Provably Secure Federated Learning against Malicious Clients

Xiaoyu Cao, Jinyuan Jia, Neil Zhenqiang Gong
Duke University
{xiaoyu.cao, jinyuan.jia, neil.gong}@duke.edu

Abstract

Federated learning enables clients to collaboratively learn a shared global model without sharing their local training data with a cloud server. However, malicious clients can corrupt the global model to predict incorrect labels for testing examples. Existing defenses against malicious clients leverage Byzantine-robust federated learning methods. However, these methods cannot provably guarantee that the predicted label for a testing example is not affected by malicious clients. We bridge this gap via *ensemble federated learning*. In particular, given any base federated learning algorithm, we use the algorithm to learn multiple global models, each of which is learnt using a randomly selected subset of clients. When predicting the label of a testing example, we take majority vote among the global models. We show that our ensemble federated learning with any base federated learning algorithm is provably secure against malicious clients. Specifically, the label predicted by our ensemble global model for a testing example is provably not affected by a bounded number of malicious clients. Moreover, we show that our derived bound is tight. We evaluate our method on MNIST and Human Activity Recognition datasets. For instance, our method can achieve a certified accuracy of 88% on MNIST when 20 out of 1,000 clients are malicious.

Introduction

Federated learning (**??**) is an emerging machine learning paradigm, which enables many clients (e.g., smartphones, IoT devices, and organizations) to collaboratively learn a model without sharing their local training data with a cloud server. Due to its promise for protecting privacy of the clients' local training data and the emerging privacy regulations such as General Data Protection Regulation (GDPR), federated learning has been deployed by industry. For instance, Google has deployed federated learning for next-word prediction on Android Gboard. Existing federated learning methods mainly follow a *single-global-model* paradigm. Specifically, a cloud server maintains a *global model* and each client maintains a *local model*. The global model is trained via multiple iterations of communications between the clients and server. In each iteration, three steps are performed: 1) the server sends the current global model to the clients; 2) the clients update their local models based on the global model and their local training data, and send the model updates to the server; and 3) the server aggregates the model updates and uses them to update the global model. The learnt global model is then used to predict labels of testing examples.

However, such single-global-model paradigm is vulnerable to security attacks. In particular, an attacker can inject fake clients to federated learning or compromise existing clients, where we call the fake/compromised clients *malicious clients*. Such malicious clients can corrupt the global model via carefully tampering their local training data or model updates sent to the server. As a result, the corrupted global model has a low accuracy for the normal testing examples (**??**) or certain attacker-chosen testing examples (**???**). For instance, when learning an image classifier, the malicious clients can re-label the cars with certain strips as birds in their local training data and scale up their model updates sent to the server, such that the learnt global model incorrectly predicts a car with the strips as bird (**?**).

Various Byzantine-robust federated learning methods have been proposed to defend against malicious clients (**???????**). The main idea of these methods is to mitigate the impact of statistical outliers among the clients' model updates. They can bound the difference between the global model parameters learnt without malicious clients and the global model parameters learnt when some clients become malicious. However, these methods cannot provably guarantee that the label predicted by the global model for a testing example is not affected by malicious clients. Indeed, studies showed that malicious clients can still substantially degrade the testing accuracy of a global model learnt by a Byzantine-robust method via carefully tampering their model updates sent to the server (**???**).

In this work, we propose *ensemble federated learning*, the first federated learning method that is provably secure against malicious clients. Specifically, given n clients, we define a *subsample* as a set of k clients sampled from the n clients uniformly at random without replacement. For each subsample, we can learn a global model using a base federated learning algorithm with the k clients in the subsample. Since there are $\binom{n}{k}$ subsamples with k clients, $\binom{n}{k}$ global models can be trained in total. Suppose we are given a test-

ing example $\mathbf{x}$. We define p_i as the fraction of the $\binom{n}{k}$ global models that predict label i for $\mathbf{x}$, where $i = 1, 2, \cdots, L$. We call p_i *label probability*. Our *ensemble global model* predicts the label with the largest label probability for $\mathbf{x}$. In other words, our ensemble global model takes a majority vote among the global models to predict label for $\mathbf{x}$. Since each global model is learnt using a subsample with k clients, a majority of the global models are learnt using normal clients when most clients are normal. Therefore, the majority vote among the global models is secure against a bounded number of malicious clients.

Theory: Our first major theoretical result is that our ensemble global model provably predicts the same label for a testing example $\mathbf{x}$ when the number of malicious clients is no larger than a threshold, which we call *certified security level*. Our second major theoretical result is that we prove our derived certified security level is tight, i.e., when no assumptions are made on the base federated learning algorithm, it is impossible to derive a certified security level that is larger than ours. Note that the certified security level may be different for different testing examples.

Algorithm: Computing our certified security level for $\mathbf{x}$ requires its largest and second largest label probabilities. When $\binom{n}{k}$ is small (e.g., the n clients are dozens of organizations (**?**) and k is small), we can compute the largest and second largest label probabilities exactly via training $\binom{n}{k}$ global models. However, it is challenging to compute them exactly when $\binom{n}{k}$ is large. To address the computational challenge, we develop a Monte Carlo algorithm to estimate them with probabilistic guarantees via training N instead of $\binom{n}{k}$ global models.

Evaluation: We empirically evaluate our method on MNIST (**?**) and Human Activity Recognition datasets (**?**). We distribute the training examples in MNIST to clients to simulate federated learning scenarios, while the Human Activity Recognition dataset represents a real-world federated learning scenario, where each user is a client. We use the popular FedAvg developed by Google (**?**) as the base federated learning algorithm. Moreover, we use *certified accuracy* as our evaluation metric, which is a lower bound of the testing accuracy that a method can provably achieve no matter how the malicious clients tamper their local training data and model updates. For instance, our ensemble FedAvg with $N = 500$ and $k = 10$ can achieve a certified accuracy of 88% on MNIST when evenly distributing the training examples among 1,000 clients and 20 of them are malicious.

In summary, our key contributions are as follows:

- **Theory:** We propose ensemble federated learning, the first provably secure federated learning method against malicious clients. We derive a certified security level for our ensemble federated learning. Moreover, we prove that our derived certified security level is tight.
- **Algorithm:** We propose a Monte Carlo algorithm to compute our certified security level in practice.
- **Evaluation:** We evaluate our methods on MNIST and Human Activity Recognition datasets.

All our proofs are shown in Supplemental Material.

Algorithm 1 Single-global-model federated learning

Input: $\mathbf{C}$, $globalIter$, $localIter$, η, Agg.
Output: Global model $\mathbf{w}$.
$\mathbf{w} \leftarrow$ random initialization.
for $Iter\_global = 1, 2, \cdots, globalIter$ **do**
  /* Step I */
  The server sends $\mathbf{w}$ to the clients.
  /* Step II */
  for $i \in \mathbf{C}$ **do**
    $\mathbf{w}_i \leftarrow \mathbf{w}$.
    for $Iter\_local = 1, 2, \cdots, localIter$ **do**
      Sample a $Batch$ from local training data $\mathcal{D}_i$.
      $\mathbf{w}_i \leftarrow \mathbf{w}_i - \eta \nabla Loss(Batch; \mathbf{w}_i)$.
    end for
    Send $\mathbf{g}_i = \mathbf{w}_i - \mathbf{w}$ to the server.
  end for
  /* Step III */
  $\mathbf{g} \leftarrow \mathrm{Agg}(\mathbf{g}_1, \mathbf{g}_2, \cdots, \mathbf{g}_{|\mathbf{C}|})$.
  $\mathbf{w} \leftarrow \mathbf{w} - \eta \cdot \mathbf{g}$.
end for
return $\mathbf{w}$.

Background on Federated Learning

Assuming we have n clients $\mathbf{C} = \{1, 2, \cdots, n\}$ and a cloud server in a federated learning setting. The ith client holds some local training dataset $\mathcal{D}_i$, where $i = 1, 2, \cdots, n$. Existing federated learning methods (**????**) mainly focus on learning a single global model for the n clients. Specifically, the server maintains a global model and each client maintains a local model. Then, federated learning iteratively performs the following three steps, which are shown in Algorithm 1. In Step I, the server sends the current global model to the clients.[1] In Step II, each client trains a local model via fine-tuning the global model to its local training dataset. In particular, each client performs *localIter* iterations of stochastic gradient descent with a learning rate η to train its local model. Then, each client sends its model update (i.e., the difference between the local model and the global model) to the server. In Step III, the server aggregates the clients' model updates according to some aggregation rule *Agg* and uses the aggregated model update to update the global model. The three steps are repeated for *globalIter* iterations. Existing federated learning algorithms essentially use different aggregation rules in Step III. For instance, Google developed FedAvg (**?**), which computes the average of the clients' model updates weighted by the sizes of their local training datasets as the aggregated model update to update the global model.

We call such a federated learning algorithm that learns a single global model *base federated learning algorithm* and denote it as $\mathcal{A}$. Note that given any subset of the n clients $\mathbf{C}$, a base federated learning algorithm can learn a global model for them. Specifically, the server learns a global model via iteratively performing the three steps between the server and

[1]The server may select a subset of clients, but we assume the server sends the global model to all clients for convenience.

the given subset of clients.

Our Ensemble Federated Learning

Unlike single-global-model federated learning, our ensemble federated learning trains multiple global models, each of which is trained using the base algorithm $\mathcal{A}$ and a subsample with k clients sampled from the n clients uniformly at random without replacement. Among the n clients $\mathbf{C}$, we have $\binom{n}{k}$ subsamples with k clients. Therefore, $\binom{n}{k}$ global models can be trained in total if we train a global model using each subsample. For a given testing input $\mathbf{x}$, these global models may predict different labels for it. We define p_i as the fraction of the $\binom{n}{k}$ global models that predict label i for $\mathbf{x}$, where $i = 1, 2, \cdots, L$. We call p_i *label probability*. Note that p_i is an integer multiplication of $\frac{1}{\binom{n}{k}}$, which we will leverage to derive a tight security guarantee of ensemble federated learning. Moreover, p_i can also be viewed as the probability that a global model trained on a random subsample with k clients predicts label i for $\mathbf{x}$. Our *ensemble global model* predicts the label with the largest label probability for $\mathbf{x}$, i.e., we define:

$$h(\mathbf{C}, \mathbf{x}) = \operatorname*{argmax}_i p_i, \tag{1}$$

where h is our ensemble global model and $h(\mathbf{C}, \mathbf{x})$ is the label that our ensemble global model predicts for $\mathbf{x}$ when the ensemble global model is trained on clients $\mathbf{C}$.

Defining provable security guarantees against malicious clients: Suppose some of the n clients $\mathbf{C}$ become malicious. These malicious clients can arbitrarily tamper their local training data and model updates sent to the server in each iteration of federated learning. We denote by $\mathbf{C}'$ the set of n clients with malicious ones. Moreover, we denote by $M(\mathbf{C}')$ the number of malicious clients in $\mathbf{C}'$, e.g., $M(\mathbf{C}') = m$ means that m clients are malicious. Note that we don't know which clients are malicious. For a testing example $\mathbf{x}$, our goal is to show that our ensemble global model h provably predicts the same label for $\mathbf{x}$ when the number of malicious clients is bounded. Formally, we aim to show the following:

$$h(\mathbf{C}', \mathbf{x}) = h(\mathbf{C}, \mathbf{x}), \forall \mathbf{C}', M(\mathbf{C}') \leq m^*, \tag{2}$$

where $h(\mathbf{C}', \mathbf{x})$ is the label that the ensemble global model trained on the clients $\mathbf{C}'$ predicts for $\mathbf{x}$. We call m^* *certified security level*. When a global model satisfies Equation (2) for a testing example $\mathbf{x}$, we say the global model achieves a provable security guarantee for $\mathbf{x}$ with a certified security level m^*. Note that the certified security level may be different for different testing examples. Next, we derive the certified security level of our ensemble global model.

Deriving certified security level using exact label probabilities: Suppose we are given a testing example $\mathbf{x}$. Assuming that, when there are no malicious clients, our ensemble global model predicts label y for $\mathbf{x}$, p_y is the largest label probability, and p_z is the second largest label probability. Moreover, we denote by p'_y and p'_z respectively the label probabilities for y and z in the ensemble global model when there are malicious clients. Suppose m clients become malicious. Then, $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$ fraction of subsamples with k clients include at least one malicious client. In the worst-case scenario, for each global model learnt using a subsample including at least one malicious client, its predicted label for $\mathbf{x}$ changes from y to z. Therefore, in the worst-case scenario, the m malicious clients decrease the largest label probability p_y by $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$ and increase the second largest label probability p_z by $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, i.e., we have $p'_y = p_y - (1 - \frac{\binom{n-m}{k}}{\binom{n}{k}})$ and $p'_z = p_z + (1 - \frac{\binom{n-m}{k}}{\binom{n}{k}})$. Our ensemble global model still predicts label y for $\mathbf{x}$, i.e., $h(\mathbf{C}', \mathbf{x}) = h(\mathbf{C}, \mathbf{x}) = y$, once m satisfies the following inequality:

$$p'_y > p'_z \iff p_y - p_z > 2 - 2\frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{3}$$

In other words, the largest integer m that satisfies the inequality (3) is our certified security level m^* for the testing example $\mathbf{x}$. The inequality (3) shows that our certified security level is related to the gap $p_y - p_z$ between the largest and second largest label probabilities in the ensemble global model trained on the clients $\mathbf{C}$ without malicious ones. For instance, when a testing example has a larger gap $p_y - p_z$, the inequality (3) may be satisfied by a larger m, which means that our ensemble global model may have a larger certified security level for the testing example.

Deriving certified security level using approximate label probabilities: When $\binom{n}{k}$ is small (e.g., several hundred), we can compute the exact label probabilities p_y and p_z via training $\binom{n}{k}$ global models, and compute the certified security level via inequality (3). However, when $\binom{n}{k}$ is large, it is computationally challenging to compute the exact label probabilities via training $\binom{n}{k}$ global models. For instance, when $n = 100$ and $k = 10$, there are already 1.73×10^{13} global models, training all of which is computationally intractable in practice. Therefore, we also derive certified security level using a lower bound $\underline{p_y}$ of p_y (i.e., $\underline{p_y} \leq p_y$) and an upper bound $\overline{p}_z$ of p_z (i.e., $\overline{p}_z \geq p_z$). We use a lower bound $\underline{p_y}$ of p_y and an upper bound $\overline{p}_z$ of p_z because our certified security level is related to the gap $p_y - p_z$ and we aim to estimate a lower bound of the gap. The lower bound $\underline{p_y}$ and upper bound $\overline{p}_z$ may be estimated by different methods. For instance, in the next section, we propose a Monte Carlo algorithm to estimate a lower bound $\underline{p_y}$ and an upper bound $\overline{p}_z$ via only training N of the $\binom{n}{k}$ global models.

Next, we derive our certified security level based on the probability bounds $\underline{p_y}$ and $\overline{p}_z$. One way is to replace p_y and p_z in inequality (3) as $\underline{p_y}$ and $\overline{p}_z$, respectively. Formally, we have the following inequality:

$$\underline{p_y} - \overline{p}_z > 2 - 2\frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{4}$$

If an m satisfies inequality (4), then the m also satisfies inequality (3), because $\underline{p_y} - \overline{p}_z \leq p_y - p_z$. Therefore, we can

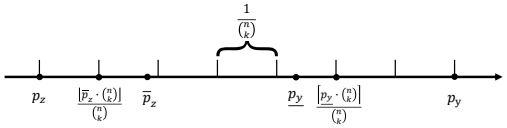


Figure 1: An example to illustrate the relationships between $p_y, \underline{p_y}$, and $\frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}}$ as well as $p_z, \overline{p}_z$, and $\frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$.

find the largest integer m that satisfies the inequality (4) as the certified security level m^*. However, we found that the certified security level m^* derived based on inequality (4) is not tight, i.e., our ensemble global model may still predict label y for $\mathbf{x}$ even if the number of malicious clients is larger than m^* derived based on inequality (4). The key reason is that the label probabilities are integer multiplications of $\frac{1}{\binom{n}{k}}$. Therefore, we normalize $\underline{p_y}$ and $\overline{p}_z$ as integer multiplications of $\frac{1}{\binom{n}{k}}$ to derive a tight certified security level. Specifically, we derive the certified security level as the largest integer m that satisfies the following inequality (formally described in Theorem 1):

$$\frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} > 2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}}. \quad (5)$$

Figure 1 illustrates the relationships between $p_y, \underline{p_y}$, and $\frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}}$ as well as $p_z, \overline{p}_z$, and $\frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$. When an m satisfies inequality (4), the m also satisfies inequality (5), because $\underline{p_y} - \overline{p}_z \leq \frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}} - \frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$. Therefore, the certified security level derived based on inequality (4) is smaller than or equals the certified security level derived based on inequality (5). Note that when $\underline{p_y} = p_y$ and $\overline{p}_z = p_z$, both (4) and (5) reduce to (3) as the label probabilities are integer multiplications of $\frac{1}{\binom{n}{k}}$. The following theorem formally summarizes our certified security level.

Theorem 1. *Given n clients $\mathbf{C}$, an arbitrary base federated learning algorithm $\mathcal{A}$, a subsample size k, and a testing example $\mathbf{x}$, we define an ensemble global model h as Equation (1). y and z are the labels that have the largest and second largest label probabilities for $\mathbf{x}$ in the ensemble global model. $\underline{p_y}$ is a lower bound of p_y and $\overline{p}_z$ is an upper bound of p_z. Formally, $\underline{p_y}$ and $\overline{p}_z$ satisfy the following conditions:*

$$\max_{i \neq y} p_i = p_z \leq \overline{p}_z \leq \underline{p_y} \leq p_y. \quad (6)$$

Then, h provably predicts y for $\mathbf{x}$ when at most m^* clients in $\mathbf{C}$ become malicious, i.e., we have:

$$h(\mathbf{C}', \mathbf{x}) = h(\mathbf{C}, \mathbf{x}) = y, \forall \mathbf{C}', M(\mathbf{C}') \leq m^*, \quad (7)$$

where m^* is the largest integer m ($0 \leq m \leq n - k$) that satisfies inequality (5).

Our Theorem 1 is applicable to any base federated learning algorithm, any lower bound $\underline{p_y}$ of p_y and any upper bound $\overline{p}_z$ of p_z that satisfy (6). When the lower bound $\underline{p_y}$ and upper bound $\overline{p}_z$ are estimated more accurately, i.e., $\underline{p_y}$ and $\overline{p}_z$ are respectively closer to p_y and p_z, our certified security level may be larger. The following theorem shows that our derived certified security level is tight, i.e., when no assumptions on the base federated learning algorithm are made, it is impossible to derive a certified security level that is larger than ours for the given probability bounds $\underline{p_y}$ and $\overline{p}_z$.

Theorem 2. *Suppose $\underline{p_y} + \overline{p}_z \leq 1$. For any $\mathbf{C}'$ satisfying $M(\mathbf{C}') > m^*$, i.e., at least $m^* + 1$ clients are malicious, there exists a base federated learning algorithm $\mathcal{A}^*$ that satisfies (6) but $h(\mathbf{C}', \mathbf{x}) \neq y$ or there exist ties.*

Computing the Certified Security Level

Suppose we are given n clients $\mathbf{C}$, a base federated learning algorithm $\mathcal{A}$, a subsample size k, and a testing dataset $\mathcal{D}$ with d testing examples. For each testing example $\mathbf{x}_t$ in $\mathcal{D}$, we aim to compute its label $\hat{y}_t$ predicted by our ensemble global model h and the corresponding certified security level $\hat{m}_t^*$. To compute the certified security level based on our Theorem 1, we need a lower bound $\underline{p_{\hat{y}_t}}$ of the largest label probability $p_{\hat{y}_t}$ and an upper bound $\overline{p}_{\hat{z}_t}$ of the second largest label probability $p_{\hat{z}_t}$. When $\binom{n}{k}$ is small, we can compute the exact label probabilities via training $\binom{n}{k}$ global models. When $\binom{n}{k}$ is large, we propose a Monte Carlo algorithm to estimate the predicted label and the two probability bounds for all testing examples in $\mathcal{D}$ simultaneously with a confidence level $1 - \alpha$ via training N of the $\binom{n}{k}$ global models.

Computing predicted label and probability bounds for one testing example: We first discuss how to compute the predicted label $\hat{y}_t$ and probability bounds $\underline{p_{\hat{y}_t}}$ and $\overline{p}_{\hat{z}_t}$ for one testing example $\mathbf{x}_t$. We sample N subsamples with k clients from the n clients uniformly at random without replacement and use them to train N global models $g_1, g_2, \cdots, g_N$. We use the N global models to predict labels for $\mathbf{x}_t$ and count the frequency of each label. We treat the label with the largest frequency as the predicted label $\hat{y}_t$. Recall that, based on the definition of label probability, a global model trained on a random subsample with k clients predicts label $\hat{y}_t$ for $\mathbf{x}_t$ with the label probability $p_{\hat{y}_t}$. Therefore, the frequency $N_{\hat{y}_t}$ of the label $\hat{y}_t$ among the N global models follows a binomial distribution $B(N, p_{\hat{y}_t})$ with parameters N and $p_{\hat{y}_t}$. Thus, given $N_{\hat{y}_t}$ and N, we can use the standard one-sided Clopper-Pearson method (**?**) to estimate a lower bound $\underline{p_{\hat{y}_t}}$ of $p_{\hat{y}_t}$ with a confidence level $1 - \alpha$. Specifically, we have $\underline{p_{\hat{y}_t}} = \mathcal{B}(\alpha; N_{\hat{y}_t}, N - N_{\hat{y}_t} + 1)$, where $\mathcal{B}(q; v, w)$ is the qth quantile from a beta distribution with shape parameters v and w. Moreover, we can estimate $\overline{p}_{\hat{z}_t} = 1 - \underline{p_{\hat{y}_t}} \geq 1 - p_{\hat{y}_t} \geq p_{z_t}$ as an upper bound of $p_{\hat{z}_t}$.

Computing predicted labels and probability bounds for d testing examples: One method to compute the predicted labels and probability bounds for the d testing examples is to apply the above process to each testing example individually. However, such method is computationally intractable

Algorithm 2 Computing Predicted Label and Certified Security Level

Input: $\mathbf{C}$, $\mathcal{A}$, k, N, $\mathcal{D}$, α.
Output: Predicted label and certified security level for each testing example in $\mathcal{D}$.
$g_1, g_2, \cdots, g_N \leftarrow$ SAMPLE&TRAIN$(\mathbf{C}, \mathcal{A}, k, N)$
for $\mathbf{x}_t$ **in** $\mathcal{D}$ **do**
  counts$[i] \leftarrow \sum_{l=1}^{N} \mathbb{I}(g_l(\mathbf{x}_t) = i), i \in \{1, 2, \cdots, L\}$
  /* $\mathbb{I}$ is the indicator function */
  $\hat{y}_t \leftarrow$ index of the largest entry in counts (ties are broken uniformly at random)
  $\underline{p_{\hat{y}_t}} \leftarrow \mathcal{B}\left(\frac{\alpha}{d}; N_{\hat{y}_t}, N - N_{\hat{y}_t} + 1\right)$
  $\overline{p}_{\hat{z}_t} \leftarrow 1 - \underline{p_{\hat{y}_t}}$
  if $\underline{p_{\hat{y}_t}} > \overline{p}_{\hat{z}_t}$ **then**
    $\hat{m}_t^* \leftarrow$ SEARCHLEVEL$(\underline{p_{\hat{y}_t}}, \overline{p}_{\hat{z}_t}, k, |\mathbf{C}|)$
  else
    $\hat{y}_t \leftarrow$ ABSTAIN, $\hat{m}_t^* \leftarrow$ ABSTAIN
  end if
end for
return $\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_d$ and $\hat{m}_1^*, \hat{m}_2^*, \cdots, \hat{m}_d^*$

because it requires training N global models for every testing example. To address the computational challenge, we propose a method that only needs to train N global models in total. Our idea is to split α among the d testing examples. Specifically, we follow the above process to train N global models and use them to predict labels for the d testing examples. For each testing example $\mathbf{x}_t$, we estimate the lower bound $\underline{p_{\hat{y}_t}} = \mathcal{B}\left(\frac{\alpha}{d}; N_{\hat{y}_t}, N - N_{\hat{y}_t} + 1\right)$ with confidence level $1 - \alpha/d$ instead of $1 - \alpha$. According to the *Bonferroni correction*, the simultaneous confidence level of estimating the lower bounds for the d testing examples is $1 - \alpha$. Following the above process, we still estimate $\overline{p}_{\hat{z}_t} = 1 - \underline{p_{\hat{y}_t}}$ as an upper bound of $p_{\hat{z}_t}$ for each testing example.

Complete algorithm: Algorithm 2 shows our algorithm to compute the predicted labels and certified security levels for the d testing examples in $\mathcal{D}$. The function SAMPLE&TRAIN randomly samples N subsamples with k clients and trains N global models using the base federated learning algorithm $\mathcal{A}$. Given the probability bounds $\underline{p_{\hat{y}_t}}$ and $\overline{p}_{\hat{z}_t}$ for a testing example $\mathbf{x}_t$, the function SEARCHLEVEL finds the certified security level $\hat{m}_t^*$ via finding the largest integer m that satisfies (5). For example, SEARCHLEVEL can simply start m from 0 and iteratively increase it by one until finding $\hat{m}_t^*$.

Probabilistic guarantees: In Algorithm 2, since we estimate the lower bound $\underline{p_{\hat{y}_t}}$ using the Clopper-Pearson method, there is a probability that the estimated lower bound is incorrect, i.e., $\underline{p_{\hat{y}_t}} > p_{\hat{y}_t}$. When the lower bound is estimated incorrectly for a testing example $\mathbf{x}_t$, the certified security level $\hat{m}_t^*$ outputted by Algorithm 2 for $\mathbf{x}_t$ may also be incorrect, i.e., there may exist an $\mathbf{C}'$ such that $M(\mathbf{C}') \leq \hat{m}_t^*$ but $h(\mathbf{C}', \mathbf{x}_t) \neq \hat{y}_t$. In other words, our Algorithm 2 has probabilistic guarantees for its outputted certified security levels. However, in the following theorem, we prove the probability that Algorithm 2 returns an incorrect certified security level for at least one testing example is at most α.

Dataset	MNIST	HAR
Model architecture	CNN	DNN
Number of clients	1,000	30
$globalIter$	3,000	5,000
$localIter$	5	
Learning rate η	0.001	
Batch size	32	

Table 1: Federated learning settings and hyperparameters.

Theorem 3. *The probability that Algorithm 2 returns an incorrect certified security level for at least one testing example in $\mathcal{D}$ is bounded by α, which is equivalent to:*

$$Pr(\cap_{\mathbf{x}_t \in \mathcal{D}}(h(\mathbf{C}', \mathbf{x}_t) = \hat{y}_t, \forall \mathbf{C}', M(\mathbf{C}') \leq \hat{m}_t^* | \hat{y}_t \neq \textit{ABSTAIN})) \geq 1 - \alpha. \quad (8)$$

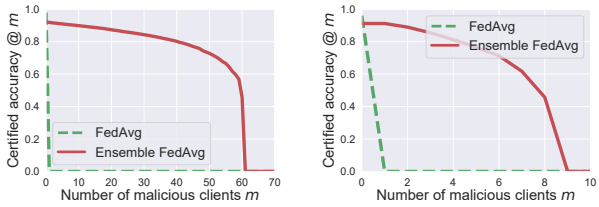
Note that when the probability bounds are estimated deterministically, e.g., when $\binom{n}{k}$ is small and the exact label probabilities can be computed via training $\binom{n}{k}$ global models, the certified security level obtained from our Theorem 1 is also deterministic.

Experiments

Experimental Setup

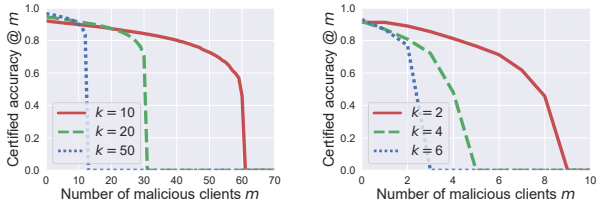
Datasets, model architectures, and base algorithm: We use MNIST (**?**) and Human Activity Recognition (HAR) datasets (**?**). MNIST is used to simulate federated learning scenarios, while HAR represents a real-world federated learning scenario. Specifically, MNIST has 60,000 training examples and 10,000 testing examples. We consider $n = 1,000$ clients and we split them into 10 groups. We assign a training example with label l to the lth group of clients with probability q and assign it to each remaining group with a probability $\frac{1-q}{9}$. After assigning a training example to a group, we distribute it to a client in the group uniformly at random. The parameter q controls local training data distribution on clients and we call q *degree of non-IID*. $q = 0.1$ means that clients' local training data are IID, while a larger q indicates a larger degree of non-IID. By default, we set $q = 0.5$. However, we will study the impact of q (degree of non-IID) on our method. HAR includes human activity data from 30 users, each of which is a client. The task is to predict a user's activity based on the sensor signals (e.g., acceleration) collected from the user's smartphone. There are 6 possible activities (e.g., walking, sitting, and standing), indicating a 6-class classification problem. There are 10,299 examples in total and each example has 561 features. We use 75% of each user's examples as training examples and the rest as testing examples.

We consider a convolutional neural network (CNN) architecture (shown in Supplemental Material) for MNIST. For HAR, we consider a deep neural network (DNN) with two fully-connected hidden layers, each of which contains 256 neurons and uses ReLU as the activation function. We



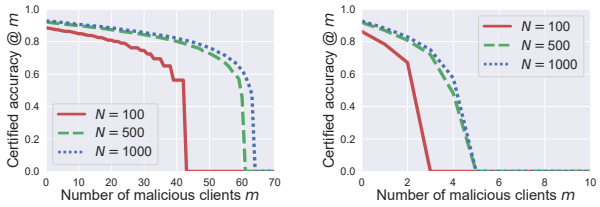
(a) MNIST (b) HAR

Figure 2: FedAvg vs. ensemble FedAvg.



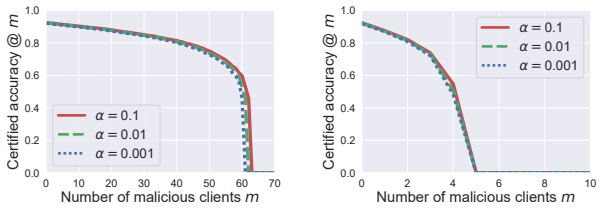
(a) MNIST (b) HAR

Figure 3: Impact of k on our ensemble FedAvg.



(a) MNIST (b) HAR

Figure 4: Impact of N on our ensemble FedAvg.



(a) MNIST (b) HAR

Figure 5: Impact of α on our ensemble FedAvg.

use the popular FedAvg (**?**) as the base federated learning algorithm. Recall that a base federated learning algorithm has hyperparameters (shown in Algorithm 1): $globalIter$, $localIter$, learning rate η, and batch size. Table 1 summarizes these hyperparameters for FedAvg in our experiments. In particular, we set the $globalIter$ in Table 1 because FedAvg converges with such settings.

Evaluation metric: We use *certified accuracy* as our evaluation metric. Specifically, we define the *certified accuracy at m malicious clients* (denoted as CA@m) for a federated learning method as the fraction of testing examples in the testing dataset $\mathcal{D}$ whose labels are correctly predicted by the method and whose certified security levels are at least m. Formally, we define CA@m as follows:

$$\text{CA@}m = \frac{\sum_{\mathbf{x}_t \in \mathcal{D}} \mathbb{I}(\hat{y}_t = y_t) \cdot \mathbb{I}(\hat{m}_t^* \geq m)}{|\mathcal{D}|}, \quad (9)$$

where $\mathbb{I}$ is the indicator function, y_t is the true label for $\mathbf{x}_t$, and $\hat{y}_t$ and $\hat{m}_t^*$ are respectively the predicted label and certified security level for $\mathbf{x}_t$. Intuitively, CA@m means that when at most m clients are malicious, the accuracy of the federated learning method for $\mathcal{D}$ is at least CA@m no matter what attacks the malicious clients use (i.e., no matter how the malicious clients tamper their local training data and model updates). Note that CA@0 reduces to the standard accuracy when there are no malicious clients.

When we can compute the exact label probabilities via training $\binom{n}{k}$ global models, the CA@m of our ensemble global model h computed using the certified security levels derived from Theorem 1 is deterministic. When $\binom{n}{k}$ is large, we estimate predicted labels and certified security levels using Algorithm 2, and thus our CA@m has a confidence level $1 - \alpha$ according to Theorem 3.

Parameter settings: Our method has three parameters: N, k, and α. Unless otherwise mentioned, we adopt the following default settings for them: $N = 500$, $\alpha = 0.001$, $k = 10$ for MNIST, and $k = 2$ for HAR. Under such default setting for HAR, we have $\binom{n}{k} = \binom{30}{2} = 435 < N = 500$ and we can compute the exact label probabilities via training 435 global models. Therefore, we have deterministic certified accuracy for HAR under the default setting. We will explore the impact of each parameter while using the default settings for the other two parameters. For HAR, we set $k = 4$ when exploring the impact of N (i.e., Figure 4(b)) and α (i.e., Figure 5(b)) since the default setting $k = 2$ gives deterministic certified accuracy, making N and α not relevant.

Experimental Results

Single-global-model FedAvg vs. ensemble FedAvg: Figure 2 compares single-global-model FedAvg and ensemble FedAvg with respect to certified accuracy on the two datasets. When there are no malicious clients (i.e., $m = 0$), single-global-model FedAvg is more accurate than ensemble FedAvg. This is because ensemble FedAvg uses a subsample of clients to train each global model. However, single-global-model FedAvg has 0 certified accuracy when just one client is malicious. This is because a single malicious client can arbitrarily manipulate the global model learnt by FedAvg (**?**). However, the certified accuracy of ensemble FedAvg reduces to 0 when up to 61 and 9 clients (6.1% and 30%) are malicious on MNIST and HAR, respectively. Note that it is unknown whether existing Byzantine-robust federated learning methods have non-zero certified accuracy when $m > 0$, and thus we cannot compare ensemble FedAvg with them.

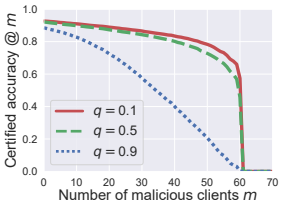


Figure 6: Impact of the degree of non-IID q on MNIST.

Impact of k, N, and α: Figure 3, 4, and 5 show the impact of k, N, and α, respectively. k achieves a trade-off between accuracy under no malicious clients and security under malicious clients. Specifically, when k is larger, the ensemble global model is more accurate at $m = 0$, but the certified accuracy drops more quickly to 0 as m increases. This is because when k is larger, it is more likely for the sampled k clients to include malicious ones. The certified accuracy increases as N or α increases. This is because training more global models or a larger α allows Algorithm 2 to estimate tighter probability bounds and larger certified security levels. When N increases from 100 to 500, the certified accuracy increases significantly. However, when N further grows to 1,000, the increase of certified accuracy is marginal. Our results show that we don't need to train too many global models in practice, as the certified accuracy saturates when N is larger than some threshold.

Impact of degree of non-IID q: Figure 6 shows the certified accuracy of our ensemble FedAvg on MNIST when the clients' local training data have different degrees of non-IID. We observe that the certified accuracy drops when q increases from 0.5 to 0.9, which represents a high degree of non-IID. However, the certified accuracy is still high when m is small for $q = 0.9$, e.g., the certified accuracy is still 83% when $m = 10$. This is because although each global model trained using a subsample of clients is less accurate when the local training data are highly non-IID, the ensemble of multiple global models is still accurate.

Related Work

In federated learning, the first category of studies (**?????**) aim to design federated learning methods that can learn more accurate global models and/or analyze their convergence properties. For instance, FedMA (**?**) constructs the global model via matching and averaging the hidden elements in a neural network with similar feature extraction signatures. The second category of studies (**??????????????????**) aim to improve the communication efficiency between the clients and server via sparsification, quantization, and/or encoding of the model updates sent from the clients to the server. The third category of studies (**????????**) aim to explore the privacy/fairness issues of federated learning and their defenses.These studies often assume a single global model is shared among the clients. Smith et al. (**?**) proposed to learn a customized model for each client via multi-task learning. Our work is on security of federated learning, which is orthogonal to the studies above. Multiple studies (**????**) showed that the global model's accuracy can be significantly downgraded by malicious clients. Existing defenses against malicious clients leverage Byzantine-robust aggregation rules such as Krum (**?**), trimmed mean (**?**), coordinate-wise median (**?**), and Bulyan (**?**). However, they cannot provably guarantee that the global model's predicted label for a testing example is not affected by malicious clients. As a result, they may be broken by strong attacks that carefully craft the model updates sent from the malicious clients to the server, e.g., (**?**). We propose ensemble federated learning whose predicted label for a testing example is provably not affected by a bounded number of malicious clients.

We note that ensemble methods were also proposed as provably secure defenses (e.g., (**?**)) against data poisoning attacks. However, they are insufficient to defend against malicious clients that can manipulate both the local training data and the model updates. In particular, a provably secure defense against data poisoning attacks guarantees that the label predicted for a testing example is unaffected by a bounded number of poisoned training examples. However, a single malicious client can poison an arbitrary number of its local training examples, breaking the assumption of provably secure defenses against data poisoning attacks.

Conclusion

In this work, we propose ensemble federated learning and derive its tight provable security guarantee against malicious clients. Moreover, we propose an algorithm to compute the certified security levels. Our empirical results on two datasets show that our ensemble federated learning can effectively defend against malicious clients with provable security guarantees. Interesting future work includes estimating the probability bounds deterministically and considering the internal structure of a base federated learning algorithm to further improve our provable security guarantees.

Acknowledgement

We thank the anonymous reviewers for insightful reviews. This work was supported by NSF grant No.1937786. Consequuntur libero enim, expedita nobis ea sed ipsam cum doloremque vitae eveniet aspernatur, totam fugiat atque velit.Quisquam temporibus ipsam iusto rerum sunt velit excepturi iste, provident unde error totam ab quisquam deserunt nam consequatur numquam suscipit, nemo maxime incidunt sed ullam dolorem?Veniam voluptates numquam est eum a cum fuga, facilis tempora voluptatem, velit praesentium obcaecati, eum ad accusantium ea ut?Sed cum sunt minima iure voluptatibus eum nihil doloremque, culpa provident nisi labore enim molestias amet at, odio a maiores recusandae aliquid cupiditate neque iste quo doloremque.Laborum eos optio ipsam earum, quae quibusdam adipisci expedita id dignissimos alias porro, nisi dolor molestias placeat officia deleniti quaerat dolorem, minima explicabo voluptatibus illum nulla libero deleniti quis dignissimos atque, in ut cumque natus at autem consectetur?Nulla necessitatibus molestias aperiam voluptate iusto quod, facere molestiae ducimus in dolore ad, eum