Byzantine-resilient Federated Learning via Gradient Memorization

Chen Chen[1*], **Lingjuan Lyu**[2], **Yuchen Liu**[1], **Fangzhao Wu**[3], **Chaochao Chen**[1], **Gang Chen**[1]

[1]College of Computer Science and Technology, Zhejiang University, China
[2]Sony AI, Japan
[3]MSRA, China

Abstract

Federated learning (FL) provides a privacy-aware learning framework by enabling a multitude of participants to jointly construct models without collecting their private training data. However, federated learning has exhibited vulnerabilities to *Byzantine attacks*. Many existing methods defend against such Byzantine attacks by monitoring the gradients of clients in the current round, i.e., *gradients in one round*. Recent works have demonstrated that such naive defend methods can hardly achieve satisfying performance. Defenses based on one-round gradients could be compromised by adding a small well-crafted bias to the benign gradients, due to the high variance of one-round (benign) gradients. To address this problem, we propose a new Average of Gradients (AG) framework, which detects Byzantine attacks with *the average of multi-round gradients* (i.e., gradients across multiple rounds). We theoretically show that our AG framework leads to lower variance of the benign gradients, and thus can reduce the effects of Byzantine attacks. Experiments on various real-world datasets verify the efficacy of our AG framework.

1 Introduction

Deep neural networks (DNNs) have demonstrated remarkable success in various machine learning applications (He et al. 2016; Chen et al. 2019; Wang et al. 2019). In conventional cloud-centric methods (Kantarci and Mouftah 2014), all clients first upload their data to a (central) server, and then the server trains DNN models on the collected data. However, such methods require centralized storage of clients' private data, which raises serious privacy concerns (Shokri and Shmatikov 2015; He et al. 2020). In particular, in privacy-sensitive applications such as biomedical (Buch, Ahmed, and Maruthappu 2018) and financial domains (Abbe, Khandani, and Lo 2012), the server is not allowed to access clients' sensitive data.

To make DNN models compliant with privacy regulations, e.g., general data protection regulation (GDPR) (Voigt and Von dem Bussche 2017), while preserving high-quality model prediction ability, federated learning (FL) has attracted significant attention in recent years (McMahan et al. 2017; Yang et al. 2019; Lyu, Yu, and Yang 2020; Tian et al. 2022; Wu et al. 2020, 2021; Lyu et al. 2020). In FL systems, clients train their local models on their own local data and upload their local gradients (instead of the private data) to the server for (secure) aggregation. Since local training data never leave the clients, FL provides a privacy-aware solution for scenarios where data is sensitive.

However, due to the distributed data storage, FL systems become vulnerable to *Byzantine attacks* (Xie, Koyejo, and Gupta 2020; Baruch, Baruch, and Goldberg 2019; Lyu et al. 2022). In FL systems, the server is not allowed to access clients' private data and therefore cannot directly monitor their behaviors. A malicious party can easily create a small number of Byzantine clients, i.e., malicious clients, to bias the model predictions (Baruch, Baruch, and Goldberg 2019). Different from the benign gradients, Byzantine clients can upload crafted Byzantine gradients to poison the model, thus degrading model utility. Therefore, developing a FL system that is robust against Byzantine attacks is of paramount importance.

A body of works have tried to defend against such Byzantine attacks (Shen, Tople, and Saxena 2016; Blanchard et al. 2017; Bernstein et al. 2019; Yin et al. 2018). Most existing defense methods in FL detect Byzantine attacks by monitoring clients' abnormal behaviors. For example, Krum (Blanchard et al. 2017) aggregated Byzantine gradients that are close to each other, AUROR (Shen, Tople, and Saxena 2016) detected Byzantine gradients with cluster-based methods.

However, most existing defense methods detect Byzantine attacks with *one-round gradients* (the gradients of clients in the current round), which can hardly achieve satisfying performance. Recent attacks (Baruch, Baruch, and Goldberg 2019; Xie, Koyejo, and Gupta 2020) have shown the possibility of compromising the existing defenses and attacking the FL systems. These attacks assume the variance of the benign gradients is large enough, and the smallest and the largest benign gradients are far away from each other. In such scenarios, Byzantine clients can upload well-crafted Byzantine gradients which are between the smallest and the largest benign gradients. As a result, the Byzantine clients can compromise existing defenses and modify the optimal gradients. The main reason for the success of these attacks is that the one-round benign gradients have a large variance, and such a large variance leads to the failure of existing defenses (Xie, Koyejo, and Gupta 2020).

[*]This work was done during an internship at Sony AI.
This work was accepted at AAAI 2022 workshop on Trustable, Verifiable and Auditable Federated Learning (FL-AAAI-22).

To address this problem, in this paper, we propose a new Average of Gradients (AG) framework. Instead of one-round gradients, we argue to use the average of multi-round gradients (i.e., gradients across multiple rounds) for detection. In particular, in each round t, besides the gradients uploaded in t-th round, we also utilize the gradients uploaded in the previous $t-1$ rounds for detection. We argue that the gradients in the previous rounds contain the information of the clients, thus can help the defense. Additionally, we theoretically prove that using *multi-round gradients* can alleviate the high variance issue of one-round gradients and lead to a more robust defense against Byzantine attacks.

We summarize our main contributions as follows.

- We propose a novel Average of Gradients (AG) framework, which utilizes the average of multi-round gradients to detect Byzantine attacks. We show that AG framework can effectively reduce the variance of one-round gradients with theoretical guarantee, thus providing a more robust defense against Byzantine attacks.
- Our AG framework is a compatible approach, which can be combined with most existing defenses such as Krum (Blanchard et al. 2017) and AUROR (Shen, Tople, and Saxena 2016). Our AG framework can reduce the variance of the original defenses and improve the effectiveness of the defenses.
- Experiments on various real-world datasets and Byzantine attacks corroborate the efficacy of our AG framework.

2 Notations and background

2.1 Notations

We use bold lower-case letters such as $\mathbf{m}$ to represent vectors, lower-case letters such as m to represent scalars, upper-case letters such as U to represent distributions, and upper-case curlicue letters such as $\mathcal{S}$ to represent sets. Aggregated vectors are denoted by a line over vectors such as $\overline{\mathbf{m}}$. Byzantine vectors are denoted by a tilde over vectors such as $\widetilde{\mathbf{m}}$. $\|\mathbf{m}\|$ denotes the Euclidean norm of $\mathbf{m}$. $|\mathcal{S}|$ is the cardinality of set $\mathcal{S}$. Gradients are denoted by $\mathbf{g}$. Model parameters are denoted by $\boldsymbol{\theta}$. We use superscripts to denote rounds, e.g., $\overline{\mathbf{g}}^t$ is the aggregated gradient in round t. We use subscripts to denote client indices, e.g., $\mathbf{g}_i^t$ is the gradient of client i in round t.

2.2 Federated learning

A federated learning (FL) system consists of a (central) server and m clients (McMahan et al. 2017). We use $\mathcal{D}_i$ to denote the data of the i-th client. In each (communication) round, client i trains the model with its own data $\mathcal{D}_i$, computes local gradients $\mathbf{g}_i$, and uploads the gradients $\mathbf{g}_i$ to the server. Then the server computes the aggregated gradients $\overline{\mathbf{g}}$ with the gradients of the clients. For fair comparison with previous defenses, we follow the setting of (Xie, Koyejo, and Gupta 2020) by assuming the data of all clients are independent and identically distributed (IID), and each client has the same number of data, i.e., $|\mathcal{D}_1| = \cdots = |\mathcal{D}_m|$. The algorithm of a FL system with different defenses is shown in Algorithm 1.

Algorithm 1: Training of a FL system with different defenses

Input: Learning rate η, number of client m, clients' datasets $\mathcal{D}_1, \ldots, \mathcal{D}_m$, number of training rounds T, and filter function $F()$.

Output: Trained model parameter $\boldsymbol{\theta}^T$

procedure
  Initialize $\boldsymbol{\theta}^0$
  for each round t = 1, 2, ..., T **do**
    for each client $i = 1, \ldots, m$ **do in parallel**
      Compute $\mathbf{g}_i^t$ with dataset $\mathcal{D}_i$
    end for
    Option **I** (no defense):
      $\overline{\mathbf{g}}^t \leftarrow \frac{1}{m}\sum_{i=1}^m \mathbf{g}_i^t$
    Option **II** (defense with one-round gradients):
      $\mathcal{S}_b \leftarrow F(\mathbf{g}_1^t, \cdots, \mathbf{g}_m^t)$
      $\overline{\mathbf{g}}^t \leftarrow \frac{1}{|\mathcal{S}_b|}\sum_{i\in\mathcal{S}_b} \mathbf{g}_i^t$
    Option **III** (our AG framework (Section 3)):
      for $i = 1, \ldots, m$ **do**
        $\mathbf{g}_{i,avg}^t \leftarrow \frac{t-1}{t}\mathbf{g}_{i,avg}^{t-1} + \frac{1}{t}\mathbf{g}_i^t$
      end for
      $\mathcal{S}_b \leftarrow F(\mathbf{g}_{1,avg}^t, \ldots, \mathbf{g}_{m,avg}^t)$
      $\overline{\mathbf{g}}^t \leftarrow \frac{1}{|\mathcal{S}_b|}\sum_{i\in\mathcal{S}_b} \mathbf{g}_i^t$
    $\boldsymbol{\theta}^t \leftarrow \boldsymbol{\theta}^{t-1} - \eta\overline{\mathbf{g}}^t$
  end for
end procedure

Client optimization First, we show how the clients train their models locally. Consider the following problem of minimizing an objective function:

$$L(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\ell(\boldsymbol{\theta}; \mathbf{x})],$$

where $\ell()$ is the loss function, $\boldsymbol{\theta}$ is the model parameter, and $\mathcal{D}$ is the whole dataset. We can easily compute the gradients of the above problem as follows.

$$\mathbf{g}^t = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} \partial\ell(\boldsymbol{\theta}^{t-1}; \mathbf{x})/\partial\boldsymbol{\theta}^{t-1}. \tag{1}$$

We call $\mathbf{g}^t$ the optimal gradients in round t.

However, in FL, the data is partitioned into m clients, and each client can only compute the local gradient with its own dataset. In particular, in round t, client i computes its local gradient as follows.

$$\mathbf{g}_i^t = \frac{1}{n_i}\sum_{\mathbf{x}\in\mathcal{D}_i} \partial\ell(\boldsymbol{\theta}^{t-1}; \mathbf{x})/\partial\boldsymbol{\theta}^{t-1}, \tag{2}$$

where $n_i = |\mathcal{D}_i|$ is the number of client i's local data, $\mathbf{g}_i^t$ are the local gradients of client i in round t, and $\boldsymbol{\theta}^{t-1}$ are the aggregated model parameters in round $t-1$, i.e., the initialized model parameters in round t. Note that in real-world scenarios, we use stochastic gradient descent (SGD) to approximate Eq. (2). We further discuss their differences in Section 3.2. Then, client i uploads $\mathbf{g}_i^t$ to the server for aggregation.

Server aggregation After receiving the uploaded gradients from the m clients, the server computes the aggregated gradients and the model parameters.

Specifically, in round t, the server aggregates the uploaded gradients according to the data size of each client (McMahan et al. 2017):

$$\overline{\mathbf{g}}^t = \sum_{i=1}^{m} \frac{n_i}{n} \mathbf{g}_i^t, \tag{3}$$

where $\overline{\mathbf{g}}^t$ are the aggregated gradients in round t, $n = \sum_{i=1}^{m} n_i$ is the total number of data points across all clients. Note that we follow the previous work (Xie, Koyejo, and Gupta 2020) to suppose all clients have the same number of data. Thus, the gradient aggregation can be simplified as:

$$\overline{\mathbf{g}}^t = \frac{1}{m} \sum_{i=1}^{m} \mathbf{g}_i^t. \tag{4}$$

The aggregation procedure is shown in Option **I** of Algorithm 1. Afterwards, the server computes the aggregated model parameter as follows.

$$\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1} - \eta \overline{\mathbf{g}}^t, \tag{5}$$

where $\boldsymbol{\theta}^t$ are the model parameters in round t, η is the learning rate. Afterward, the server distributes $\boldsymbol{\theta}^t$ to the clients for training in the next round.

Byzantine attacks in FL Nevertheless, in real-world applications, not all clients are benign, i.e., there are Byzantine clients in the FL systems (Blanchard et al. 2017). Since the server cannot access the clients' private data, a malicious party can easily create a small number of Byzantine clients to attack the FL system. Benign clients always upload the gradients computed with their own datasets honestly, while Byzantine clients can upload arbitrary gradients to bias the model. For the gradients uploaded by benign clients, we call them *benign gradients*. Similarly, for the gradients uploaded by Byzantine clients, we call them *Byzantine gradients*.

We suppose Byzantine clients conduct untargeted attacks, i.e., they aim to reduce the overall performance on the main task of FL (e.g., accuracy on classification tasks). To guarantee the applicability of the defenses, we consider a stronger attack scenario where the Byzantine clients have access to the benign gradients of all benign clients before conducting the attacks. Moreover, Byzantine clients may cooperate with each other.

If a FL system directly computes the aggregated gradients by averaging (by following Eq. (4)), the FL system can be easily attacked by the Byzantine clients (Blanchard et al. 2017). For example, suppose there are $m-1$ benign clients and 1 Byzantine client. The ground truth aggregated gradients are the average of all benign clients' gradients, formally, $\overline{\mathbf{g}}^t = \frac{1}{m-1} \sum_{i=1}^{m-1} \mathbf{g}_i^t$ in round t. A Byzantine client j can make the model train in an opposite direction by uploading Byzantine gradients $\widetilde{\mathbf{g}}_j^t = -\frac{2m-1}{m-1} \sum_{i=1}^{m-1} \mathbf{g}_i^t$. As a result, the aggregated gradients computed by the server (according to Eq. (4)) are $\overline{\mathbf{g}}^t = \frac{1}{m} \left(\sum_{i=1}^{m-1} \mathbf{g}_i^t + \widetilde{\mathbf{g}}_j^t \right) = -\frac{1}{m-1} \sum_{i=1}^{m-1} \mathbf{g}_i^t$, and such aggregated gradients make the global model unable to converge and adversely hurt the FL system. Therefore, developing defense methods against Byzantine attacks in FL systems is an urgent need.

Defenses in FL We first define the defense in FL. Since the uploaded gradients are not guaranteed benign, the server needs to filter out the Byzantine gradients before aggregation. In particular, in round t, after receiving the gradients, the server uses a filter function $F()$ to select the benign gradients with one-round gradients:

$$\mathcal{S}_b = F(\mathbf{g}_1^t, \cdots, \mathbf{g}_m^t), \tag{6}$$

where $\mathcal{S}_b$ is the set of benign clients chosen by the filter function. We can choose any defense methods as the filter function, e.g., Krum (Blanchard et al. 2017), AUROR (Shen, Tople, and Saxena 2016), etc. Afterward, the server aggregates all the gradients of clients in $\mathcal{S}$ as follows,

$$\overline{\mathbf{g}}^t = \frac{1}{|\mathcal{S}_b|} \sum_{i \in \mathcal{S}_b} \mathbf{g}_i^t. \tag{7}$$

The Byzantine gradients filtering process with one-round gradients is shown in Option **II** of Algorithm 1.

Second, we briefly introduce two state-of-the-art defense methods, i.e., Krum (Blanchard et al. 2017) and AUROR (Shen, Tople, and Saxena 2016).

Suppose there are m clients in a FL system, $\widetilde{m}$ of the clients are Byzantine, the other $m - \widetilde{m}$ clients are benign. Krum precluded the clients' gradients that are too far away. Specifically, for each client i in round t, Krum defined a score $s(i) = \sum_{i \to j} \left\| \mathbf{g}_i^t - \mathbf{g}_j^t \right\|$, where $i \to j$ denotes the indices of the $m - \widetilde{m} - 2$ nearest gradients of $\mathbf{g}_i^t$. Then, Krum kept the gradients of the client that has the minimum score and removed all other clients' gradients, i.e., $\mathcal{S}_b = \{i_*\}$, where $i_* = \arg\min_i s(i)$. Last, Krum assigned the aggregated gradients with the gradients of i_*, i.e., $\overline{\mathbf{g}}^t = \mathbf{g}_{i_*}^t$. Blanchard et al. also proposed a variant of Krum, namely MultiKrum. Instead of keeping the gradients of a single client, MultiKrum aggregated gradients of $n - \widetilde{n}$ clients that have the minimum scores.

AUROR divided all the clients $\{1, \ldots, m\}$ into 2 clusters: $\mathcal{S}_b = \{s_1, \ldots, s_{m_1}\}$ and $\mathcal{L} = \{l_1, \ldots, l_{m_2}\}$, where $s_i, l_j \in \{1, \ldots, m\}$ are the indices of clients, m_1 and m_2 are the number of clients in each cluster that satisfy $m_1 + m_2 = m$ and $m_1 \geq m_2$. Then, AUROR removed all the clients in $\mathcal{L}$ and aggregated the gradients of clients in $\mathcal{S}_b$ (Eq. (7)).

However, we argue that most existing defense methods cannot achieve satisfying defense performance against Byzantine attacks, due to the fact that they only consider *one-round gradients*, i.e., gradients in one round. Recent studies (Baruch, Baruch, and Goldberg 2019; Xie, Koyejo, and Gupta 2020) showed that they can compromise the existing defenses and launch effective attacks by adding a small bias to the benign gradients in each round. Xie, Koyejo, and Gupta claimed that as long as the variance of the benign gradients is high, they can successfully attack the server and modify the aggregated gradients. In particular, when the variance is high, the smallest benign gradients and the largest benign gradients are far away from each other. The Byzantine clients can

always find Byzantine gradients between the smallest and the largest gradients that can attack the global model without being detected.

3 Average of Gradients (AG) framework

In this section, we focus on detecting Byzantine gradients in FL systems, i.e., the filter function $F()$ in Eq. (6). We first introduce our proposed Average of Gradients (AG) framework. Then, we theoretically prove that our AG framework can reduce the variance of benign gradients.

3.1 Detection with AG framework

As discussed in Section 2.2, recent attacks (Baruch, Baruch, and Goldberg 2019; Xie, Koyejo, and Gupta 2020) can always compromise the defenses which use one-round gradients for detection. To this end, we propose a new Average of Gradients (AG) framework, which utilizes *multi-round gradients* (i.e., gradients across multiple rounds) to detect Byzantine attacks. Our detection procedure with multi-round gradients is shown in Option **III** of Algorithm 1. Specifically, in each round t, the server first averages the gradients of each client from the first round till the current round t:

$$\mathbf{g}_{i,avg}^t = \frac{1}{t}\sum_{k=1}^{t}\mathbf{g}_i^k, \tag{8}$$

where $i = 1, \ldots, m$ are the indices of clients, $\mathbf{g}_{i,avg}^t$ is the average of gradients for the i-th client in round t. To accelerate the computation, we can calculate the average of gradients in round t by:

$$\mathbf{g}_{i,avg}^t = \frac{t-1}{t}\mathbf{g}_{i,avg}^{t-1} + \frac{1}{t}\mathbf{g}_i^t. \tag{9}$$

Then, the server detects Byzantine attacks with the average of gradients:

$$\mathcal{S}_b = F(\mathbf{g}_{1,avg}^t, \ldots, \mathbf{g}_{m,avg}^t). \tag{10}$$

Note that previous defenses (e.g., Krum, AUROR) improve their effectiveness by modifying the filter function $F()$, while our AG framework changes the inputs of $F()$ by utilizing multi-round gradients. Thus, our AG framework is a compatible approach, which can be combined with most existing defenses by modifying the filter function $F()$.

Moreover, our AG framework can generalize to a more practical setting, where only a portion of clients take part in the training in each round. In particular, in round t, suppose $\mathcal{P}_t \subseteq \{1, \cdots, m\}$ with $\frac{|\mathcal{P}_t|}{m} = \alpha$ is the proportion of clients that take part in the training, and we call α the client training rate. The server computes average gradients for the clients in $\mathcal{P}_t$ as follows.

$$\mathbf{g}_{i,avg}^t = \frac{1}{\mathcal{T}_i}\sum_{k\in\mathcal{T}_i}\mathbf{g}_i^k, \tag{11}$$

where $i \in \mathcal{P}_t$ are the clients that take part in the training in round t, $\mathcal{T}_i \subseteq \{1, \cdots, t\}$ are the rounds that client i took part in. Then, the server uses $\{\mathbf{g}_{i,avg}^t\}_{i\in\mathcal{P}_t}$ to filter out Byzantine clients.

3.2 Theoretical analysis

As discussed in Section 2.2, we use $\mathbf{g}_i^t$ to denote the gradients computed by gradient descent (GD) (shown in Eq. (2)) for client i in round t. However, in real-world scenarios, the local gradients of the clients are computed by SGD instead of GD. We use gradients computed by SGD $\widehat{\mathbf{g}}_i^t$ to approximate $\mathbf{g}_i^t$ as

$$\widehat{\mathbf{g}}_i^t = \frac{1}{B}\sum_{\mathbf{x}\in\mathcal{B}_i^t}\partial\ell(\boldsymbol{\theta}^{t-1};\mathbf{x})/\partial\boldsymbol{\theta}^{t-1}, \tag{12}$$

where B is the size of minibatch for SGD, $\mathcal{B}_i^t \subseteq \mathcal{D}_i$ is a random minibatch of SGD for client i in round t.

In t-th round, there are certain discrepancies between local SGD gradients $\widehat{\mathbf{g}}_i^t$ and optimal gradients $\mathbf{g}^t$. The discrepancies originate from two aspects: one is the differences between local datasets and the whole dataset used to compute gradients; the other is the random sampling in SGD. Therefore, we model the discrepancies $\widehat{\mathbf{g}}_i^t - \mathbf{g}^t$ as follows.

$$\widehat{\mathbf{g}}_i^t - \mathbf{g}^t = (\widehat{\mathbf{g}}_i^t - \mathbf{g}_i^t) + (\mathbf{g}_i^t - \mathbf{g}^t), \tag{13}$$

$$\widehat{\mathbf{g}}_i^t - \mathbf{g}_i^t \approx \boldsymbol{\zeta}_i^t \sim \frac{1}{\sqrt{B}}\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_i^t) \tag{14}$$

$$\mathbf{g}_i^t - \mathbf{g}^t = \boldsymbol{\mu}_i^t, \tag{15}$$

where $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_i^t)$ are normal distributions with zero mean and covariance matrix $\boldsymbol{\Sigma}_i^t$. We use $\boldsymbol{\mu}_i^t$ to represent the gradient discrepancies caused by the difference of datasets in round t and $\boldsymbol{\zeta}_i^t$ to characterize the gradient discrepancies caused by random sampling in SGD. Particularly, the stochastic gradient is a sum of independent, uniformly sampled contributions. Invoking the central limit theorem, $\boldsymbol{\zeta}_i^t$ is assumed to be Gaussian with covariance $\boldsymbol{\Sigma}_i^t/B$ (Mandt, Hoffman, and Blei 2017).

The average of gradients (computed by SGD) of client i in the first t rounds are denoted as

$$\widehat{\mathbf{g}}_{i,avg}^t = \frac{1}{t}\sum_{k=1}^{t}\widehat{\mathbf{g}}_i^k. \tag{16}$$

In order to measure the dispersion of $\{\widehat{\mathbf{g}}_{i,avg}^t\}_{i=1}^m$, we define variance v^t as

$$v^t = \frac{1}{m}\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \widehat{\boldsymbol{\mu}}^t\|^2, \tag{17}$$

where $\widehat{\boldsymbol{\mu}}^t = \sum_{i=1}^m \widehat{\mathbf{g}}_{i,avg}^t/m$.

We make the following assumptions for theoretical analysis.

Assumption 1. [Gradient bound] *Client gradients $\mathbf{g}_i^t$ and optimal gradients $\mathbf{g}^t$ are bounded, i.e., $\|\mathbf{g}_i^t\| \le c_1$, $\|\mathbf{g}^t\| \le c_1$, $t = 0, 1, \cdots, T$.*

Assumption 2. [Variance bound] (Bernstein et al. 2019) *$\sqrt{B}\boldsymbol{\zeta}_i^t$ have coordinate bounded variance, i.e. $Var[(\sqrt{B}\boldsymbol{\zeta}_i^t)_j] \le \sigma_{ij}^2$, $j = 1, 2, \cdots, d$, where $(\sqrt{B}\boldsymbol{\zeta}_i^t)_j$ is the j-th component of $\sqrt{B}\boldsymbol{\zeta}_i^t$, $Var[(\sqrt{B}\boldsymbol{\zeta}_i^t)_j]$ represents the variance of $(\sqrt{B}\boldsymbol{\zeta}_i^t)_j$, and d is the dimension of model*

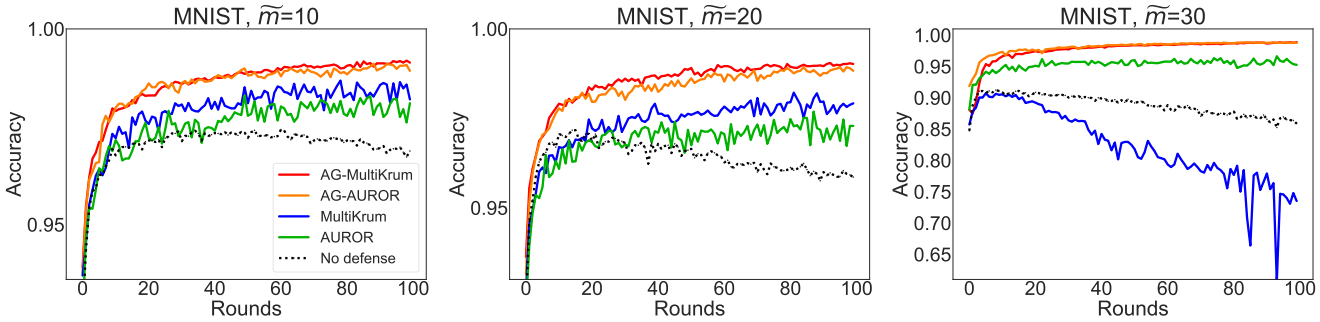

Figure 1: Accuracy of different defense methods under LIE attack on MNIST across $\widetilde{m}$={10, 20, 30} Byzantine clients.

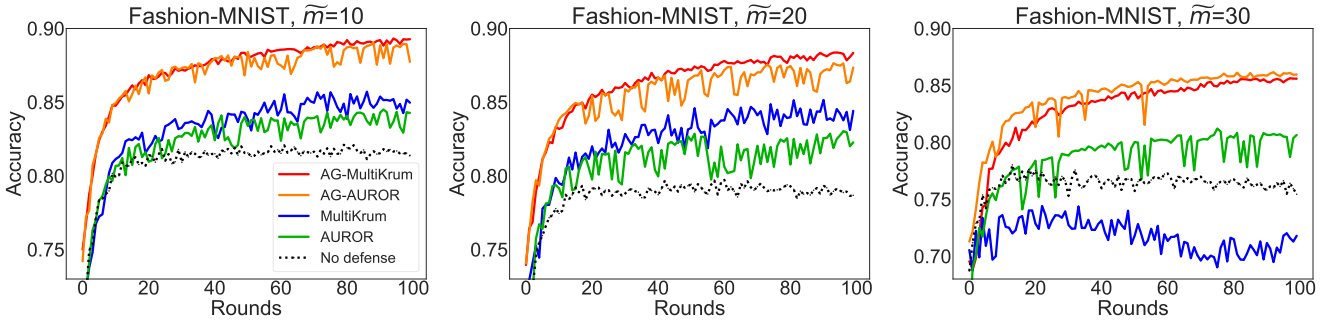

Figure 2: Accuracy of different defense methods under LIE attack on Fashion-MNIST across $\widetilde{m}$={10, 20, 30} Byzantine clients.

parameters $\boldsymbol{\theta}$.

Assumption 3. [Client independence and sampling independence] (Mandt, Hoffman, and Blei 2017) *Gradient computation (by SGD) on different clients is independent, and random sampling in SGD in different rounds is independent, i.e.,* $\{\zeta_i^t\}_{i,t}$ *are independent random variables.*

Assumption 1 can be easily satisfied by clipping the gradients during training. Assumption 2 directly follows (Bernstein et al. 2019). In Assumption 3, the independence of different clients is a basic setting in FL while the independence of random sampling follows from (Mandt, Hoffman, and Blei 2017). With the above assumptions, we provide the following variance reduction guarantee of $\widehat{\mathbf{g}}_{i,avg}^t$.

Proposition 1. *If Assumption 1, 2, 3 hold, then the expectation of the variance of the averaged gradients* $\{\hat{\mathbf{g}}_{i,avg}^t\}_{i=1}^m$ *is upper bounded as follows.*

$$\mathbb{E}[v^t] \leq (1-\frac{1}{m})\frac{c_2}{B}t^{-1} + 8(1-\frac{1}{m})c_1^2, \qquad (18)$$

where $c_2 = \sum_{i=1}^m \sum_{j=1}^d \sigma_{ij}^2/m$, d is the dimension of model parameters $\boldsymbol{\theta}$.

In Proposition 1, c_1, c_2, and p are constants, B is the mini-batch size for SGD, and m is the number of clients. The proof is provided in Appendix A. Since $p \in [0, 1)$, the upper bound of the expectation of the variance $\mathbb{E}[v^t]$ gets smaller as the number of rounds t increases. Therefore, we can conclude that the variance of one-round gradients has a higher upper bound, while the variance of multi-round gradients has a lower upper bound. Consequently, our AG framework can reduce the variance of benign gradients by utilizing multi-round gradients, and further improve the strength of the defenses.

4 Experiments

4.1 Experimental setup

Attack methods In our experiments, we apply two state-of-the-art attack methods: "fall of empires" (FE) (Xie, Koyejo, and Gupta 2020) and "a little is enough" (LIE) (Baruch, Baruch, and Goldberg 2019).

Suppose there are m clients: $\widetilde{m}$ of them are Byzantine and their gradients are $\widetilde{\mathbf{g}}_1^t, \ldots, \widetilde{\mathbf{g}}_{\widetilde{m}}^t$ in round t, and $m - \widetilde{m}$ of the clients are benign and their gradients are $\mathbf{g}_1^t, \ldots, \mathbf{g}_{(m-\widetilde{m})}^t$.

FE attacks the global model by sending the opposite values of the benign gradients to the server, i.e., $\widetilde{\mathbf{g}}_1^t = \ldots = \widetilde{\mathbf{g}}_{\widetilde{m}}^t = \frac{\epsilon}{m-\widetilde{m}} \sum_{i=1}^{m-\widetilde{m}} -\mathbf{g}_i^t$. We set the coefficient $\epsilon = 1$ according to the original paper.

LIE adds the standard deviations to the benign gradients, i.e., $\widetilde{\mathbf{g}}_1^t = \ldots = \widetilde{\mathbf{g}}^{t\widetilde{m}} = \frac{1}{m-\widetilde{m}} \sum_{i=1}^{m-\widetilde{m}} \mathbf{g}^{ti} + \epsilon\boldsymbol{\sigma}$, where $\boldsymbol{\sigma}$ is the element-wise standard deviation of $\mathbf{g}_1^t, \ldots, \mathbf{g}^{t(m-\widetilde{m})}$. We set the coefficient $\epsilon = 1.5$ according to the original paper.

Baselines We modify MultiKrum (Blanchard et al. 2017) and AUROR (Shen, Tople, and Saxena 2016) to their AG versions, i.e., AG-MultiKrum and AG-AUROR. MultiKrum and

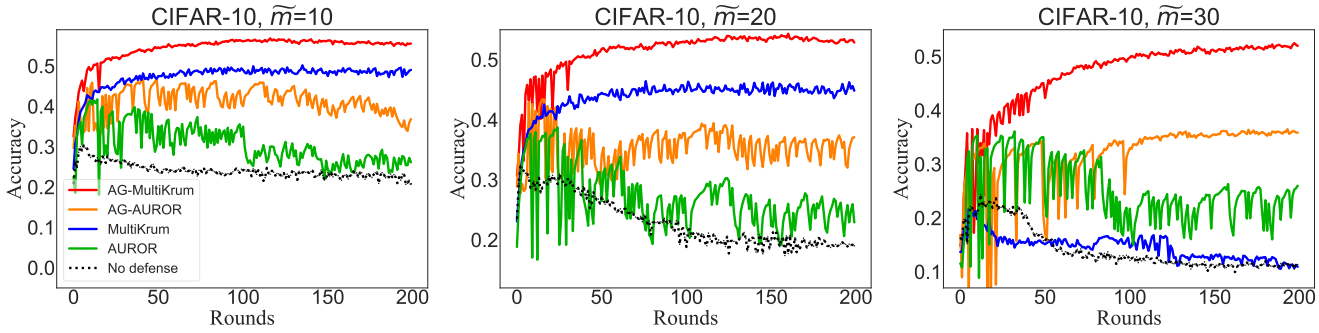

Figure 3: Accuracy of different defense methods under LIE attack on CIFAR-10 across $\widetilde{m}$={10, 20, 30} Byzantine clients.

Table 1: Accuracy (mean ± standard deviation) of different defenses under FE attack on MNIST, Fashion-MNIST, and CIFAR-10 datasets. Best results are shown in bold.

$\widetilde{m}$	Dataset	MultiKrum	AG-MultiKrum	AUROR	AG-AUROR	No defense
10	MNIST	97.58 ± 0.05	99.03 ± 0.07	98.06 ± 0.04	**99.18** ± 0.06	97.05 ± 0.01
	FMNIST	84.26 ± 0.04	89.01 ± 0.04	84.95 ± 0.08	**89.45** ± 0.10	82.91 ± 0.02
	CIFAR-10	51.61 ± 0.63	53.16 ± 0.50	51.78 ± 1.80	**54.55** ± 1.02	49.59 ± 1.55
20	MNIST	94.87 ± 0.12	97.59 ± 0.07	96.52 ± 0.08	**98.98** ± 0.03	94.90 ± 0.09
	FMNIST	82.89 ± 0.19	87.53 ± 0.19	83.63 ± 0.20	**89.11** ± 0.38	79.27 ± 0.16
	CIFAR-10	44.10 ± 1.33	46.59 ± 0.57	50.75 ± 1.31	**54.42** ± 0.72	44.34 ± 0.75
30	MNIST	93.90 ± 0.30	95.94 ± 0.24	95.80 ± 0.56	**98.44** ± 0.04	11.37 ± 1.55
	FMNIST	72.70 ± 0.96	82.29 ± 1.55	81.88 ± 1.06	**86.39** ± 0.22	12.43 ± 1.07
	CIFAR-10	35.16 ± 1.05	39.40 ± 1.61	46.56 ± 0.54	**49.28** ± 1.10	10.00 ± 0.00

AUROR use one-round gradients to detect Byzantine attacks, while AG-MultiKrum and AG-AUROR utilize multi-round gradients for detection. Overall, we compare the performance of four defense methods: MultiKrum, AG-MultiKrum, AUROR, and AG-AUROR.

Datasets Our experiments are conducted on 3 real-world datasets in CV domain: MNIST (LeCun et al. 1998), Fashion-MNIST (Xiao, Rasul, and Vollgraf 2017), and CIFAR-10 (Krizhevsky, Hinton et al. 2009). MNIST dataset contains binary images of handwritten digits. There are 60,000 training images and 10,000 testing images in MNIST dataset. Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images in CIFAR-10 dataset.

Settings The number of total client m is set to 100. We consider different capabilities of the attacker, where the number of Byzantine clients $\widetilde{m}$={10, 20, 30}. We train the model using SGD with momentum=0.5, and learning rate $\eta = 0.1$. We set the number of training rounds $T = 100$ for MNIST and Fashion-MNIST datasets, and $T = 200$ for CIFAR-10 dataset. In each round, the client trains its local data for 5 epochs and the batch size is 64. Following the setting of (McMahan et al. 2017), we utilize a 4-layer CNN to train the model on MNIST and Fashion-MNIST datasets and a 5-layer CNN on CIFAR-10 dataset. All experiments are run on the same machine with Intel E5-2665 CPU, 32GB RAM, and four GeForce GTX 1080Ti GPU. We set the client training rate α=0.1, i.e., randomly select 10% of the clients for training in each round. All experiments are run five times, and we report the average results.

4.2 Results under "a little is enough" (LIE) attack

Figure 1, Figure 2, and Figure 3 illustrate the results of different defense methods under LIE attack across $\widetilde{m}$={10, 20, 30} Byzantine clients on MNIST, Fashion-MNIST, and CIFAR-10 datasets, respectively. From these three figures, we can observe that:

(1) Our AG framework generally outperforms all the original version of defenses on all datasets across $\widetilde{m}$={10, 20, 30} Byzantine clients, which verifies the efficacy of our AG framework.

(2) When $\widetilde{m}$={10, 20}, AG-MultiKrum only gives a slightly better performance than MultiKrum. Our interpretation is that when there are few Byzantine clients, MultiKrum is capable of dealing with the Byzantine attacks. Nevertheless, when $\widetilde{m}$=30, MultiKrum fails to detect the Byzantine attacks. We hypothesise that the reason is that when the number of benign clients becomes small, the variance of benign gradients becomes high, and MultiKrum fails to defend against the Byzantine attacks when $\widetilde{m}$ becomes large. By contrast, our AG framework can successfully reduces the variance of benign gradients by utilizing the multi-round gradients for detection. Thus, AG-MultiKrum can effectively detect

Table 2: Accuracy of MultiKrum and AG-MultiKrum under LIE attack on CIFAR-10 dataset across different α. Best results are shown in bold.

α	0.05	0.1	0.2	0.3	0.4	0.5
MultiKrum	21.18	22.61	24.90	24.81	25.53	26.61
AG-MultiKrum	**44.33**	**52.50**	**52.76**	**52.84**	**52.99**	**53.36**

Table 3: Variance of one-round (benign) gradients and multi-round (benign) gradients on CIFAR-10 dataset across α={0.05, 0.1, 0.2, 0.3, 0.4, 0.5}. Lower variances are in bold.

α	0.05	0.1	0.2	0.3	0.4	0.5
One-round gradients	12.33	11.14	10.91	10.72	10.59	10.44
Multi-round gradients	**5.67**	**4.62**	**4.49**	**4.43**	**4.37**	**4.32**

Byzantine attacks even when $\widetilde{m}$ is large.

(3) When $\widetilde{m}$=10, all defense methods have similarly high accuracies on MNIST and Fashion-MNIST datasets. Even the accuracy of "No defense" is very high. We hypothesise that the reason is that when the number of Byzantine clients is small, these Byzantine clients can hardly affect the global model. As $\widetilde{m}$ becomes larger, the accuracy of MultiKurm and AUROR start to drop while AG-MultiKrum and AG-AUROR can still maintain high accuracy.

(4) As shown in Figure 3, AUROR exhibits bad performance on CIFAR-10 dataset and oscillates during the training progress, we hypothesise that this is because the variance of benign gradients is high enough for the attack to compromise AUROR. By contrast, our AG-AUROR can largely improve the performance of AUROR by utilizing multi-round gradients.

4.3 Results under "fall of empires" (FE) attack

Table 1 demonstrates the results of different defense methods under FE attack across $\widetilde{m}$={10, 20, 30} Byzantine clients on CIFAR-10, MNIST, and Fashion-MNIST datasets, respectively. As evidenced by Table 1, our AG framework achieves a better performance than the baseline methods.

From all the results, we can also observe that when $\widetilde{m}$={10, 20}, AG-MultiKrum and AG-AUROR only achieve a slightly better performance than their original versions, and when $\widetilde{m}$={30}, combining with our AG framework can significantly outperform the baselines. Our interpretation is that when benign clients are the overwhelming majority, the variance of the benign gradients is low, and therefore the Byzantine clients are unable to effectively attack the model. As the number of benign clients decreases, the variance of benign gradients becomes large, and Byzantine clients can achieve more efficient attack performance. Our AG framework can decrease the variance of benign gradients, make the Byzantine clients more difficult to attack the model, and thus can improve the performance of MultiKrum and AUROR.

4.4 Results on different client training rate

In this section, we discuss the impact of client training rate α. A higher α means more benign clients. Table 2 demonstrates the accuracy of MultiKrum and AG-MultiKrum under "a little is enough" (LIE) attack on CIFAR-10 dataset across α={0.05, 0.1, 0.2, 0.3, 0.4, 0.5}. As shown in Table 2, AG-MultiKrum outperforms MultiKrum, which validates that our AG framework can effectively defend against Byzantine attacks under various client training rates. In Table 3, we list the variance of one-round (benign) gradients and multi-round (benign) gradients across different α. The variance of multi-round gradients is consistently lower than one-round gradients across all α, which verifies that our AG framework can indeed decrease the variance of benign gradients. Moreover, in Table 2, the performance of all defense methods increases as α becomes larger. We attribute this to the decrease of the variance of benign gradients as the number of benign clients increases.

5 Conclusion and future work

In this paper, we show that existing defenses cannot well defend against the Byzantine attacks, due to the high variance of one-round gradients. To address this problem, we propose a novel Average of Gradients (AG) framework, which uses multi-round gradients to detect Byzantine attacks. We theoretically show that our AG framework can reduce the variance of benign gradients, and lead to better defense performance against Byzantine attacks. Empirical studies on three real-world datasets justify the efficacy of our AG framework. In this paper, we only discuss Byzantine attacks that degrade the overall performance, i.e., untargeted attacks. Defenses against targeted attacks need further investigation.

References

Abbe, E. A.; Khandani, A. E.; and Lo, A. W. 2012. Privacy-preserving methods for sharing financial risk exposures. *American Economic Review*, 102(3): 65–70.

Baruch, G.; Baruch, M.; and Goldberg, Y. 2019. A little is enough: Circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems*, 8632–8642.

Bernstein, J.; Zhao, J.; Azizzadenesheli, K.; and Anandkumar, A. 2019. signSGD with Majority Vote is Communication Efficient and Fault Tolerant. In *7th International Conference on Learning Representations*.

Blanchard, P.; El Mhamdi, E. M.; Guerraoui, R.; and Stainer, J. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 118–128.

Buch, V. H.; Ahmed, I.; and Maruthappu, M. 2018. Artificial intelligence in medicine: current trends and future possibilities. *Br J Gen Pract.*, 68(668): 143–144.

Chen, C.; Wang, H.; Liu, W.; Zhao, X.; Hu, T.; and Chen, G. 2019. Two-stage label embedding via neural factorization machine for multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3304–3311.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, Y.; Meng, G.; Chen, K.; Hu, X.; and He, J. 2020. Towards Security Threats of Deep Learning Systems: A Survey. *IEEE Transactions on Software Engineering*.

Kantarci, B.; and Mouftah, H. T. 2014. Trustworthy sensing for public safety in cloud-centric internet of things. *IEEE Internet of Things Journal*, 1(4): 360–368.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 2278–2324.

Lyu, L.; Li, Y.; Nandakumar, K.; Yu, J.; and Ma, X. 2020. How to democratise and protect AI: Fair and differentially private decentralised deep learning. *IEEE Transactions on Dependable and Secure Computing*.

Lyu, L.; Yu, H.; Ma, X.; Chen, C.; Sun, L.; Zhao, J.; Yang, Q.; and Yu, P. S. 2022. Privacy and Robustness in Federated Learning: Attacks and Defenses. *arXiv preprint arXiv:2012.06337*.

Lyu, L.; Yu, H.; and Yang, Q. 2020. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*.

Mandt, S.; Hoffman, M. D.; and Blei, D. M. 2017. Stochastic gradient descent as approximate bayesian inference. *arXiv preprint arXiv:1704.04289*.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 1273–1282.

Shen, S.; Tople, S.; and Saxena, P. 2016. Auror: Defending against poisoning attacks in collaborative deep learning systems. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, 508–519.

Shokri, R.; and Shmatikov, V. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 1310–1321.

Tian, Y.; Wan, Y.; Lyu, L.; Yao, D.; Jin, H.; and Sun, L. 2022. FedBERT: When Federated Learning Meets Pre-Training. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Voigt, P.; and Von dem Bussche, A. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10: 3152676.

Wang, C.; Zhou, T.; Chen, C.; Hu, T.; and Chen, G. 2019. CAMO: A collaborative ranking method for content based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5224–5231.

Wu, C.; Wu, F.; Liu, R.; Lyu, L.; Huang, Y.; and Xie, X. 2021. FedKD: Communication Efficient Federated Learning via Knowledge Distillation. *arXiv preprint arXiv:2108.13323*.

Wu, C.; Wu, F.; Lyu, L.; Di, T.; Huang, Y.; and Xie, X. 2020. Fedctr: Federated native ad ctr prediction with multi-platform user behavior data. *arXiv preprint arXiv:2007.12135*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xie, C.; Koyejo, O.; and Gupta, I. 2020. Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation. In *Uncertainty in Artificial Intelligence*, 261–270.

Yang, Q.; Liu, Y.; Cheng, Y.; Kang, Y.; Chen, T.; and Yu, H. 2019. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 1–207.

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, 5650–5659.

A Proof of Proposition 1

We make the following assumptions for theoretical analysis.

Assumption 1. [Gradient bound] *Client gradients $\mathbf{g}_i^t$ and optimal gradients $\mathbf{g}^t$ are bounded, i.e., $\|\mathbf{g}_i^t\| \leq c_1$, $\|\mathbf{g}^t\| \leq c_1$, $t = 0, 1, \cdots, T$.*

Assumption 2. [Variance bound] *$\sqrt{B}\boldsymbol{\zeta}_i^t$ have coordinate bounded variance, i.e. $Var[(\sqrt{B}\boldsymbol{\zeta}_i^t)_j] \leq \sigma_{ij}^2$, $j = 1, 2, \cdots, d$, where $(\sqrt{B}\boldsymbol{\zeta}_i^t)_j$ is the j-th component of $\sqrt{B}\boldsymbol{\zeta}_i^t$, $Var[(\sqrt{B}\boldsymbol{\zeta}_i^t)_j]$ represents the variance of $(\sqrt{B}\boldsymbol{\zeta}_i^t)_j$, and d is the dimension of model parameters $\boldsymbol{\theta}$.*

Assumption 3. [Client independence and sampling independence] *Gradient computation (by SGD) on different clients is independent, and random sampling in SGD in different rounds is independent, i.e., $\{\boldsymbol{\zeta}_i^t\}_{i,t}$ are independent random variables.*

With the above assumptions, we provide the following variance reduction guarantee of $\widehat{\mathbf{g}}_{i,avg}^t$.

Proposition 1. *If Assumption 1, 2, 3 hold, then the expectation of the variance of the averaged gradients $\{\hat{\mathbf{g}}_{i,avg}^t\}_{i=1}^m$ is upper bounded as follows.*

$$\mathbb{E}[v^t] \leq (1 - \frac{1}{m})\frac{c_2}{B}t^{-1} + 8(1 - \frac{1}{m})c_1^2, \tag{19}$$

where $c_2 = \sum_{i=1}^m \sum_{j=1}^d \sigma_{ij}^2/m$, d is the dimension of model parameters $\boldsymbol{\theta}$.

Proof. Let $\mathbf{g}_{avg}^t = \sum_{k=1}^t \mathbf{g}^k/t$, then

$$\begin{aligned}
\|\mathbf{g}_{avg}^t\| =& \|\frac{1}{t}\sum_{k=1}^t \mathbf{g}^k\| \\
\leq& \frac{1}{t}\sum_{k=1}^t \|\mathbf{g}^k\| \\
\leq& \frac{1}{t} \cdot tc_1 \\
=& c_1
\end{aligned} \tag{20}$$

Consider the distribution of $\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t$ as follows.

$$\begin{aligned}
\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t =& \frac{1}{t}\sum_{k=1}^t \hat{\mathbf{g}}_{i,avg}^k - \frac{1}{t}\sum_{k=1}^t \mathbf{g}^k \\
=& \frac{1}{t}\sum_{k=1}^t (\hat{\mathbf{g}}_{i,avg}^k - \mathbf{g}^k) \\
\approx& \frac{1}{t}\sum_{k=1}^t (\boldsymbol{\mu}_i^k + \boldsymbol{\zeta}_i^k) \\
\sim& \mathcal{N}(\frac{1}{t}\sum_{k=1}^t \boldsymbol{\mu}_i^k, \frac{1}{Bt^2}\sum_{k=1}^t \boldsymbol{\Sigma}_i^k).
\end{aligned}$$

Then, the variance of $(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j$ can be bounded as follows.

$$\begin{aligned}
Var[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j] =& \frac{1}{Bt^2}\sum_{k=1}^t (\boldsymbol{\Sigma}_i^k)_{jj} \\
\leq& \frac{1}{Bt^2} \cdot t\sigma_{ij}^2 \\
=& \frac{\sigma_{ij}^2}{B} \cdot t^{-1},
\end{aligned} \tag{21}$$

where $()_j$ denotes the j-th component of the vector, $()_{jj}$ denotes the j-th diagonal element of the matrix. Here, the inequality is due to Assumption 2, more specifically, $(\boldsymbol{\Sigma}_i^k)_{jj} = Var[(\sqrt{B}\boldsymbol{\zeta}_i^k)_j] \leq \sigma_{ij}^2$.

In order to bound $\mathbb{E}[v^t]$, we rewrite v^t as follows.

$$
\begin{aligned}
v^t =& \frac{1}{m}\sum_{i=1}^{m}\|(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t) - (\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t)\|^2 \\
=& \frac{1}{m}\sum_{i=1}^{m}(\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 + \|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2 - 2(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t)) \\
=& \frac{1}{m}\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 + \frac{1}{m}\sum_{i=1}^{m}\|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2 - 2\cdot\frac{1}{m}\sum_{i=1}^{m}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t) \\
=& \frac{1}{m}\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 + \|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2 - 2\|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2 \\
=& \frac{1}{m}\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 - \|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2.
\end{aligned} \tag{22}
$$

Here, we can rewrite $\|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2$ as follows.

$$
\begin{aligned}
\|\widehat{\boldsymbol{\mu}}^t - \mathbf{g}_{avg}^t\|^2 =& \|\frac{1}{m}\sum_{i=1}^{m}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)\|^2 \\
=& \frac{1}{m^2}(\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 + 2\sum_{i=1}^{m}\sum_{l=i+1}^{m}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\mathbf{g}}_{l,avg}^t - \mathbf{g}_{avg}^t)) \\
=& \frac{1}{m^2}\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 + \frac{2}{m^2}\sum_{i=1}^{m}\sum_{l=i+1}^{m}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\mathbf{g}}_{l,avg}^t - \mathbf{g}_{avg}^t).
\end{aligned} \tag{23}
$$

By combining Eq. (22) and Eq. (23), we can get

$$
v^t = (\frac{1}{m} - \frac{1}{m^2})\sum_{i=1}^{m}\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2 - \frac{2}{m^2}\sum_{i=1}^{m}\sum_{l=i+1}^{m}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\mathbf{g}}_{l,avg}^t - \mathbf{g}_{avg}^t).
$$

Then, we can compute $\mathbb{E}[v^t]$ as follows.

$$
\begin{aligned}
\mathbb{E}[v^t] =& (\frac{1}{m} - \frac{1}{m^2})\sum_{i=1}^{m}\mathbb{E}[\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2] - \frac{2}{m^2}\sum_{i=1}^{m}\sum_{l=i+1}^{m}\mathbb{E}[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)^\top(\widehat{\mathbf{g}}_{l,avg}^t - \mathbf{g}_{avg}^t)] \\
=& (\frac{1}{m} - \frac{1}{m^2})\sum_{i=1}^{m}\mathbb{E}[\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2] - \frac{2}{m^2}\sum_{i=1}^{m}\sum_{l=i+1}^{m}(\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t]^\top - \mathbf{g}_{avg}^t)(\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t] - \mathbf{g}_{avg}^t).
\end{aligned} \tag{24}
$$

Here, the second equality is due to the independence of $\widehat{\mathbf{g}}_{i,avg}^t$ and $\widehat{\mathbf{g}}_{j,avg}^t$ for $i \neq j$ (client independence, Assumption 3).

We bound $\mathbb{E}[\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2]$ as follows.

$$\begin{aligned}
\mathbb{E}[\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2] =& \mathbb{E}[\sum_{j=1}^{d}(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j^2] \\
=& \sum_{j=1}^{d}\mathbb{E}[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j^2] \\
=& \sum_{j=1}^{d}(Var[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j] + (\mathbb{E}[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j])^2) \\
=& \sum_{j=1}^{d}Var[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j] + \sum_{j=1}^{d}(\mathbb{E}[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j])^2 \\
=& \sum_{j=1}^{d}Var[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j] + \|\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t]\|^2 \\
\leq& \sum_{j=1}^{d}Var[(\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t)_j] + (\|\mathbb{E}\widehat{\mathbf{g}}_{i,avg}^t\| + \|\mathbf{g}_{avg}^t\|)^2 \\
\leq& \sum_{j=1}^{d}\frac{\sigma_{ij}^2}{B}\cdot t^{-1} + 4c_1^2. \qquad (25)
\end{aligned}$$

Here, the first inequality is due to triangle inequality, the second inequality is due to (20), (21)and Assumption 1.

We bound $(\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t] - \mathbf{g}_{avg}^t)^\top(\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t] - \mathbf{g}_{avg}^t)$ as follows.

$$\begin{aligned}
(\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t] - \mathbf{g}_{avg}^t)^\top(\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t] - \mathbf{g}_{avg}^t) \geq& -\|\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t] - \mathbf{g}_{avg}^t\| \cdot \|\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t] - \mathbf{g}_{avg}^t\| \\
\geq& -(\|\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t]\| + \|\mathbf{g}_{avg}^t\|) \cdot (\|\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t]\| + \|\mathbf{g}_{avg}^t\|) \\
\geq& -4c_1^2. \qquad (26)
\end{aligned}$$

Here, the first inequality is due to the lower bound of inner product, the second inequality due to triangle inequality, the third inequality is due to (20) and Assumption 1.

Finally, by combining the (24), (25), and (26), we can get:

$$\begin{aligned}
\mathbb{E}[v^t] =& (\frac{1}{m} - \frac{1}{m^2})\sum_{i=1}^{m}\mathbb{E}[\|\widehat{\mathbf{g}}_{i,avg}^t - \mathbf{g}_{avg}^t\|^2] - \frac{2}{m^2}\sum_{i=1}^{m}\sum_{l=i+1}^{m}(\mathbb{E}[\widehat{\mathbf{g}}_{i,avg}^t]^\top - \mathbf{g}_{avg}^t)(\mathbb{E}[\widehat{\mathbf{g}}_{l,avg}^t] - \mathbf{g}_{avg}^t) \\
\leq& (\frac{1}{m} - \frac{1}{m^2})\sum_{i=1}^{m}(\sum_{j=1}^{d}\frac{\sigma_{ij}^2}{B}\cdot t^{-1} + 4c_1^2) - \frac{2}{m^2}\cdot\frac{m(m-1)}{2}(-4c_1^2) \\
=& (1 - \frac{1}{m})\frac{c_2}{B}t^{-1} + 8(1 - \frac{1}{m})c_1^2.
\end{aligned}$$

□