vary significantly with different data samples. In Figs. 3–6, Krum performs worse than other methods. But we do find in other examples that Krum's performance could be comparable to other methods, which is consistent with the observations in [7], [10]. In contrast, the performance of LICM-SGD is insensitive to the changes in data samples.

2) Cost of Byzantine attacks: Next, we evaluate the sensitivity of our LICM-SGD to the number of Byzantine attackers. Fig. 7 illustrates the results of MLR with Omniscient Attacks on MNIST. We can see that the presence of Byzantine attacks only has a small impact on accuracy. The accuracy drops from 87.5% to 83.2% with the attackers' number ascendant from 0 to 18, which shows that our LICM-SGD algorithm is insensitive to the number of Byzantine attackers. Also, LICM-SGD is a more practical algorithm than other methods since it does not require any prior information about the number of attackers. The number of attackers can vary greatly in practice, which could be problematic for other methods. Note that the value of γ does not change in the experiment of each dataset, no matter how q changes. In other words, γ depends only on the dataset, but not the number of attackers. Note that these performances are achieved with an optimal time-complexity $O(md)$, i.e., same as the baseline mean method with no attacks.

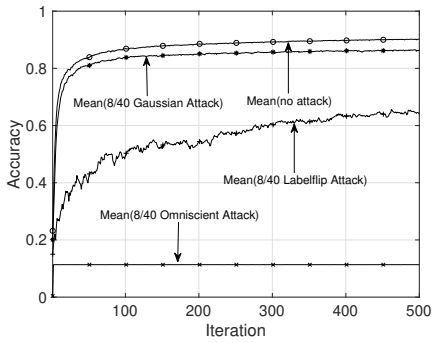

Fig. 2. MLR with different Byzantine attacks on MNIST, $q = 8, m = 40$.

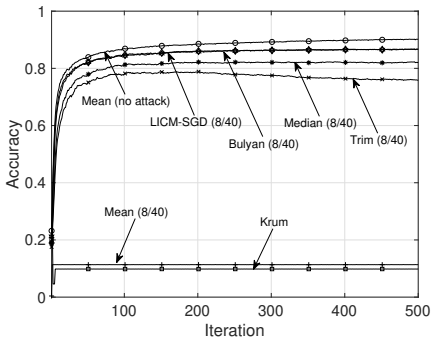

Fig. 3. MLR with Omniscient Attacks on MNIST, $q = 8, m = 40$.

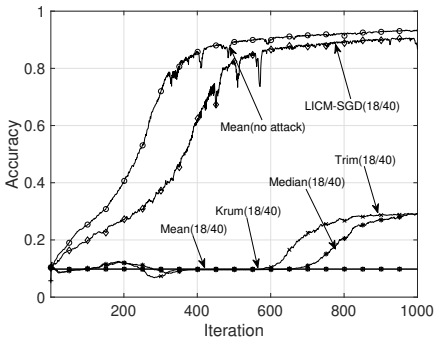

Fig. 4. CNN with Omniscient Attacks on MNIST, $q = 18, m = 40$.

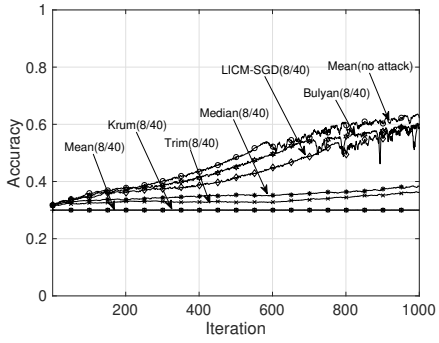

Fig. 5. CNN with Omniscient Attacks on CIFAR-10, $q = 8, m = 40$.

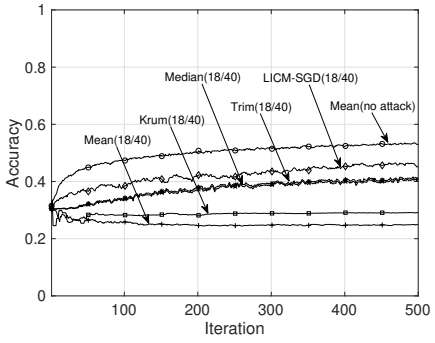

Fig. 6. MLR with Omniscient Attacks on CIFAR-10, $q = 18, m = 40$.

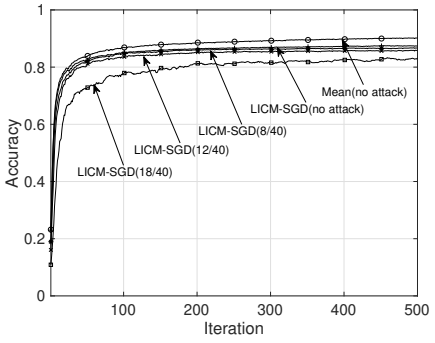

Fig. 7. MLR with Omniscient Attacks on MNIST for LICM-SGD, $q = 0, 8, 12, 18, m = 40$.

VI. Conclusion

In this paper, we proposed a new Lipschitz-inspired coordinate-wise stochastic gradient descent method (LICM-SGD). Our LICM-SGD method can resist up to one half of the workers being Byzantine, while still achieving the same convergence performance compared to the no-attack scenario. Note that our LICM-SGD algorithm does not require knowledge of the number of Byzantine workers. Thus, it is more practical compared to existing works in the literature. Also, our proposed LICM-SGD approach enjoys a low $O(md)$ computational time-complexity, which is the same as the basic distributed SGD algorithm and significantly lower than the $O(m^2d)$ time-complexity of existing methods. We have also conducted extensive numerical studies to verify the performance of our LICM-SGD method. Our numerical results showed that the classification accuracy under LICM-SGD is near that of the standard distributed SGD method with no attacks (at most 5% lower), and consistently performs better than existing works. Our results in this work contribute to the increasingly important field of Byzantine-resilient distributed learning.

References

[1] M. Abadi, P. Barham *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. of USENIX OSDI*, 2016.

[2] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems," in *Proc. of NIPS Workshop on Machine Learning Systems (LearningSys)*, 2016.

[3] Microsoft cognitive toolkit. [Online]. Available: https://www.microsoft.com/en-us/cognitive-toolkit/

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. of the 22nd ACM International Conference on Multimedia*, 2014, pp. 675–678.

[5] P. Blanchard, R. Guerraoui, J. Stainer *et al.*, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 119–129.

[6] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," *arXiv preprint arXiv:1803.01498*, 2018.

[7] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," *arXiv preprint arXiv:1802.07927*, 2018.

[8] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, p. 44, 2017.

[9] C. Xie, O. Koyejo, and I. Gupta, "Generalized byzantine-tolerant sgd," *arXiv preprint arXiv:1802.10116*, 2018.

[10] ——, "Zeno: Byzantine-suspicious stochastic gradient descent," *arXiv preprint arXiv:1805.10032*, 2018.

[11] ——, "Phocas: dimensional byzantine-resilient stochastic gradient descent," *arXiv preprint arXiv:1805.09682*, 2018.

[12] G. Damaskinos, R. Guerraoui, R. Patra, M. Taziki *et al.*, "Asynchronous byzantine machine learning (the case of SGD)," in *International Conference on Machine Learning*, 2018, pp. 1153–1162.

[13] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 1951.

[14] D. Alistarh, Z. Allen-Zhu, and J. Li, "Byzantine stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2018, pp. 4618–4628.

[15] X. Cao and L. Lai, "Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers," in *Proc. IEEE ICASSP*, 2018.

[16] L. Bottou, "Online learning and stochastic approximations," in *On-line Learning in Neural Networks*, D. Saad, Ed. New York, NY, USA: Cambridge University Press, 1998, vol. 17, no. 9, pp. 9–42.