Generalized Byzantine-tolerant SGD

Cong Xie [1] **Oluwasanmi Koyejo** [1] **Indranil Gupta** [1]

Abstract

We propose three new robust aggregation rules for distributed synchronous Stochastic Gradient Descent (SGD) under a general Byzantine failure model. The attackers can arbitrarily manipulate the data transferred between the servers and the workers in the parameter server (PS) architecture. We prove the Byzantine resilience properties of these aggregation rules. Empirical analysis shows that the proposed techniques outperform current approaches for realistic use cases and Byzantine attack scenarios.

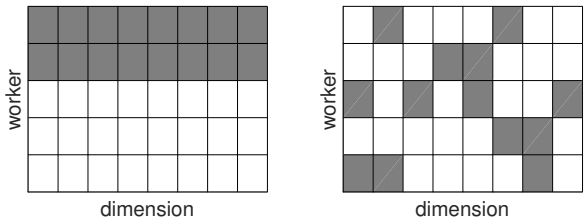
1. Introduction

The failure resilience of distributed machine-learning systems has attracted increasing attention (Blanchard et al., 2017; Chen et al., 2017) in the community. Larger clusters can accelerate training. However, this makes the distributed system more vulnerable to different kinds of failures or even attacks, including crashes and computation errors, stalled processes, or compromised sub-systems (Harinath et al., 2017). Thus, failure/attack resilience is becoming more and more important for distributed machine-learning systems, especially for large-scale deep learning (Dean et al., 2012; McMahan et al., 2017).

In this paper, we consider the most general failure model, Byzantine failures (Lamport et al., 1982), where the attackers can know any information of the other processes, and attack any value in transmission. To be more specific, the data transmission between the machines can be replaced by arbitrary values. Under such model, there are no constraints on the failures or attackers.

The distributed training framework studied in this paper is the Parameter Server (PS). The PS architecture is composed of the server nodes and the worker nodes. The server nodes maintain a global copy of the model, aggregate the gradients from the workers, apply the gradients to the model, and broadcast the latest model to the workers. The worker nodes pull the latest model from the server nodes, compute the gradients according to the local portion of the training data, and send the gradients to the server nodes. The entire dataset and the corresponding workload is distributed to multiple worker nodes, thus parallelizing the computation via partitioning the dataset. There exist several distributed machine learning systems using the PS architecture. For instance, Tensorflow (Abadi et al., 2016), CNTK (Seide & Agarwal, 2016), and MXNet (Chen et al., 2015) implement internal PS's.

In this paper, we study the Byzantine resilience of synchronous Stochastic Gradient Descent (SGD), which is a popular class of learning algorithms using PS architecture. Its variants are widely used in training deep neural networks (Kingma & Ba, 2014; Mukkamala & Hein, 2017). Such algorithms always wait to collect gradients from all the worker nodes before moving on to the next iteration.



(a) Classic Byzantine (b) Generalized Byzantine

Figure 1. The 2 figures visualize 5 workers with 8-dimensional gradients. The ith row represents the gradient vector produced by the ith worker. The jth column represents the jth dimension of the gradients. A shadow block represents that the corresponding value is replaced by a Byzantine value. In the two examples, the maximal number of Byzantine values for each dimension is 2. For the classic Byzantine model, all the Byzantine values must lie in the same workers (rows), while for the generalized Byzantine model there is no such constraint. Thus, (a) is a special case of (b).

The failure model can be described by using an $n \times d$ matrix consisting of the d-dimensional gradients produced by n workers, as visualized in Figure 1. A previous work (Blanchard et al., 2017) discusses a special case of our failure model, where the Byzantine values must lie in the same rows (workers) as shown in Figure 1(a). Our failure model generalize the classic Byzantine failure model by placing the Byzantine values anywhere in the matrix without any constraint.

There are many possible types of attacks. In general, the attackers want to disturb the model training, i.e., make SGD

converge slowly or converge to a bad solution. We list some of the possible attacks in the following three paragraphs.

We name the most general type of attacks as *gamber*. The attackers can change a portion of data on the communication media such as the wires or the network interfaces. The attackers randomly pick the data and maliciously change them (e.g., multiply them by a large negative value). As a result, on the server nodes, the collected gradients are partially replaced by arbitrary values.

Another possible type of attack is called *omniscient*. The attackers are supposed to know the gradients sent by all the workers, and use the sum of all the gradients, scaled by a large negative value, to replace some of the gradient vectors. The goal is to mislead SGD to go into an opposite direction with a large step size.

There are also some weaker attacks, such as *Gaussian attack*, where some of the gradient vectors are replaced by random vectors sampled from a Gaussian distribution with large variances. Such attackers do not require any information from the workers.

With the generalized Byzantine failure model, we ask that using what aggregation rules and on what conditions, the synchronous SGD can still converge to good solutions. We propose novel median-based aggregation rules, with which SGD is Byzantine resilient on a certain condition: for each dimension, in all the n values provided by the n workers, the number of Byzantine values must be less than half of n. Such Byzantine resilience property is called "dimensional Byzantine resilience". The main contributions of this paper are listed below:

- We propose three aggregation rules for synchronous SGD with provable convergence to critical points: geometric median (Definition 6), marginal median (Definition 7), and "mean around median" (Definition 8). As far as we know, this paper is the first to theoretically and empirically study median-based aggregation rules under non-convex settings.

- We show that the three proposed robust aggregation rules have low computation cost. The time complexities are nearly linear, which are in the same order of the default choice for non-Byzantine aggregation, i.e., averaging.

- We formulate the dimensional Byzantine resilience property, and prove that marginal median and "mean around median" are dimensional Byzantine-resilient (Definition 5). As far as we know, this paper is the first one to study generalized Byzantine failures and dimensional Byzantine resilience for synchronous SGD.

2. Model

We consider the Parameter Server architecture consisting of n workers. The goal is to find the optimizer of the following problem:

$$\min_x \mathbb{E}\left[f(x,\xi)\right],$$

where the expectation is with respect to the random variable ξ. The PS executes synchronous SGD for distributed training. In each round, the server nodes collect n gradients from the workers. In the t^{th} round, the server nodes aggregate the gradients $\{\tilde{v}_i^t : i \in [n]\}$ from the workers, and broadcast the updated parameters x^{t+1} to the workers. $\tilde{v}_i^t$ is the vector sent by the ith worker in the tth round, potentially Byzantine. Using aggregation rule $Aggr(\cdot)$, the server nodes update the parameters as follows:

$$x^{t+1} \leftarrow x^t - \gamma^t Aggr(\{\tilde{v}_i^t : i \in [n]\}),$$

where γ^t is the learning rate. The worker nodes pull the latest parameters from the server nodes, compute the gradients according to the local portion of the training data, and send the gradients to the server nodes. Without the Byzantine failures, the ith worker will calculate $v_i^t \sim G^t$, where $G^t = \nabla f(x^t, \xi)$. With Byzantine failures, v_i^t are partially replaced by any arbitrary values, which results $\tilde{v}_i^t$.

Since the Byzantine failure assumes the worst cases, the attackers may have full knowledge of the entire system, including the gradients generated by all the workers, and the aggregation rule $Aggr(\cdot)$. The malicious processes can even collaborate with each other (Lynch, 1996).

3. Byzantine Resilience

In this section, we formally define the classic Byzantine resilience property and its generalized version: dimensional Byzantine resilience.

Suppose that in a specific round, the correct vectors $\{v_i : i \in [n]\}$ are i.i.d samples drawn from the random variable $G = \nabla f(x, \xi)$, where $\mathbb{E}[G] = g$ is an unbiased estimator of the gradient. Thus, $\mathbb{E}[v_i] = \mathbb{E}[G] = g$, for any $i \in [n]$. We simplify the notations by ignoring the index of round t.

We first introduce the classic Byzantine model proposed by Blanchard et al. (2017). With the Byzantine workers, the actual vectors $\{\tilde{v}_i : i \in [n]\}$ received by the server nodes are as follows:

Definition 1 (Classic Byzantine Model)**.**

$$\tilde{v}_i = \begin{cases} v_i, \textit{if the } i\textit{th worker is correct,} \\ \textit{arbitrary, if the } i\textit{th worker is Byzantine.} \end{cases} \tag{1}$$

Note that the indices of Byzantine workers can change throughout different rounds. Furthermore, the server nodes

are not aware of which workers are Byzantine. The only information given is the number of Byzantine workers, if necessary.

We directly use the same definition of classic Byzantine resilience proposed in (Blanchard et al., 2017).

Definition 2. *(Classic (α, q)-Byzantine Resilience). Let $0 \leq \alpha < \pi/2$ be any angular value, and any integer $0 \leq q \leq n$. Let $\{v_i : i \in [n]\}$ be any i.i.d. random vectors in $\mathbb{R}^d$, $v_i \sim G$, with $\mathbb{E}[G] = g$. Let $\{\tilde{v}_i : i \in [n]\}$ be the set of vectors, of which up to q of them are replaced by arbitrary vectors in $\mathbb{R}^d$, while the others still equal to the corresponding $\{v_i\}$. Aggregation rule $Aggr(\cdot)$ is said to be classic (α, q)-Byzantine resilient if $Aggr(\{\tilde{v}_i : i \in [n]\})$ satisfies (i) $\langle \mathbb{E}[Aggr], g\rangle \geq (1 - \sin\alpha)\|g\|^2 > 0$ and (ii) for $r = 2, 3, 4$, $\mathbb{E}\|Aggr\|^r$ is bounded above by a linear combination of terms $\mathbb{E}\|G\|^{r_1}, \ldots, \mathbb{E}\|G\|^{r_{n-q}}$ with $r_1 + \ldots + r_{n-q} = r$.*

The baseline algorithm *Krum*, denoted as $Krum(\{\tilde{v}_i : i \in [n]\})$ (Blanchard et al., 2017), is defined as follows

Definition 3.

$$Krum(\{\tilde{v}_i : i \in [n]\}) = \tilde{v}_k,$$
$$k = \underset{i \in [n]}{\operatorname{argmin}} \sum_{i \to j} \|\tilde{v}_i - \tilde{v}_j\|^2,$$

where $i \to j$ is the indices of the $n-q-2$ nearest neighbours of $\tilde{v}_i$ in $\{\tilde{v}_i : i \in [n]\}$ measured by Euclidean distance.

The *Krum* aggregation is classic (α, q)-Byzantine resilient under certain assumptions:

Lemma 1 (Blanchard et al. (2017))**.** *Let $v_1, \ldots, v_n$ be any i.i.d. random d-dimensional vectors s.t. $v_i \sim G$, with $\mathbb{E}[G] = g$ and $\mathbb{E}\|G - g\|^2 = d\sigma^2$. q of $\{v_i : i \in [n]\}$ are replaced by arbitrary d-dimensional vectors $b_1, \ldots, b_q$. If $2q + 2 < n$ and $\eta_0(n, q)\sqrt{d}\sigma < \|g\|$, where*

$$\eta_0^2(n, q) = 2\left(n - q + \frac{q(n - q - 2) + q^2(n - q - 1)}{n - 2q - 2}\right),$$

then the $Krum$ function is classic (α_0, q)-Byzantine resilient where $0 \leq \alpha_0 < \pi/2$ is defined by $\sin\alpha_0 = \frac{\eta_0(n,q)\sqrt{d}\sigma}{\|g\|}$.

The generalized Byzantine model is denoted as:

Definition 4 (Generalized Byzantine Model)**.**

$$(\tilde{v}_i)_j = \begin{cases} (v_i)_j, \text{if the the } j\text{th dimension of } v_i \text{ is correct,} \\ arbitrary, \text{otherwise,} \end{cases} \tag{2}$$

where $(v_i)_j$ is the jth dimension of the vector v_i.

Based on the Byzantine model above, we introduce a generalized Byzantine resilience property, dimensional (α, q)-Byzantine resilience, which is defined as follows:

Definition 5. *(Dimensional (α, q)-Byzantine Resilience). Let $0 \leq \alpha < \pi/2$ be any angular value, and any integer $0 \leq q \leq n$. Let $\{v_i : i \in [n]\}$ be any i.i.d. random vectors in $\mathbb{R}^d$, $v_i \sim G$, with $\mathbb{E}[G] = g$. Let $\{\tilde{v}_i : i \in [n]\}$ be the set of vectors. For each dimension, up to q of the n values are replaced by arbitrary values, i.e., for dimension $j \in [d]$, q of $\{(\tilde{v}_i)_j : i \in [n]\}$ are Byzantine, where $(\tilde{v}_i)_j$ is the jth dimension of the vector $\tilde{v}_i$. Aggregation rule $Aggr(\cdot)$ is said to be dimensional (α, q)-Byzantine resilient if $Aggr(\{\tilde{v}_i : i \in [n]\})$ satisfies (i) $\langle \mathbb{E}[Aggr], g\rangle \geq (1 - \sin\alpha)\|g\|^2 > 0$ and (ii) for $r = 2, 3, 4$, $\mathbb{E}\|Aggr\|^r$ is bounded above by a linear combination of terms $\mathbb{E}\|G\|^{r_1}, \ldots, \mathbb{E}\|G\|^{r_{n-q}}$ with $r_1 + \ldots + r_{n-q} = r$.*

Note that classic (α, q)-Byzantine resilience is a special case of dimensional (α, q)-Byzantine resilience. For classic Byzantine resilience defined in Definition 2, all the Byzantine values must lie in the same subset of workers, as shown in Figure 1(a).

In the following theorems, we show that *Mean* and *Krum* are not dimensional Byzantine resilient. The proofs are provided in the appendix.

Theorem 1. *Averaging is not dimensional Byzantine resilient.*

Theorem 2. *Any aggregation rule $Aggr(\{\tilde{v}_i : i \in [n]\})$ that outputs $Aggr \in \{\tilde{v}_i : i \in [n]\}$ is not dimensional Byzantine resilient.*

Note that *Krum* chooses the vector $v \in \{\tilde{v}_i : i \in [n]\}$ with the minimal score. Thus, based on the theorem above, we obtain the following corollary.

Corollary 1. *$Krum(\cdot)$ is not dimensional Byzantine resilient.*

If an aggregation rule is dimensional/classic (α, q)-Byzantine resilient with satisfied assumptions, it converges to critical points almost surely, by reusing the Proposition 2 in (Blanchard et al., 2017). We provide the following lemma without proof.

Lemma 2 (Blanchard et al. (2017))**.** *Assume that (i) the cost function f is three times differentiable with continuous derivatives, and is non-negative, $f(x) \geq 0$; (ii) the learning rates satisfy $\sum_t \gamma_t = \infty$ and $\sum_t \gamma_t^2 < \infty$; (iii) the gradient estimator satisfies $\mathbb{E}[\nabla f(x, \xi)] = \nabla F(x)$ and $\forall r \in \{2, 3, 4\}$, $\mathbb{E}\|\nabla f(x, \xi)\|^r \leq A_r + B_r\|x\|^r$ for some constants A_r, B_r; (iv) there exists a constant $0 \leq \alpha < \pi/2$ such that for all x $\eta(n, q)\sqrt{d}\sigma(x) \leq \|\nabla F(x)\| \sin\alpha$, where $d\sigma^2(x) = \mathbb{E}\|\nabla f(x, \xi) - \nabla F(x)\|^2$; (v) finally, beyond a certain horizon, $\|x\|^2 \geq D$, there exist $\epsilon > 0$ and $0 \leq \beta < \pi/2 - \alpha$ such that $\|\nabla F(x)\| \geq \epsilon > 0$, and*

$\frac{\langle x, \nabla F(x)\rangle}{\|x\|\cdot\|\nabla F(x)\|} \geq \cos\beta$. *Then the sequence of gradients* $\nabla F(x^t)$ *converges almost surely to zero, if the aggregation rule satisfies* (α, q)*-Byzantine Resilience defined in Definition 2 or 5.*

4. Median-based Aggregation

With the Byzantine failure model defined in Equation (1) and (2), we propose three median-based aggregation rules, which are Byzantine resilient under certain conditions.

4.1. Geometric Median

The geometric median is used as a robust estimator of mean (Chen et al., 2017).

Definition 6. *The geometric median of* $\{\tilde{v}_i : i \in [n]\}$*, denoted by* $GeoMed(\{\tilde{v}_i : i \in [n]\})$*, is defined as*

$$\lambda = GeoMed(\{\tilde{v}_i : i \in [n]\}) = \operatorname*{argmin}_{v\in\mathbb{R}^d} \sum_{i=1}^{n} \|v - \tilde{v}_i\|.$$

The following theorem shows the classic (α_1, q)-Byzantine resilience of geometric median. A proof is provided in the appendix.

Theorem 3. *Let* $v_1, \ldots, v_n$ *be any i.i.d. random* d*-dimensional vectors s.t.* $v_i \sim G$*, with* $\mathbb{E}[G] = g$ *and* $\mathbb{E}\|G - g\|^2 = d\sigma^2$*. q of* $\{v_i : i \in [n]\}$ *are replaced by arbitrary* d*-dimensional vectors* $b_1, \ldots, b_q$*. If* $q \leq \lceil \frac{n}{2} \rceil - 1$ *and* $\eta_1(n, q)\sqrt{d}\sigma < \|g\|$*, where* $\eta_1(n, q) = \frac{2n-2q}{n-2q}\sqrt{n-q}$*, then the* $GeoMed$ *function is classic* (α_1, q)*-Byzantine resilient where* $0 \leq \alpha_1 < \pi/2$ *is defined by* $\sin\alpha_1 = \frac{\eta_1(n,q)\sqrt{d}\sigma}{\|g\|}$.

4.2. Marginal Median

The marginal median is another generalization of one-dimensional median.

Definition 7. *We define the marginal median aggregation rule* $MarMed(\cdot)$ *as*

$$\mu = MarMed(\{\tilde{v}_i : i \in [n]\}),$$

where for any $j \in [d]$*, the* j*th dimension of* μ *is* $\mu_j = median\left(\{(\tilde{v}_1)_j, \ldots, (\tilde{v}_n)_j\}\right)$*,* $(\tilde{v}_i)_j$ *is the* j*th dimension of the vector* $\tilde{v}_i$*,* $median(\cdot)$ *is the one-dimensional median.*

The following theorem claims that by using $MarMed(\cdot)$, the resulting vector is dimensional (α_2, q)-Byzantine resilient. A proof is provided in the appendix.

Theorem 4. *Let* $v_1, \ldots, v_n$ *be any i.i.d. random* d*-dimensional vectors s.t.* $v_i \sim G$*, with* $\mathbb{E}[G] = g$ *and* $\mathbb{E}\|G - g\|^2 = d\sigma^2$*. For any dimension* $j \in [d]$*, q of* $\{(v_1)_j, \ldots, (v_n)_j\}$ *are replaced by arbitrary values, where* $(v_i)_j$ *is the* j*th dimension of the vector* v_i*. If* $q \leq \lceil \frac{n}{2} \rceil - 1$ *and* $\eta_2(n, q)\sqrt{d}\sigma < \|g\|$*, where* $\eta_2(n, q) = \sqrt{n-q}$*, then the* $MarMed$ *function is dimensional* (α_2, q)*-Byzantine resilient where* $0 \leq \alpha_2 < \pi/2$ *is defined by* $\sin\alpha_2 = \frac{\eta_2(n,q)\sqrt{d}\sigma}{\|g\|}$.

4.3. Beyond Median

We can also utilize more values for each dimension along with the median, if q is given or easily estimated. To be more specific, for each dimension, we take the average of the $n-q$ values nearest to the median (including the median itself). We call the resulting aggregation rule *"mean around median"*, which is defined as follows:

Definition 8. *We define the mean-around-median aggregation rule* $MeaMed(\cdot)$ *as*

$$\rho = MeaMed(\{\tilde{v}_i : i \in [n]\}),$$

where for any $j \in [d]$*, the* j*th dimension of* ρ *is* $\rho_j = \frac{1}{n-q}\sum_{\mu_j \to i}(\tilde{v}_i)_j$*,* $\mu_j \to i$ *is the indices of the top-*$(n-q)$ *values lying in* $\{(\tilde{v}_1)_j, \ldots, (\tilde{v}_n)_j\}$ *nearest to the median* μ_j*,* $(\tilde{v}_i)_j$ *is the* j*th dimension of the vector* $\tilde{v}_i$*.*

We show that $MeaMed$ is dimensional (α_3, q)-Byzantine resilient.

Theorem 5. *Let* $v_1, \ldots, v_n$ *be any i.i.d. random* d*-dimensional vectors s.t.* $v_i \sim G$*, with* $\mathbb{E}[G] = g$ *and* $\mathbb{E}\|G - g\|^2 = d\sigma^2$*. For any dimension* $j \in [d]$*, q of* $\{(v_1)_j, \ldots, (v_n)_j\}$ *are replaced by arbitrary values, where* $(v_i)_j$ *is the* j*th dimension of the vector* v_i*. If* $q \leq \lceil \frac{n}{2} \rceil - 1$ *and* $\eta_3(n, q)\sqrt{d}\sigma < \|g\|$*, where* $\eta_3(n, q) = \sqrt{10(n-q)}$*, then the* $MeaMed$ *function is dimensional* (α_3, q)*-Byzantine resilient where* $0 \leq \alpha_3 < \pi/2$ *is defined by* $\sin\alpha_3 = \frac{\eta_3(n,q)\sqrt{d}\sigma}{\|g\|}$.

The mean-around-median aggregation can be viewed as a trimmed average centering at the median, which filters out the values far away from the median.

4.4. Time Complexity

For geometric median $GeoMed(\cdot)$, there are no closed-form solutions. The $(1+\epsilon)$-approximate geometric median can be computed in $O(dn\log^3\frac{1}{\epsilon})$ time (Cohen et al., 2016), which is nearly linear to $O(dn)$. To compute the marginal median $MarMed(\cdot)$, we only need to compute the median value of each dimension. The simplest way is to apply any sorting algorithm to each dimension, which yields the time complexity $O(dn\log n)$. To obtian median values, there also exists an algorithm called *selection algorithm* (Blum et al., 1973) with average time complexity $O(n)$ ($O(n^2)$ in the worst case). Thus, we can get the marginal median with time complexity $O(dn)$ on average, which is in the same order of using mean value for aggregation. For $MeaMed(\cdot)$, the

computation additional to computing the marginal median takes linear time $O(dn)$. Thus, the time complexity is the same as $MarMed(\cdot)$. Note that for Krum and Multi-Krum, the time complexity is $O(dn^2)$ (Blanchard et al., 2017).

5. Experiments

In this section, we evaluate the convergence and Byzantine resilience properties of the proposed algorithms. We consider two image classification tasks: handwritten digits classification on MNIST dataset using multi-layer perceptron (MLP) with two hidden layers, and object recognition on convolutional neural network (CNN) with five convolutional layers and two fully-connected layers. The details of these two neural networks can be found in the appendix. There are $n = 20$ worker processes. We repeat each experiment for ten times and take the average. To make the conditions as fair as possible for all the algorithms, we ensure that all the algorithms are run with the same set of random seeds. The details of the datasets and the default hyperparameters of the corresponding models are listed in Table 1. We use top-1 or top-3 accuracy on testing sets (disjoint with the training sets) as evaluation metrics.

The baseline aggregation rules are *Mean*, *Medoid*, *Krum* (Definition 3), and *Multi-Krum*. Medoid, defined as follows, is a computation-efficient version of geometric median.

Definition 9. *The medoid of* $\{\tilde{v}_i : i \in [n]\}$*, denoted by* $Medoid(\{\tilde{v}_i : i \in [n]\})$*, is defined as* $Medoid(\{\tilde{v}_i : i \in [n]\}) = \text{argmin}_{v \in \{\tilde{v}_i : i \in [n]\}} \sum_{i=1}^{n} \|v - \tilde{v}_i\|$.

Multi-Krum is a variant of Krum defined in Blanchard et al. (2017), which takes the average on several vectors selected by multiple rounds of Krum. We compare these baseline algorithms with the proposed algorithms: geometric median (*GeoMed* defined in Definition 6), marginal median (*MarMed* defined in Definition 7), and "mean around median" (*MeaMed* defined in Definition 8) under different settings in the following subsections.

Note that all the experiments of CNN on CIFAR10 show similar results with the experiments of MLP on MNIST. Thus, we only show the results of CNN in Section 5.5 as an example. The remaining results are provided in the appendix.

5.1. Convergence without Byzantine Failures

First, we evaluate the convergence without Byzantine failures. The goal is to empirically evaluate the bias and variance caused by the robust aggregation rules.

In Figure 10, we show the top-1 accuracy on the testing set of MNIST. The gaps between different algorithms are tiny. Among all the algorithms, *Multi-Krum*, *GeoMed*, and *MeaMed* have the least bias. They act just the same as averaging. $MarMed$ converges slightly slower. *Medoid* and *Krum* both have slowest convergence.

5.2. Gaussian Attack

We test classic Byzantine resilience in this experiment. We consider the attackers that replace some of the gradient vectors with Gaussian random vectors with zero mean and isotropic covariance matrix with standard deviation 200. We refer to this kind of attack as *Gaussian Attack*. Within the figure, we also include the averaging without Byzantine failures as a baseline. 6 out of the 20 gradient vectors are Byzantine. The results are shown in Figure 3. As expected, averaging is not Byzantine resilient. The gaps between all the other algorithms are still tiny. *GeoMed* and *MeaMed* performs like there are no Byzantine failures at all. *Multi-Krum* and *MarMed* converges slightly slower. *Medoid* and *Krum* performs worst. Although *Medoid* is not Byzantine resilient, the Gaussian attack is weak enough so that *Medoid* is still effective.

5.3. Omniscient Attack

We test classic Byzantine resilience in this experiment. This kind of attacker is assumed to know the all the correct gradients. For each Byzantine gradient vector, the gradient is replaced by the negative sum of all the correct gradients, scaled by a large constant (1e20 in the experiments). Roughly speaking, this attack tries to make the parameter server go into the opposite direction with a long step. 6 out of the 20 gradient vectors are Byzantine. The results are shown in Figure 4. *MeaMed* still performs just like there is no failure. *Multi-Krum* is not as good as *MeaMed*, but the gap is small. *Krum* converges slower but still converges to the same accuracy. However, *GeoMed* and *MarBed* converge to bad solutions. *Mean* and *Medoid* are not tolerant to this attack.

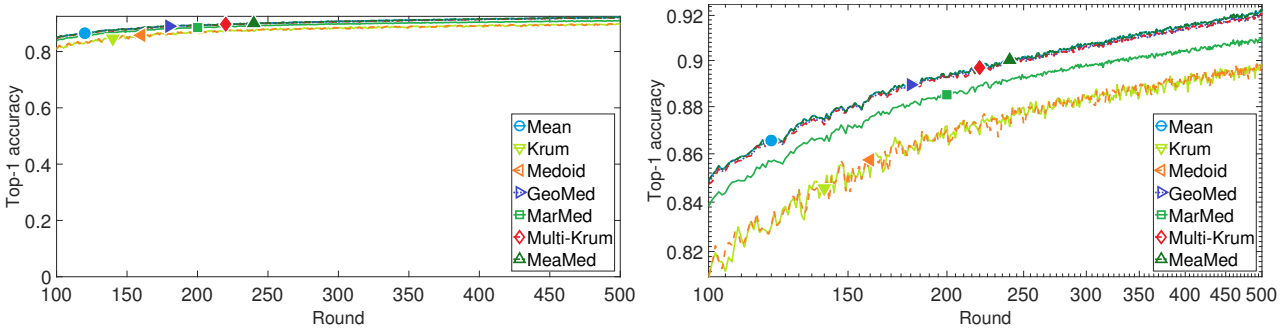
5.4. Bit-flip Attack

We test dimensional Byzantine resilience in this experiment. Knowing the information of other workers can be difficult in practice. Thus, we use more realistic scenario in this experiment. The attacker only manipulates some individual floating numbers by flipping the 22th, 30th, 31th and 32th bits. Furthermore, we test dimensional Byzantine resilience in this experiment. For each of the first 1000 dimensions, 1 of the 20 floating numbers is manipulated using the bit-flip attack. The results are shown in Figure 5. As expected, only *MarMed* and *MeaMed* are dimensional Byzantine resilient.

Note that for *Krum* and *Multi-Krum*, their assumption requires the number of Byzantine vectors q to satisfy $2q+2 < n$, which means $q \leq 8$ in our experiments. However, be-

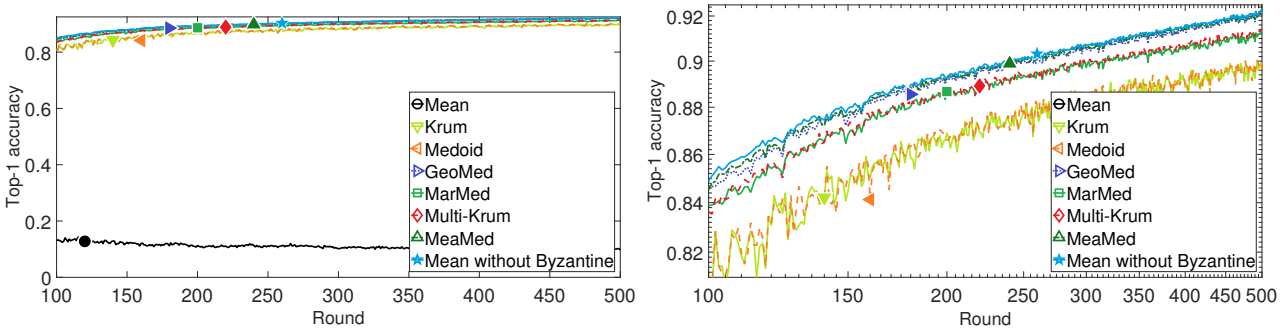
Table 1. Experiment Summary

Dataset	# train	# test	γ	# rounds	Batchsize	Evaluation metric
MNIST (Loosli et al., 2007)	60k	10k	0.1	500	32	top-1 accuracy
CIFAR10 (Krizhevsky & Hinton, 2009)	50k	10k	5e-4	4000	128	top-3 accuracy



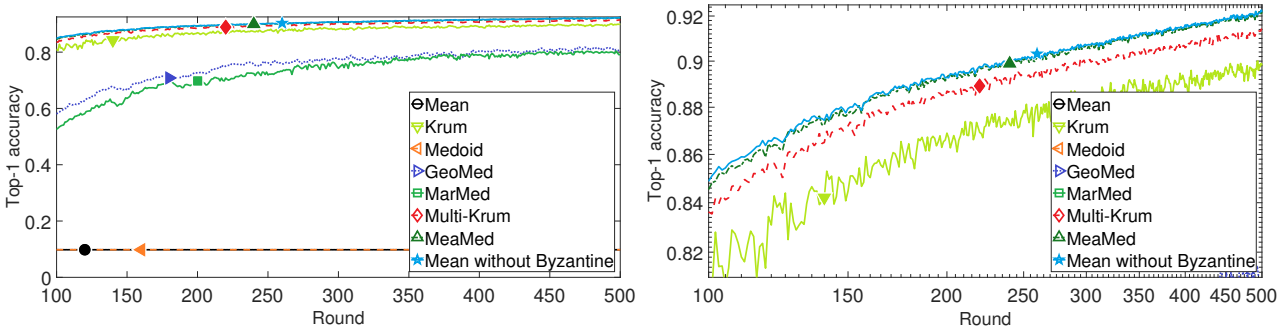
(a) MNIST without Byzantine (b) MNIST without Byzantine (zoomed)

Figure 2. Top-1 accuracy of MLP on MNIST without Byzantine failures.



(a) MNIST with Gaussian (b) MNIST with Gaussian (zoomed)

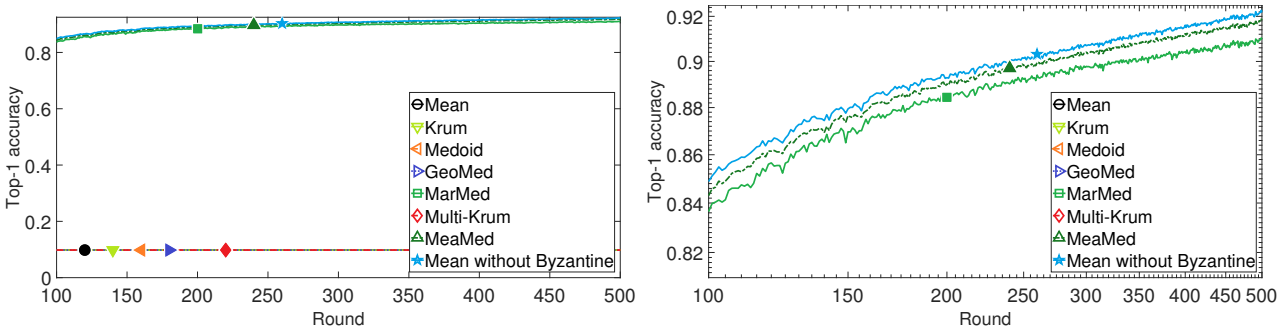
Figure 3. Top-1 accuracy of MLP on MNIST with Gaussian Attack. 6 out of 20 gradient vectors are replaced by i.i.d. random vectors drawn from a Gaussian distribution with 0 mean and 200 standard deviation.



(a) MNIST with omniscient (b) MNIST with omniscient (zoomed)

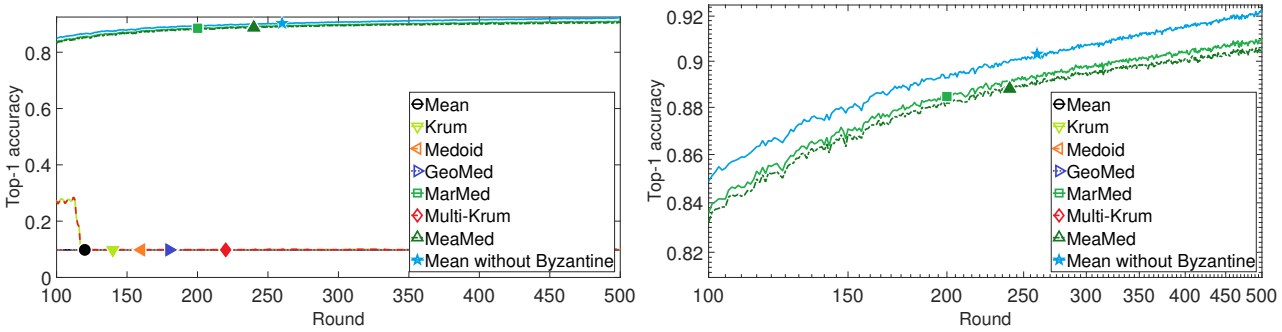
Figure 4. Top-1 accuracy of MLP on MNIST with Omniscient Attack. 6 out of 20 gradient vectors are replaced by the negative sum of all the correct gradients, scaled by a large constant (1e20 in the experiments).

cause each gradient is partially manipulated, all the n vectors are Byzantine, which breaks the assumption of the Krum-based algorithms. Furthermore, to compute the distances to the $(n-q-2)$-nearest neighbours, $n-q-2$ must



(a) MNIST with bit-flip (b) MNIST with bit-flip (zoomed)

Figure 5. Top-1 accuracy of MLP on MNIST with Bit-flip Attack. For the first 1000 dimensions, 1 of the 20 floating numbers is manipulated by flipping the 22th, 30th, 31th and 32th bits.



(a) MNIST with gambler (b) MNIST with gambler (zoomed)

Figure 6. Top-1 accuracy of MLP on MNIST with gambler attack. The parameters are evenly assigned to 20 servers. For one single server, any received value is multiplied by $-1e20$ with probability 0.05%.

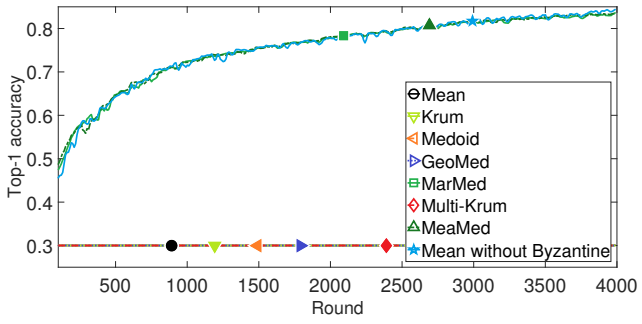


Figure 7. Top-3 Accuracy of CNN on CIFAR10 with gambler.

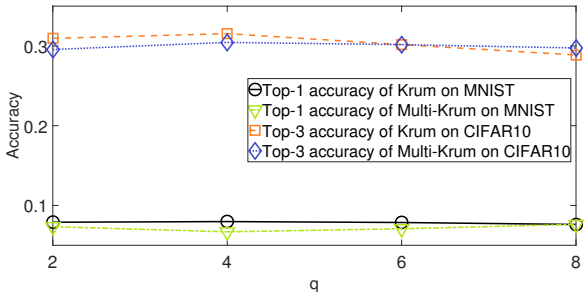


Figure 8. Accuracy of $Krum$-based aggregations, at the end of training, when q varies. With 20 servers, q must satisfy $q \leq 8$.

be positive. To test the performance of *Krum* and *Multi-Krum*, we set $q = 8$ for these two algorithms so that they can still be executed. Furthermore, we test whether tuning q can make a difference. The results are shown in Figure 8. Obviously, whatever q we use, *Krum*-based algorithms get stuck around bad solutions.

5.5. General Attack with Multiple Servers

We test general Byzantine resilience in this experiment. We evaluate the robust aggregation rules under a more general and realistic type of attack. It is very popular to partition the parameters into disjoint subsets, and use multiple server nodes to storage and aggregate them (Li et al., 2014a;b; Ho et al., 2013). We assume that the parameters are evenly partitioned and assigned to the server nodes. The attacker picks one single server, and manipulates any floating number by

multiplying $-1e20$, with probability of 0.05%. We call this attack *gambler*, because the attacker randomly manipulate the values, and wish that in some rounds the assumptions/prerequisites of the robust aggregation rules are broken, which crashes the training. Such attack requires less global information, and can be concentrated on one single server, which makes it more realistic and easier to implement.

In Figure 6 and 7, we evaluate the performance of all the robust aggregation rules under the gambler attack. The number of servers is 20. For *Krum*, *Multi-Krum* and *MeaMed*, the estimated Byzantine number q is set as 8. We also show the performance of averaging without Byzantine values as the benchmark. It is shown that only marginal median *MarMed* and "mean around median" *MeaMed* survive under this attack. The convergence is slightly slower than the averaging without Byzantine values, but the gaps are small.

5.6. Discussion

As expected, *mean* aggregation is not Byzantine resilient. Although *medoid* is not Byzantine resilient, as proved by Blanchard et al. (2017), it can still make reasonable progress under some attacks such as Gaussian attack. *Krum*, *Multi-Krum*, and *GeoMed* are classic Byzantine resilient but not dimensional Byzantine resilient. *MarMed* and *MeaMed* are dimensional Byzantine resilient. However, under omniscient attack, *MarMed* suffers from larger variances, which slow down the convergence.

The gambler attack shows the true advantage of dimensional Byzantine resilience: higher probability of survival. Under such attack, chances are that the assumptions/prerequisites of *MarMed* and *MeaMed* may still get broken. However, their probability of crashing is less than the other algorithms because dimensional Byzantine resilience generalizes classic Byzantine resilience. An interesting observation is that *MarMed* is slightly better than *MeaMed* under gambler attack. That is because the estimation of $q = 8$ is not accurate, which will cause some unpredictable behavior for *MeaMed*. We choose $q = 8$ because it is the maximal value we can take for *Krum* and *Multi-Krum*.

It is obvious that *MeaMed* performs best in almost all the cases. *Multi-Krum* is also good, except that it is not dimensional Byzantine resilient. The reason why *MeaMed* and *Multi-Krum* have better performance is that they utilize the extra information of the number of Byzantine values. Note that *MeaMed* not only performs just as well as or even better than *Multi-Krum*, but also has lower time complexity.

Marginal median *MarMed* has the cheapest computation. Its worst case, omniscient attack, is hard to implement in reality. Thus, for most applications, we suggest MarMed as an easy-to-implement aggregation rule with robust performance, which (importantly) does not require knowledge of the number of byzantine values.

6. Related Works

There are few papers studying Byzantine resilience for machine learning algorithms. Our work is closely related to Blanchard et al. (2017). Another paper (Chen et al., 2017) proposed grouped geometric median for Byzantine resilience, with strongly convex functions.

Our approach offers the following important advantages over the previous work.

- **Cheaper computation compared to Krum.** Geometric median has nearly linear (approximately $O(nd)$) time complexity (Cohen et al., 2016). Marginal median and "mean around median" have linear time complexity $O(nd)$ on average (Blum et al., 1973), while the time complexity of Krum is $O(n^2 d)$.
- **Less prior knowledge required.** Both geometric median and marginal median do not require q, the number of Byzantine workers, to be given, while Krum needs q to calculate the sum of Euclidean distances of the $n - q - 2$ nearest neighbours. Furthermore, when q is known or well estimated, *MeaMed* show better robustness than *Krum* and *Multi-Krum* in most cases.
- **Dimensional Byzantine resilience.** Marginal median and "mean around median" tolerate a more general type of Byzantine failures described in Equation (2) and Definition 5, while Krum and geometric median can only tolerate the classic Byzantine failures described in Equation (1) and Definition 2.
- **Better support for multiple server nodes.** If the entire set of parameters is disjointly partitioned and stored on multiple server nodes, marginal median and "mean around median" need no additional communication, while Krum and geometric median requires communication among the server nodes.

7. Conclusion

We investigate the generalized Byzantine resilience of parameter server architecture. We proposed three novel median-based aggregation rules for synchronous SGD. The algorithms have low time complexity and provable convergence to critical points. Our empirical results show good performance in practice.

References

Abadi, Martín, Barham, Paul, Chen, Jianmin, Chen, Zhifeng, Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Irving, Geoffrey, Isard, Michael, Kudlur, Manjunath, Levenberg, Josh, Monga, Rajat, Moore, Sherry, Murray, Derek Gordon, Steiner, Benoit, Tucker, Paul A., Vasudevan, Vijay, Warden, Pete, Wicke, Martin, Yu, Yuan, and Zhang, Xiaoqiang. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016.

Blanchard, Peva, Guerraoui, Rachid, Stainer, Julien, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, pp. 118–128, 2017.

Blum, Manuel, Floyd, Robert W, Pratt, Vaughan, Rivest, Ronald L, and Tarjan, Robert E. Time bounds for selection. *Journal of computer and system sciences*, 7(4): 448–461, 1973.

Chen, Tianqi, Li, Mu, Li, Yutian, Lin, Min, Wang, Naiyan, Wang, Minjie, Xiao, Tianjun, Xu, Bing, Zhang, Chiyuan, and Zhang, Zheng. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015.

Chen, Yudong, Su, Lili, and Xu, Jiaming. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *arXiv preprint arXiv:1705.05491*, 2017.

Cohen, Michael B, Lee, Yin Tat, Miller, Gary, Pachocki, Jakub, and Sidford, Aaron. Geometric median in nearly linear time. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 9–21. ACM, 2016.

Dean, Jeffrey, Corrado, Gregory S., Monga, Rajat, Chen, Kai, Devin, Matthieu, Le, Quoc V., Mao, Mark Z., Ranzato, Marc'Aurelio, Senior, Andrew W., Tucker, Paul A., Yang, Ke, and Ng, Andrew Y. Large scale distributed deep networks. In *NIPS*, 2012.

Harinath, Depavath, Satyanarayana, P, and Murthy, MV Ramana. A review on security issues and attacks in distributed systems. *Journal of Advances in Information Technology*, 8(1), 2017.

Ho, Qirong, Cipar, James, Cui, Henggang, Lee, Seunghak, Kim, Jin Kyu, Gibbons, Phillip B., Gibson, Garth A., Ganger, Gregory R., and Xing, Eric P. More effective distributed ml via a stale synchronous parallel parameter server. *Advances in neural information processing systems*, 2013:1223–1231, 2013.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. 2009.

Lamport, Leslie, Shostak, Robert E., and Pease, Marshall C. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4:382–401, 1982.

Li, Mu, Andersen, David G., Park, Jun Woo, Smola, Alexander J., Ahmed, Amr, Josifovski, Vanja, Long, James, Shekita, Eugene J., and Su, Bor-Yiing. Scaling distributed machine learning with the parameter server. In *OSDI*, 2014a.

Li, Mu, Andersen, David G., Smola, Alexander J., and Yu, Kai. Communication efficient distributed machine learning with the parameter server. In *NIPS*, 2014b.

Loosli, Gaëlle, Canu, Stéphane, and Bottou, Léon. Training invariant support vector machines using selective sampling. *Large scale kernel machines*, pp. 301–320, 2007.

Lynch, Nancy A. *Distributed algorithms*. Morgan Kaufmann, 1996.

McMahan, H. Brendan, Moore, Eider, Ramage, Daniel, Hampson, Seth, and y Arcas, Blaise Aguera. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

Minsker, Stanislav et al. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4):2308–2335, 2015.

Mukkamala, Mahesh Chandra and Hein, Matthias. Variants of rmsprop and adagrad with logarithmic regret bounds. In *ICML*, 2017.

Seide, Frank and Agarwal, Amit. Cntk: Microsoft's open-source deep-learning toolkit. In *KDD*, 2016.

8. Appendix

In the appendix, we introduce several useful lemmas and use them to derive the detailed proofs of the theorems in this paper.

8.1. Dimensional Byzantine Resilience

Theorem 1. *Averaging is not dimensional Byzantine resilient.*

Proof. We demonstrate a counter example. Consider the case where

$$\tilde{v}_i = \begin{cases} v_i, & \forall i \in [n-1] \\ -g - \sum_{i=1}^{n-1} v_i, & i = n, \end{cases} \tag{3}$$

where $g = \mathbb{E}[v_i], \forall i \in [n]$. Thus, the resulting aggregation is $Aggr = -g/n$. The inner product $\langle \mathbb{E}[Aggr], g \rangle$ is always negative under the Byzantine attack. Thus, SGD is not expectedly descendant, which means it will not converge to critical points. Note that in this counter example, the number of Byzantine values of each dimension is 1.

Hence, averaging is not dimensional (α, q)-Byzantine resilient with $\forall \alpha, \forall q > 0$. □

Theorem 2. *Any aggregation rule $Aggr(\{\tilde{v}_i : i \in [n]\})$ that outputs $Aggr \in \{\tilde{v}_i : i \in [n]\}$ is not dimensional Byzantine resilient.*

Proof. We demonstrate a counter example. Consider the case where the ith dimension of the ith vector v_i is manipulated by the malicious workers (e.g. multiplied by an arbitrarily large negative value), where $i \in [n]$. Thus, up to 1 value of each dimension is Byzantine. However, no matter which vector is chosen, as long as the aggregation is chosen from $\{\tilde{v}_i : i \in [n]\}$, the inner product $\langle \mathbb{E}[Aggr], g \rangle$ can be arbitrarily large negative value under the Byzantine attack. Thus, SGD is not expectedly descendant, which means it will not converge to critical points.

Hence, any aggregation rule that outputs $Aggr \in \{\tilde{v}_i : i \in [n]\}$ is not dimensional (α, q)-Byzantine resilient with $\forall \alpha, \forall q > 0$. □

8.2. Geometric Median

We use the following lemma (Minsker et al., 2015; Cohen et al., 2016) without proof to bound the geometric median.

Lemma 3. *Let $z_1, \ldots, z_n$ denote n points in a Hilbert space. Let z_* denote a $(1+\epsilon)$-approximation of their geometric median, i.e., $\sum_{i \in [n]} \|z_* - z_i\| \leq (1+\epsilon) \min_z \sum_{i \in [n]} \|z - z_i\|$ for $\epsilon \geq 0$. For any q such that $\frac{q}{n} \in (0, 1/2)$ and given $r \in \mathbb{R}$, if $\sum_{i \in [n]} \mathbf{1}_{\|z_i\| \leq r} \geq (1 - q/n)n$, then*

$$\|z_*\| \leq c_q r + \epsilon c_z,$$

where $c_q = \frac{2n-2q}{n-2q}$, $c_z = \frac{\min_z \sum_{i \in [n]} \|z - z_i\|}{n-2q}$.

Ideally, the geometric median ($\epsilon = 0$) ignores the second term ϵc_z.

Using the lemma above, we can prove the classic Byzantine resilience of geometric median.

Theorem 3. *Let $v_1, \ldots, v_n$ be any i.i.d. random d-dimensional vectors s.t. $v_i \sim G$, with $\mathbb{E}[G] = g$ and $\mathbb{E}\|G - g\|^2 = d\sigma^2$. q of $\{v_i : i \in [n]\}$ are replaced by arbitrary d-dimensional vectors $b_1, \ldots, b_q$. If $q \leq \lceil \frac{n}{2} \rceil - 1$ and $\eta_1(n,q)\sqrt{d}\sigma < \|g\|$, where $\eta_1(n,q) = \frac{2n-2q}{n-2q}\sqrt{n-q}$, then the $GeoMed$ function is classic (α_1, q)-Byzantine resilient where $0 \leq \alpha_1 < \pi/2$ is defined by $\sin \alpha_1 = \frac{\eta_1(n,q)\sqrt{d}\sigma}{\|g\|}$.*

Proof. We only need to prove that $GeoMed(\cdot)$ satisfies the two conditions of classic (α_1, q)-Byzantine resilience defined in Definition 2.

Condition (i):
Let the sequence $\{\tilde{v}_j : j \in [n]\}$ be defined as

$$\tilde{v}_j = \begin{cases} v_j, \text{for correct } j, \\ arbitrary, \text{for Byzantine } j. \end{cases}$$

Let λ denote the geometric median of $\{\tilde{v}_j : j \in [n]\}$. Thus, $z_* = \lambda - g$ is the geometric median of $\{\tilde{v}_j - g : j \in [n]\}$. Using Lemma 3, and taking $r = \max_{\text{correct } j} \|\tilde{v}_j - g\|$, under the assumption $q \leq \lceil \frac{n}{2} \rceil - 1 < n/2$, we obtain

$$\|\lambda - g\| \leq \frac{2n-2q}{n-2q} \max_{\text{correct } j} \|\tilde{v}_j - g\|.$$

Now, we can bound $\|\mathbb{E}[\lambda] - g\|^2$ as follows:

$$\begin{aligned} &\|\mathbb{E}[\lambda] - g\|^2 \\ &\leq \mathbb{E}\|\lambda - g\|^2 \quad \text{(Jensen's inequality)} \\ &\leq \mathbb{E}\left[\left(\frac{2n-2q}{n-2q}\right)^2 \max_{\text{correct } j} \|\tilde{v}_j - g\|^2\right] \\ &\leq \mathbb{E}\left[\left(\frac{2n-2q}{n-2q}\right)^2 \sum_{\text{correct } j} \|\tilde{v}_j - g\|^2\right] \\ &= \underbrace{\left(\frac{2n-2q}{n-2q}\right)^2 (n-q)}_{\eta_1^2(n,q)} d\sigma^2. \end{aligned}$$

By assumption, $\eta_1(n,q)\sqrt{d}\sigma < \|g\|$, i.e. $\mathbb{E}[\lambda]$ belongs to a ball centered at g with radius $\eta_1(n,q)\sqrt{d}\sigma$. This implies

$$\langle \mathbb{E}[\lambda], g \rangle \geq (1 - \sin^2 \alpha_1)\|g\|^2 \geq (1 - \sin \alpha_1)\|g\|^2,$$

where $\sin\alpha_1 = \eta_1(n,q)\sqrt{d}\sigma/\|g\|$.

Condition (ii):
We re-use Lemma 3 by taking $z_* = \lambda$, $z_i = \tilde{v}_i$ for $\forall i \in [n]$, and $r = \max_{\text{correct } j} \|\tilde{v}_j\|$. Thus, we have

$$\|\lambda\| \quad \leq c_q \max_{\text{correct } j} \|\tilde{v}_j\| \leq c_q \sum_{\text{correct } j} \|\tilde{v}_j\|.$$

Without loss of generality, we denote the sequence $\{\tilde{v}_j : \text{correct } j\}$ as $\{v_1, \ldots, v_{n-q}\}$. Thus, there exists a constant c_0 such that

$$\|\lambda\|^r \leq c_0 \sum_{r_1+\ldots+r_{n-q}=r} \|v_1\|^{r_1} \ldots \|v_{n-q}\|^{r_{n-q}}.$$

Since v_i's are i.i.d., we obtain that $\mathbb{E}\|\lambda\|^r$ is bounded above by a linear combination of terms of the form $\mathbb{E}\|v_1\|^{r_1} \ldots \mathbb{E}\|v_{n-q}\|^{r_{n-q}} = \mathbb{E}\|G\|^{r_1} \ldots \mathbb{E}\|G\|^{r_{n-q}}$ with $r_1 + \ldots + r_{n-q} = r$, which completes the proof of condition (ii). □

8.3. Marginal Median

We use the following lemma to bound the one-dimensional median.

Lemma 4. *For a sequence composed of q Byzantine values and $n-q$ correct values $u_1, \ldots, u_{n-q}$, if $q \leq \lceil \frac{n}{2} \rceil - 1$ (the correct value dominates the sequence), then the median value m of this sequence satisfies $m \in [\min_i u_i, \max_i u_i]$, $i \in [n]$.*

Proof. If m comes from correct values, then the result is trivial. Thus, we only need to consider the cases where m comes from Byzantine values.

If n is odd, then in the sorted sequence, there will be $\frac{n-1}{2}$ values on both sides of m. However, the number of correct values $n-q \geq \frac{n+1}{2} > \frac{n-1}{2}$. Thus, on both sides of m, there will be at least one correct value, which yields the desired result.

Furthermore, if n is even, we can re-use the same technique above to prove $m \in [\min_i u_i, \max_i u_i]$. □

Theorem 4. *Let $v_1, \ldots, v_n$ be any i.i.d. random d-dimensional vectors s.t. $v_i \sim G$, with $\mathbb{E}[G] = g$ and $\mathbb{E}\|G - g\|^2 = d\sigma^2$. For any dimension $j \in [d]$, q of $\{(v_1)_j, \ldots, (v_n)_j\}$ are replaced by arbitrary values, where $(v_i)_j$ is the jth dimension of the vector v_i. If $q \leq \lceil \frac{n}{2} \rceil - 1$ and $\eta_2(n,q)\sqrt{d}\sigma < \|g\|$, where $\eta_2(n,q) = \sqrt{n-q}$, then the $MarMed$ function is dimensional (α_2, q)-Byzantine resilient where $0 \leq \alpha_2 < \pi/2$ is defined by $\sin\alpha_2 = \frac{\eta_2(n,q)\sqrt{d}\sigma}{\|g\|}$.*

Proof. We only need to prove that $MarMed(\cdot)$ satisfies the two conditions of dimensional (α_2, q)-Byzantine resilience defined in Definition 5.

Condition (i):
Without loss of generality, we assume that $\mathbb{E}[G_i - g_i]^2 = \sigma_i^2$, $\mathbb{E}\|G - g\|^2 = \mathbb{E}\sum_{i=1}^d [G_i - g_i]^2 = \sum_{i=1}^d \sigma_i^2 = d\sigma^2$. For any dimension $j \in [d]$, let the sequence $\{(\tilde{v}_1)_j, \ldots, (\tilde{v}_n)_j\}$ be defined as

$$(\tilde{v}_i)_j = \begin{cases} (v_i)_j, \text{for correct } j, \\ arbitrary, \text{for Byzantine } j. \end{cases}$$

For the jth dimension, $j \in [d]$, the median value $\mu_j \in [\min_{\text{correct } i}(\tilde{v}_i)_j, \max_{\text{correct } i}(\tilde{v}_i)_j]$.

Thus, we have

$$\begin{aligned}
&\mathbb{E}[\mu_j - g_j]^2 \leq \mathbb{E}\left[\max_{\text{correct } i} ((\tilde{v}_i)_j - g_j)^2\right] \\
&\leq \mathbb{E}\left[\sum_{\text{correct } i} ((\tilde{v}_i)_j - g_j)^2\right] = \sum_{\text{correct } i} \mathbb{E}\left[((\tilde{v}_i)_j - g_j)^2\right] \\
&= (n-q)\mathbb{E}[G_j - g_j]^2 \quad \text{(i.i.d. over } i\text{)} \\
&= (n-q)\sigma_j^2.
\end{aligned}$$

Now, we can bound $\|\mathbb{E}[\mu] - g\|^2$ as follows:

$$\begin{aligned}
&\|\mathbb{E}[\mu] - g\|^2 \leq \mathbb{E}\|\mu - g\|^2 \quad \text{(Jensen's inequality)} \\
&= \mathbb{E}\left[\sum_{j=1}^d (\mu_j - g_j)^2\right] = \sum_{j=1}^d \mathbb{E}\left[(\mu_j - g_j)^2\right] \\
&\leq \sum_{j=1}^d (n-q)\sigma_j^2 = (n-q)\sum_{j=1}^d \sigma_j^2 = \underbrace{(n-q)}_{\eta_2^2(n,q)} d\sigma^2.
\end{aligned}$$

By assumption, $\eta_2(n,q)\sqrt{d}\sigma < \|g\|$, i.e. $\mathbb{E}[\mu]$ belongs to a ball centered at g with radius $\eta_2(n,q)\sqrt{d}\sigma$. This implies

$$\langle \mathbb{E}[\mu], g\rangle \geq (1 - \sin^2\alpha_2)\|g\|^2 \geq (1 - \sin\alpha_2)\|g\|^2,$$

where $\sin\alpha_2 = \eta_2(n,q)\sqrt{d}\sigma/\|g\|$.

Condition (ii):
By using the equivalence of norms in finite dimension, there exists a constant c_1 such that

$$\begin{aligned}
\|\mu\| &= \sqrt{\sum_{j=1}^d \mu_j^2} \leq \sqrt{\sum_{j=1}^d \max_{\text{correct } i} (\tilde{v}_i)_j^2} \\
&\leq \sqrt{\sum_{j=1}^d \sum_{\text{correct } i} (\tilde{v}_i)_j^2} = \sqrt{\sum_{\text{correct } i} \|\tilde{v}_i\|^2} \\
&\leq c_1 \sum_{\text{correct } i} \|\tilde{v}_i\|.
\end{aligned}$$

(equivalence between ℓ_2-norm and ℓ_1-norm)

Without loss of generality, we denote the sequence $\{\tilde{v}_i : \text{correct } i\}$ as $\{v_1, \ldots, v_{n-q}\}$. Thus, there exists a constant c_2 such that

$$\|\mu\|^r \leq c_2 \sum_{r_1+\ldots+r_{n-q}=r} \|v_1\|^{r_1} \ldots \|v_{n-q}\|^{r_{n-q}}.$$

Since v_i's are i.i.d., we obtain that $\mathbb{E}\|\mu\|^r$ is bounded above by a linear combination of terms of the form $\mathbb{E}\|v_1\|^{r_1} \ldots \mathbb{E}\|v_{n-q}\|^{r_{n-q}} = \mathbb{E}\|G\|^{r_1} \ldots \mathbb{E}\|G\|^{r_{n-q}}$ with $r_1 + \ldots + r_{n-q} = r$, which completes the proof of condition (ii). □

8.4. Mean around Median

The following lemma bounds the one-dimensional mean around median.

Lemma 5. *For a sequence (of scalar values) composed of q Byzantine values and $n-q$ correct values $u_1, \ldots, u_{n-q}$, if $q \leq \lceil \frac{n}{2} \rceil - 1$ (the correct value dominates the sequence), then the mean-around-median value ρ (defined in Definition 8) and the median μ (defined in Definition 7) of this sequence satisfies $|\rho - \mu| \leq \max_i |u_i - \mu|$.*

Proof. According to the definition of the mean around median ρ, it is the mean value over the top-$(n-1)$ values in the sequence, nearest to the median μ. Denote such set of nearest values as $\{w_1, \ldots, w_{n-q}\}$. If any w_i satisfies that $|w_i - \mu| > \max_i |u_i - \mu|$, then it cannot be in the set of the top-$(n-q)$ nearest values because all the $n-q$ correct values are nearer to μ ($|u_i - \mu| \leq \max_i |u_i - \mu|$). Since all w_i satisfies $|w_i - \mu| \leq \max_i |u_i - \mu|$, the average over them must also satisfies $|\frac{1}{n-q}\sum_i w_i - \mu| \leq \max_i |u_i - \mu|$. □

Theorem 5. *Let $v_1, \ldots, v_n$ be any i.i.d. random d-dimensional vectors s.t. $v_i \sim G$, with $\mathbb{E}[G] = g$ and $\mathbb{E}\|G - g\|^2 = d\sigma^2$. For any dimension $j \in [d]$, q of $\{(v_1)_j, \ldots, (v_n)_j\}$ are replaced by arbitrary values, where $(v_i)_j$ is the jth dimension of the vector v_i. If $q \leq \lceil \frac{n}{2} \rceil - 1$ and $\eta_3(n,q)\sqrt{d}\sigma < \|g\|$, where $\eta_3(n,q) = \sqrt{10(n-q)}$, then the $MeaMed$ function is dimensional (α_3, q)-Byzantine resilient where $0 \leq \alpha_3 < \pi/2$ is defined by $\sin\alpha_3 = \frac{\eta_3(n,q)\sqrt{d}\sigma}{\|g\|}$.*

Proof. We only need to prove that $MeaMed(\cdot)$ satisfies the two conditions of (α_3, q)-Byzantine resilience defined in Definition 5.

Condition (i):
Without loss of generality, we assume that $\mathbb{E}[G_i - g_i]^2 = \sigma_i^2$, $\mathbb{E}\|G - g\|^2 = \mathbb{E}\sum_{i=1}^d [G_i - g_i]^2 = \sum_{i=1}^d \sigma_i^2 = d\sigma^2$. For any dimension $j \in [d]$, let the sequence $\{(\tilde{v}_1)_j, \ldots, (\tilde{v}_n)_j\}$ be defined as

$$(\tilde{v}_i)_j = \begin{cases} (v_i)_j, \text{for correct } j, \\ arbitrary, \text{for Byzantine } j. \end{cases}$$

For the jth dimension, $j \in [d]$, using Lemma 5, we have $|\rho_j - \mu_j| \leq \max_{\text{correct } i} |(\tilde{v}_i)_j - \mu_j|$, where μ_j is the median of the jth dimension.

Thus, we have

$$\begin{aligned}
&\mathbb{E}[\rho_j - g_j]^2 \\
&\leq 2\mathbb{E}[\rho_j - \mu_j]^2 + 2\mathbb{E}[\mu_j - g_j]^2 \\
&\leq 2\mathbb{E} \max_{\text{correct } i} [(\tilde{v}_i)_j - \mu_j]^2 + 2\mathbb{E}[\mu_j - g_j]^2 \\
&\leq 4\mathbb{E} \max_{\text{correct } i} [(\tilde{v}_i)_j - g_j]^2 + 6\mathbb{E}[\mu_j - g_j]^2 \\
&\leq 10\mathbb{E}\left[\max_{\text{correct } i} ((\tilde{v}_i)_j - g_j)^2\right] \\
&\leq 10\mathbb{E}\left[\sum_{\text{correct } i} ((\tilde{v}_i)_j - g_j)^2\right] \\
&= 10 \sum_{\text{correct } i} \mathbb{E}\left[((\tilde{v}_i)_j - g_j)^2\right] \\
&= 10(n-q)\mathbb{E}[G_j - g_j]^2 \quad \text{(i.i.d. over } i\text{)} \\
&= 10(n-q)\sigma_j^2.
\end{aligned}$$

Now, we can bound $\|\mathbb{E}[\rho] - g\|^2$ as follows:

$$\begin{aligned}
&\|\mathbb{E}[\rho] - g\|^2 \leq \mathbb{E}\|\rho - g\|^2 \quad \text{(Jensen's inequality)} \\
&= \mathbb{E}\left[\sum_{j=1}^d (\rho_j - g_j)^2\right] = \sum_{j=1}^d \mathbb{E}\left[(\rho_j - g_j)^2\right] \\
&\leq \sum_{j=1}^d 10(n-q)\sigma_j^2 = 10(n-q)\sum_{j=1}^d \sigma_j^2 = \underbrace{10(n-q)}_{\eta_3^2(n,q)} d\sigma^2.
\end{aligned}$$

By assumption, $\eta_3(n,q)\sqrt{d}\sigma < \|g\|$, i.e. $\mathbb{E}[\rho]$ belongs to a ball centered at g with radius $\eta_3(n,q)\sqrt{d}\sigma$. This implies

$$\langle \mathbb{E}[\rho], g\rangle \geq (1 - \sin^2\alpha_3)\|g\|^2 \geq (1 - \sin\alpha_3)\|g\|^2,$$

where $\sin\alpha_3 = \eta_3(n,q)\sqrt{d}\sigma/\|g\|$.

Condition (ii):
By using the equivalence of norms in finite dimension, there

exists a constant c_3 such that

$$\begin{aligned}
\|\rho\| &= \sqrt{\sum_{j=1}^{d} \rho_j^2} \\
&\leq \sqrt{\sum_{j=1}^{d} 2[\rho_j - \mu_j]^2 + 2\mu_j^2} \\
&\leq \sqrt{\sum_{j=1}^{d} \max_{\text{correct } i} 2[(\tilde{v}_i)_j - \mu_j]^2 + 2\mu_j^2} \\
&\leq \sqrt{\sum_{j=1}^{d} 10 \max_{\text{correct } i} (\tilde{v}_i)_j^2} \\
&\leq \sqrt{10 \sum_{j=1}^{d} \sum_{\text{correct } i} (\tilde{v}_i)_j^2} = \sqrt{10 \sum_{\text{correct } i} \|\tilde{v}_i\|^2} \\
&\leq c_3 \sum_{\text{correct } i} \|\tilde{v}_i\|.
\end{aligned}$$

(equivalence between ℓ_2-norm and ℓ_1-norm)

Without loss of generality, we denote the sequence $\{\tilde{v}_i : \text{correct } i\}$ as $\{v_1, \ldots, v_{n-q}\}$. Thus, there exists a constant c_4 such that

$$\|\rho\|^r \leq c_4 \sum_{r_1 + \ldots + r_{n-q} = r} \|v_1\|^{r_1} \ldots \|v_{n-q}\|^{r_{n-q}}.$$

Since v_i's are i.i.d., we obtain that $\mathbb{E}\|\rho\|^r$ is bounded above by a linear combination of terms of the form $\mathbb{E}\|v_1\|^{r_1} \ldots \mathbb{E}\|v_{n-q}\|^{r_{n-q}} = \mathbb{E}\|G\|^{r_1} \ldots \mathbb{E}\|G\|^{r_{n-q}}$ with $r_1 + \ldots + r_{n-q} = r$, which completes the proof of condition (ii). □

8.5. Experimental Details

In Table 8.5 and 8.5, we show the detailed network structures of the MLP and CNN used in our experiments.

Table 2. MLP Summary

Layer (type)	Parameters	Previous Layer
flatten(Flatten)	null	data
fc1(FullyConnected)	#output=128	flatten
relu1(Activation)	null	fc1
fc2(FullyConnected)	#output=128	relu1
relu2(Activation)	null	fc2
fc3(FullyConnected)	#output=10	relu2
softmax(SoftmaxOutput)	null	fc3

Table 3. CNN Summary

Layer (type)	Parameters	Previous Layer
conv1(Convolution)	channels=32, kernel_size=3, padding=1	data
activation1(Activation)	null	conv1
conv2(Convolution)	channels=32, kernel_size=3, padding=1	activation1
activation2(Activation)	null	conv2
pooling1(Pooling)	pool_size=2	activation2
dropout1(Dropout)	probability=0.2	pooling1
conv3(Convolution)	channels=64, kernel_size=3, padding=1	dropout1
activation2(Activation)	null	conv3
conv4(Convolution)	channels=64, kernel_size=3, padding=1	activation2
activation4(Activation)	null	conv4
pooling2(Pooling)	pool_size=2	activation4
dropout2(Dropout)	probability=0.2	pooling2
flatten1(Flatten)	null	dropout2
fc1(FullyConnected)	#output=512	flatten1
activation5(Activation)	null	fc1
dropout3(Dropout)	probability=0.2	activation5
fc2(FullyConnected)	#output=512	dropout3
activation6(Activation)	null	fc2
dropout4(Dropout)	probability=0.2	activation6
fc3(FullyConnected)	#output=10	dropout4
softmax(SoftmaxOutput)	null	fc3

8.6. Additional Experiments

In this section, we illustrate the additional empirical results.

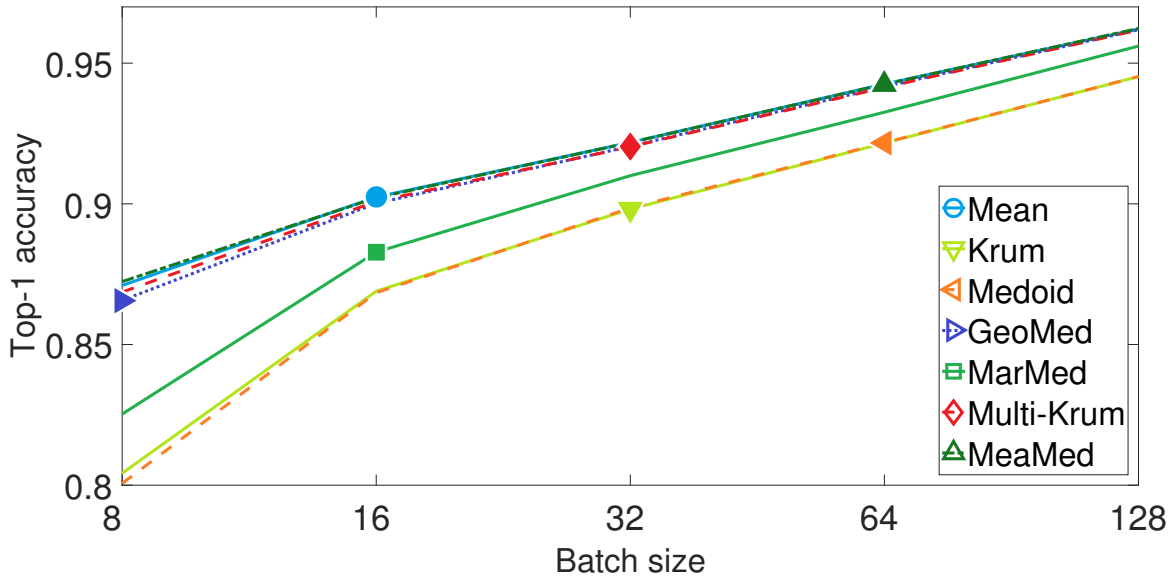
In Figure 9, we illustrate the top-1 accuracy of MLP on MNIST when batch-size varies, without Byzantine failures. The learning rate is

$$\gamma = \frac{0.1 \times batchsize}{32}.$$

The results show that when there is no Byzantine failures, *GeoMed*, *Multi-Krum*, and *MeaMed* performs just like *Mean*. *MarMed* has slightly slower convergence. *Krum* and *Medoid* are the slowest. The gap is narrowed when the batch size increases.

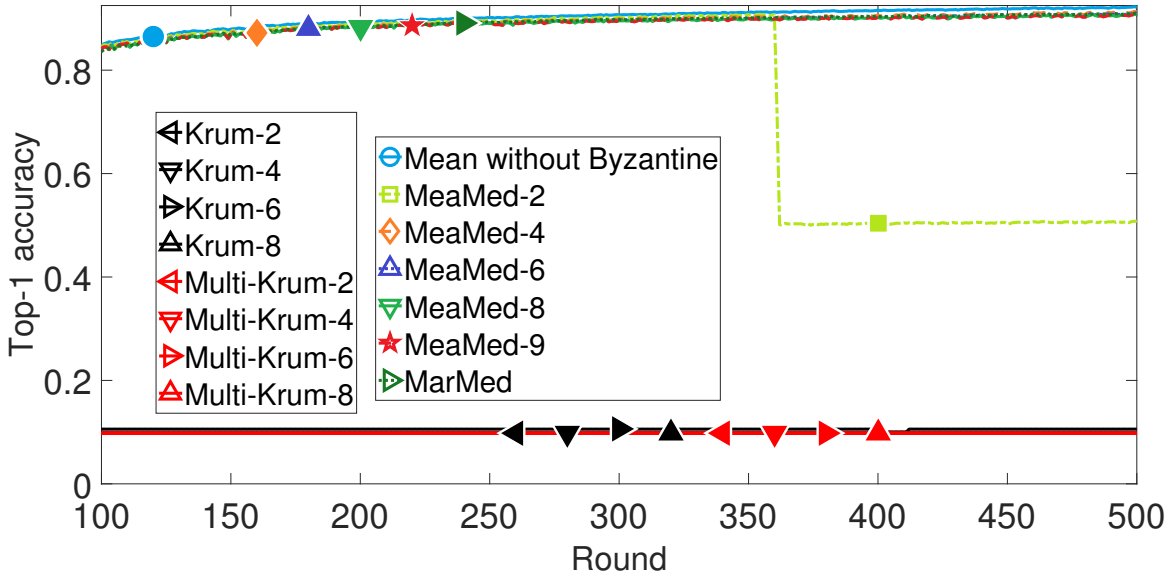
In Figure 10, we illustrate the top-1 accuracy of MLP on MNIST with gambler attack, when the estimated q varies for *Krum*, *Multi-Krum*, and *MeaMed*. *Mean* without Byzantine failures and *MarMed* are used as baselines. No matter what q we use, the Krum-based algorithms always crash. For *MeaMed*, when the estimated q is too small (e.g., $q = 2$), it will also crash. In most cases, *MeaMed* performs well. The performance of *MeaMed* is similar to *MarMed*.

We illustrate all the experimental results of CNN on CIFAR10 additional to Section 5. For completeness, we also illustrate the experimental results of MLP on MNIST. The results are shown in Figure 11-20. In general, all the experiments of CNN on CIFAR10 show similar results with the experiments of MLP on MNIST.

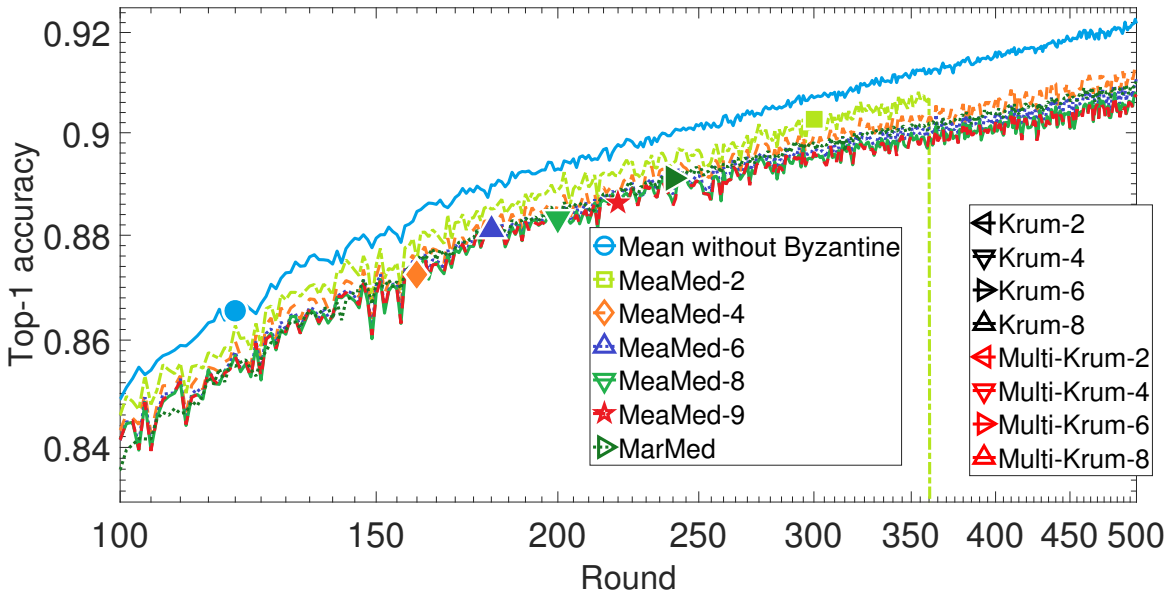


(a) MLP on MNIST without Byzantine with different batch sizes

Figure 9. Top-1 accuracy of MLP on MNIST without Byzantine failures, when batch size varies. The learning rate is $\gamma = \frac{0.1 \times batchsize}{32}$.

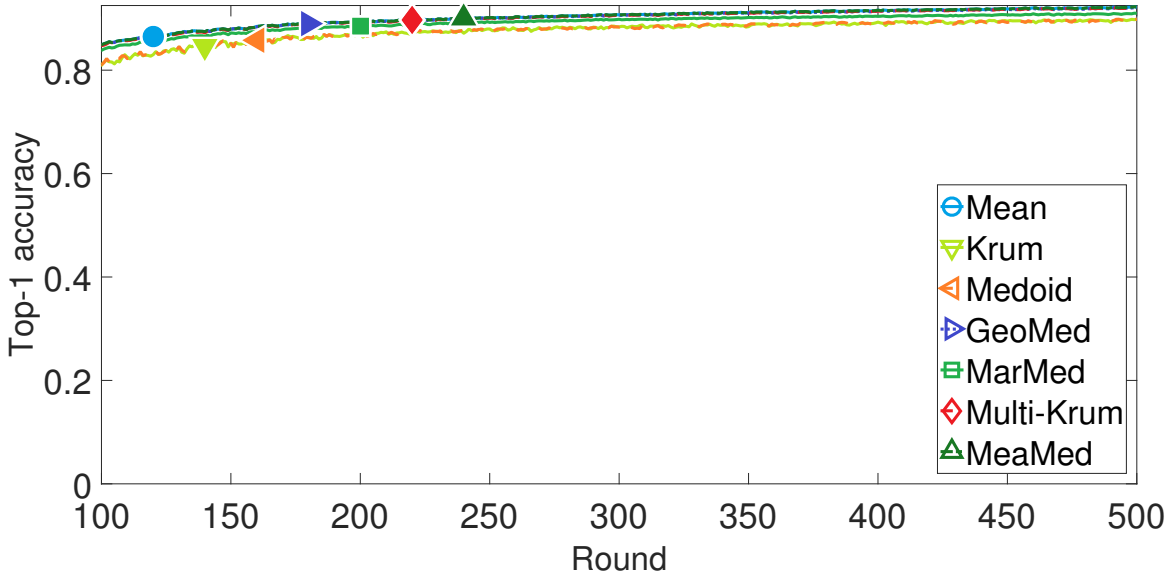


(a) MLP on MNIST with gambler

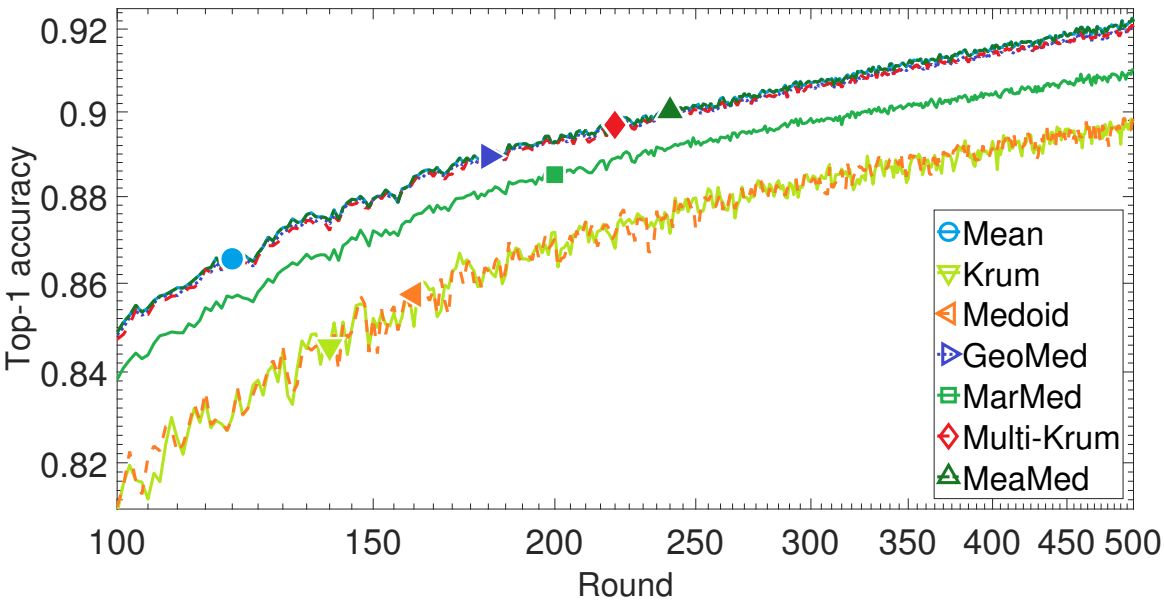


(b) MLP on MNIST with gambler (zoomed)

Figure 10. Top-1 accuracy of MLP on MNIST with gambler attack, when q varies

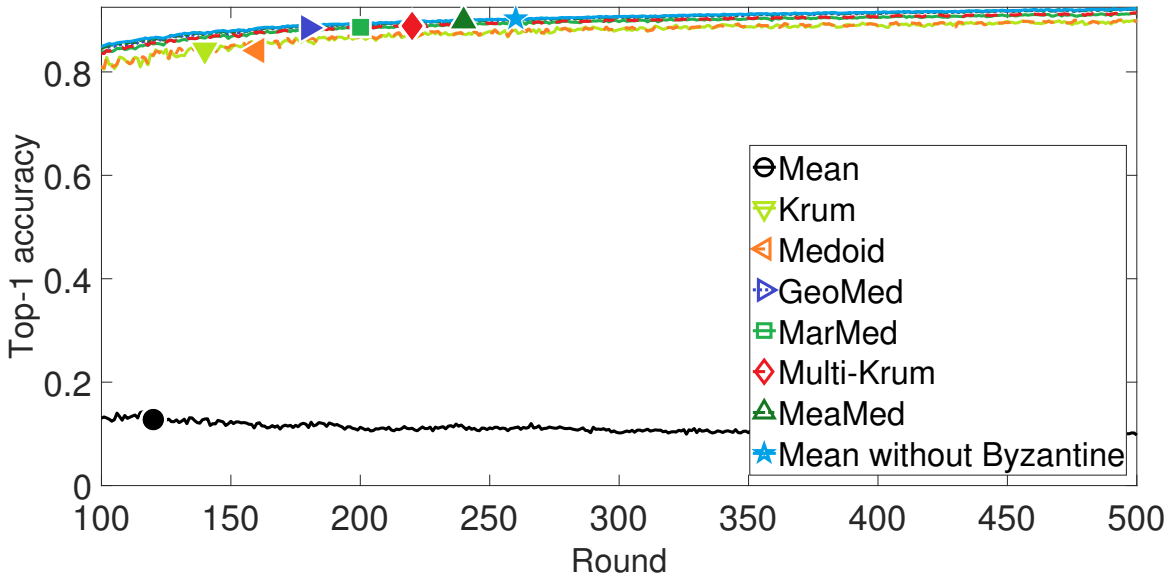


(a) MLP on MNIST without Byzantine

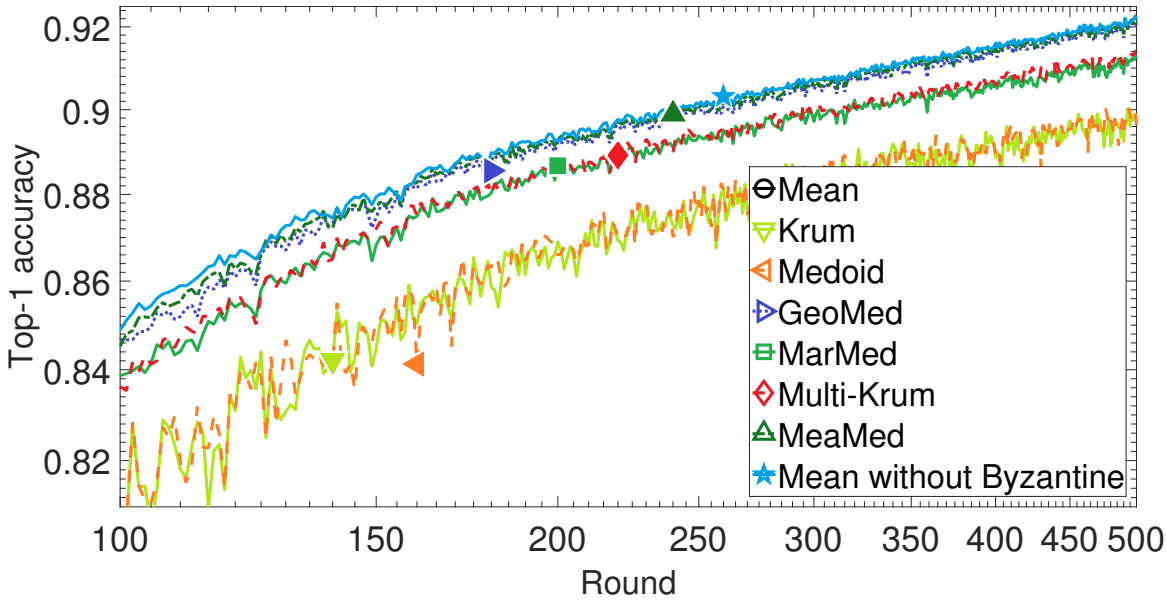


(b) MLP on MNIST without Byzantine (zoomed)

Figure 11. Top-1 accuracy of MLP on MNIST without Byzantine failures.

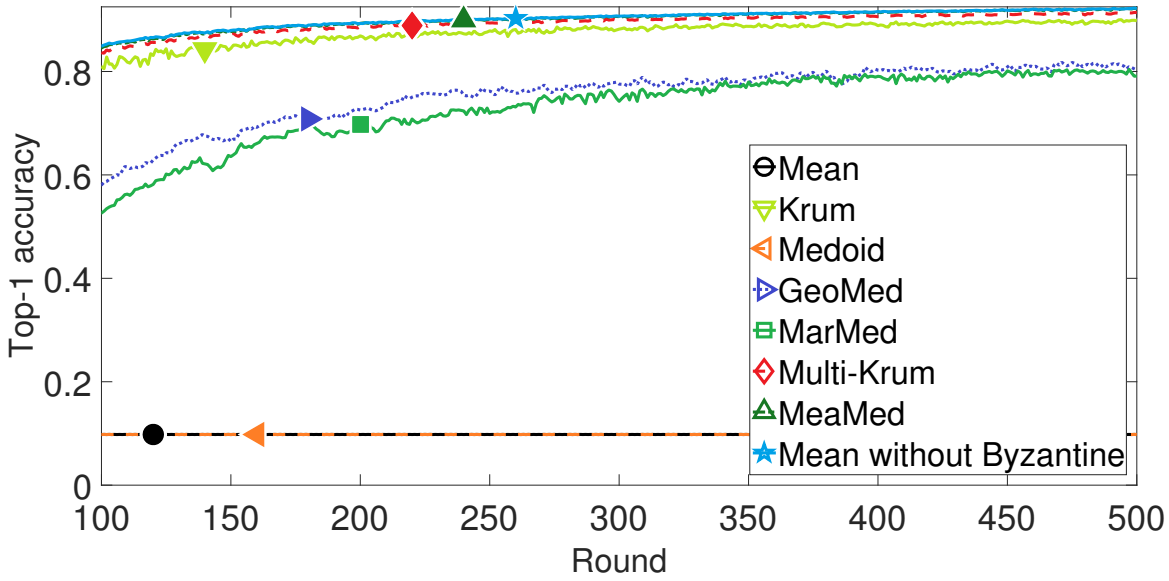


(a) MLP on MNIST with Gaussian

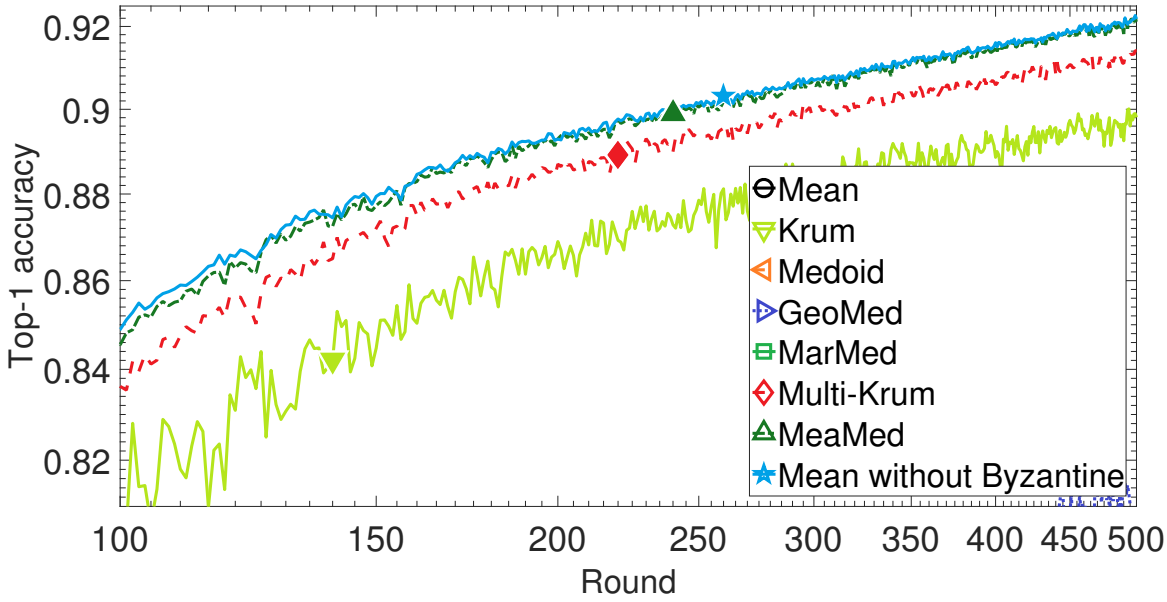


(b) MLP on MNIST with Gaussian (zoomed)

Figure 12. Top-1 accuracy of MLP on MNIST with Gaussian Attack. 6 out of 20 gradient vectors are replaced by i.i.d. random vectors drawn from a Gaussian distribution with 0 mean and 200 standard deviation.

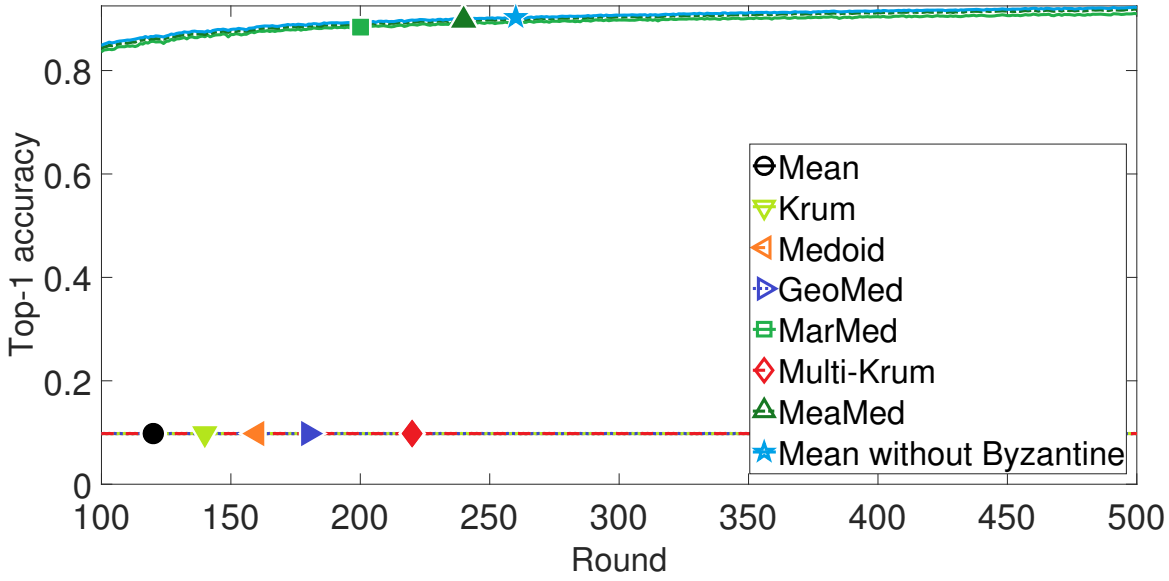


(a) MLP on MNIST with omniscient

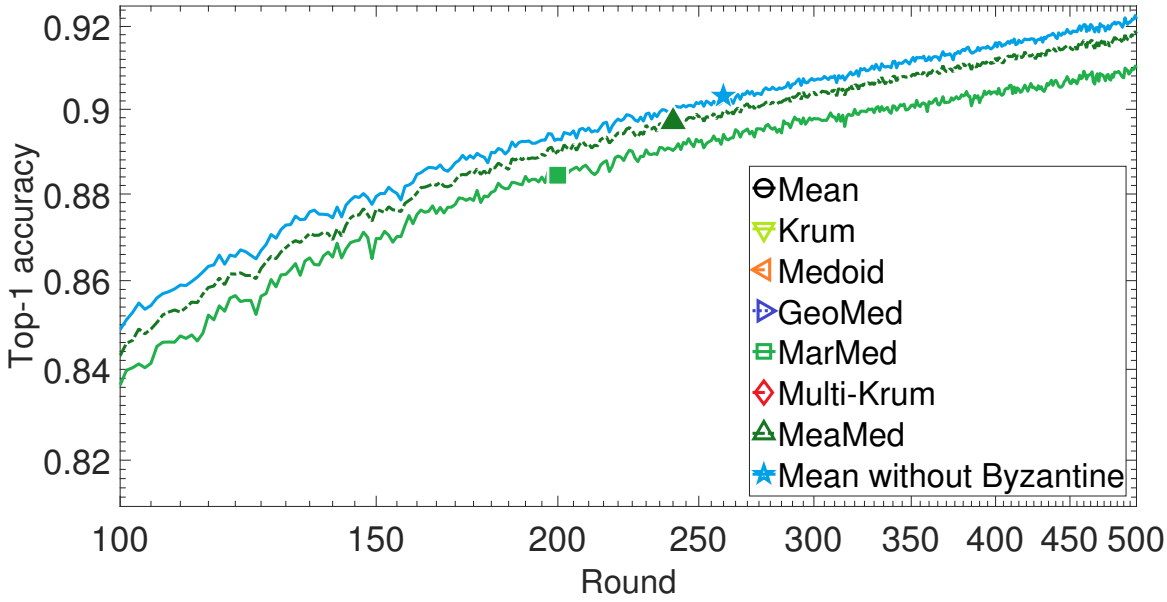


(b) MLP on MNIST with omniscient (zoomed)

Figure 13. Top-1 accuracy of MLP on MNIST with Omniscient Attack. 6 out of 20 gradient vectors are replaced by the negative sum of all the correct gradients, scaled by a large constant (1e20 in the experiments).

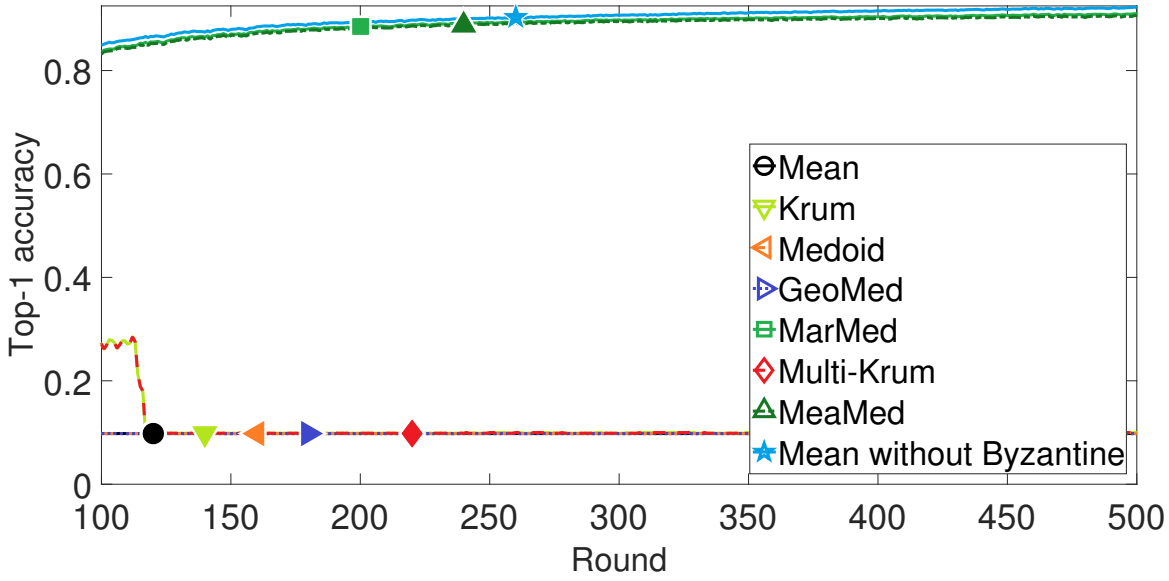


(a) MLP on MNIST with bit-flip

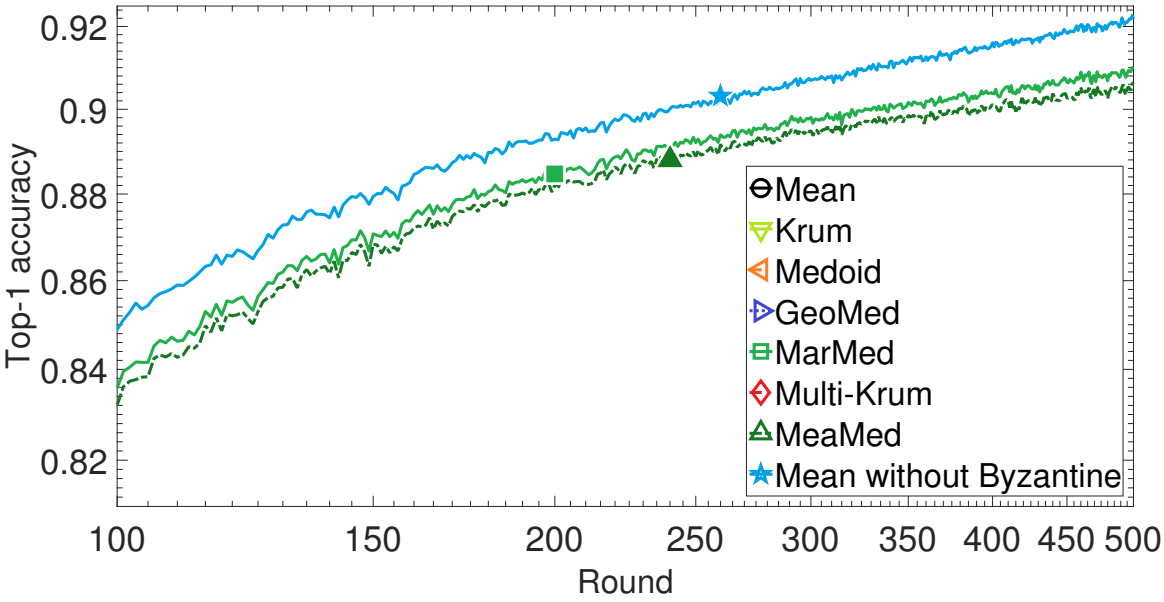


(b) MLP on MNIST with bit-flip (zoomed)

Figure 14. Top-1 accuracy of MLP on MNIST with Bit-flip Attack. For the first 1000 dimensions, 1 of the 20 floating numbers is manipulated by flipping the 22th, 30th, 31th and 32th bits.



(a) MLP on MNIST with gambler



(b) MLP on MNIST with gambler (zoomed)

Figure 15. Top-1 accuracy of MLP on MNIST with gambler attack. The parameters are evenly assigned to 20 servers. For one single server, any received value is multiplied by $-1e20$ with probability 0.05%.

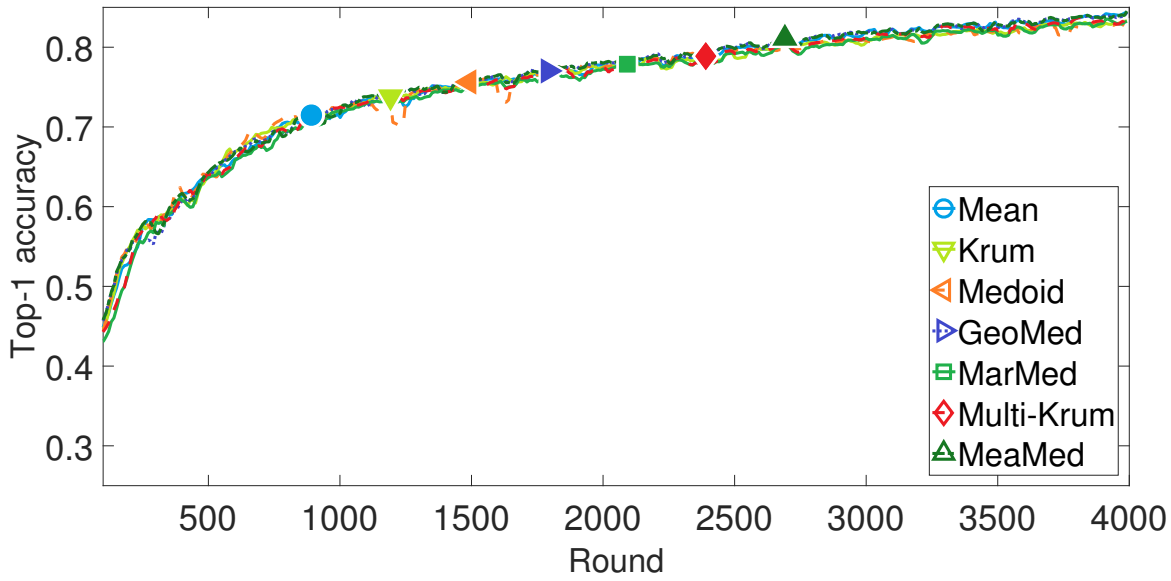


Figure 16. Top-3 Accuracy of CNN VS. # rounds evaluated on CIFAR10 without Byzantine failures

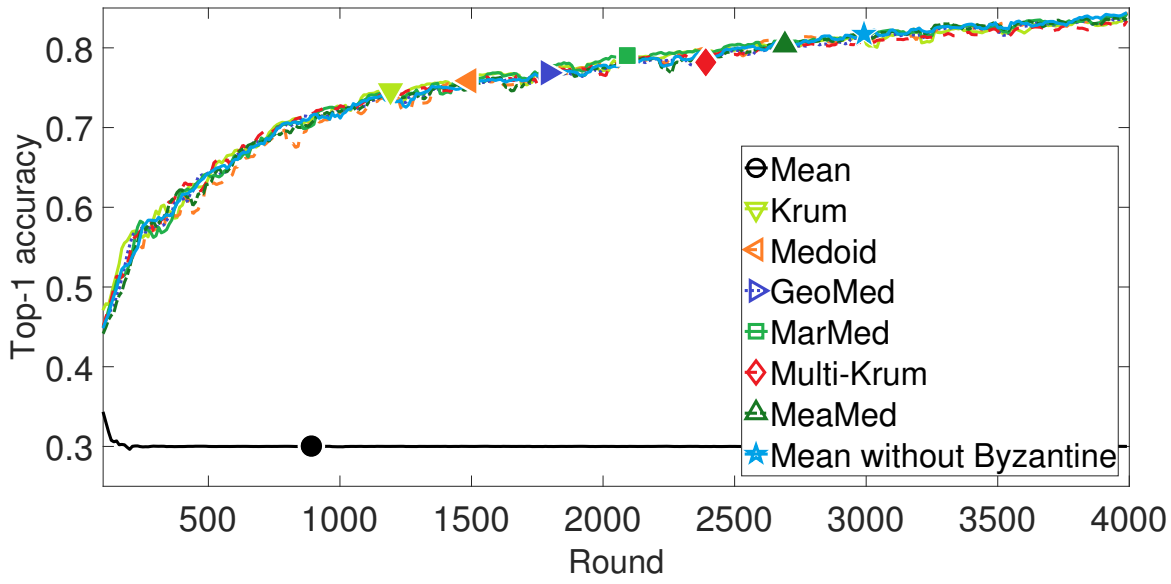


Figure 17. Top-3 Accuracy of CNN VS. # rounds evaluated on CIFAR10 with Gaussian Attack

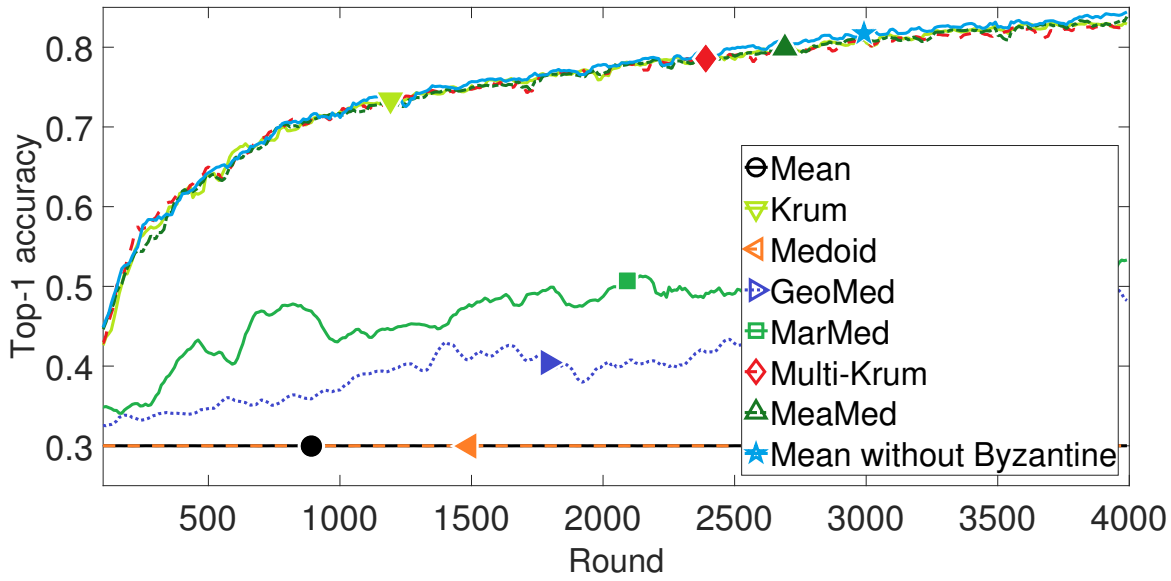


Figure 18. Top-3 Accuracy of CNN VS. # rounds evaluated on CIFAR10 with Omniscient Attack

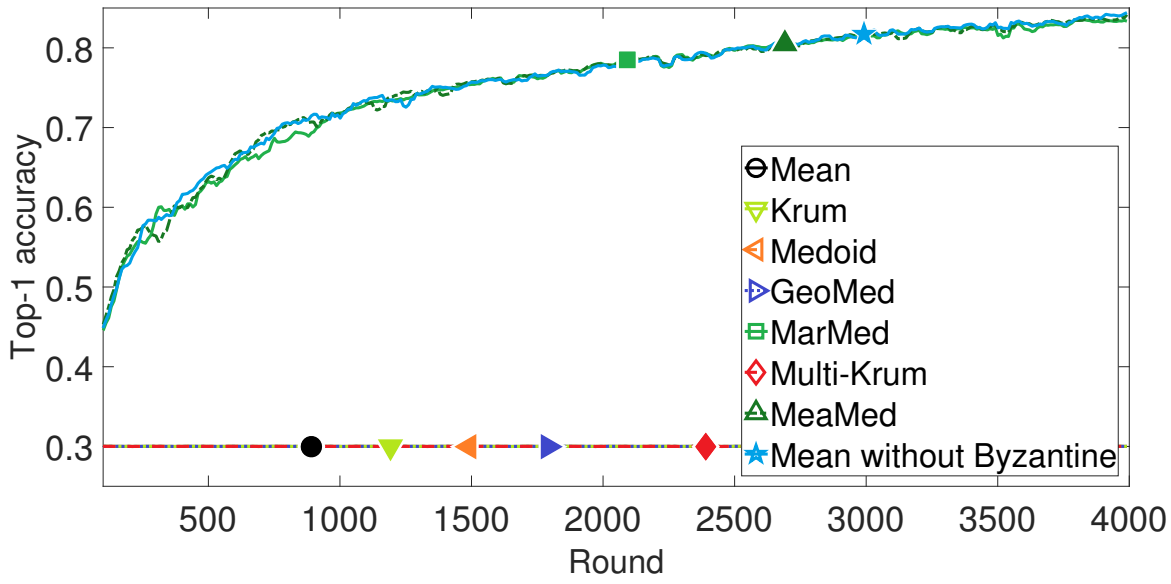


Figure 19. Top-3 Accuracy of CNN VS. # rounds evaluated on CIFAR10 with Bit-flip Attack

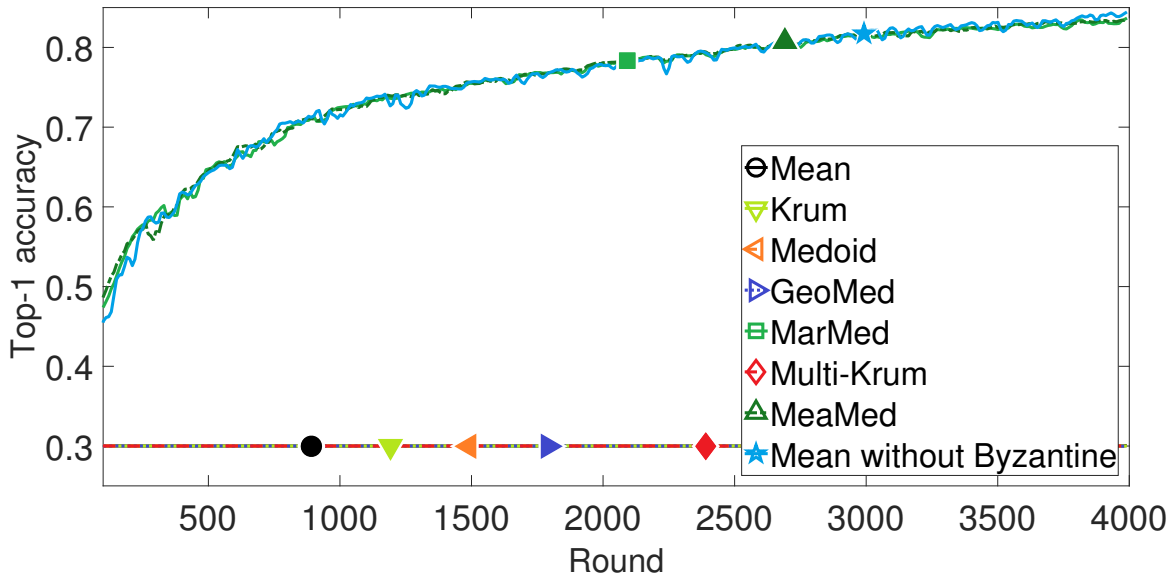


Figure 20. Top-3 Accuracy of CNN VS. # rounds evaluated on CIFAR10 with Gambler attack.