# Provably Secure Federated Learning

**Xiaoyu Cao**
Duke University
xiaoyu.cao@duke.edu

**Jinyuan Jia**
Duke University
jinyuan.jia@duke.edu

**Neil Zhenqiang Gong**
Duke University
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 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.

## 1 Introduction

Federated learning [19, 29] 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. 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 [43, 13] or certain attacker-chosen testing examples [4, 7, 42].

Various Byzantine-robust federated learning methods have been proposed to defend against malicious clients [8, 11, 31, 45, 44, 10, 1]. 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

NeurIPS-SpicyFL 2020, Vancouver, Canada.

Byzantine-robust method via carefully tampering their model updates sent to the server [7, 13, 43]. We discuss more related work in Appendix C.

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 testing 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.

**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 [18] 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 evaluate our method on MNIST [20] and Human Activity Recognition [3] datasets. We distribute the training examples in MNIST to clients to simulate federated learning scenarios with different degrees of non-IID clients' local training data, while the Human Activity Recognition dataset represents a real-world federated learning scenario. We use the popular FedAvg [29] 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.

All our proofs are shown in the Appendix.

## 2 Our Ensemble Federated Learning

Unlike single-global-model federated learning, our ensemble federated learning trains multiple global models, each of which is trained using a base federated learning algorithm $\mathcal{A}$ and a subsample with $k$ clients sampled from the $n$ clients uniformly at random without replacement. Among the $n$ clients $\mathbf{C} = \{C_1, C_2, \cdots, C_n\}$, 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 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 $h(\mathbf{C}, \mathbf{x}) = \operatorname{argmax}_i p_i$, 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. 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:

$$h(\mathbf{C}', \mathbf{x}) = h(\mathbf{C}, \mathbf{x}), \forall \mathbf{C}', M(\mathbf{C}') \leq m^*, \tag{1}$$

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 (1) 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:

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

The largest integer $m$ that satisfies the inequality (2) is our certified security level $m^*$ for the testing example $\mathbf{x}$. The inequality (2) 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 (2) 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 in one of our experiments), 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 (2). 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 $1.73 \times 10^{13}$ global models, training all of which is computationally intractable. 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 the bounds 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 (2) as $\underline{p_y}$ and $\overline{p}_z$, respectively. Formally, we have the following:

$$\underline{p_y} - \overline{p}_z > 2 - 2\frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{3}$$

If an $m$ satisfies inequality (3), then the $m$ also satisfies inequality (2), because $\underline{p_y} - \overline{p}_z \leq p_y - p_z$. Therefore, we can find the largest integer $m$ that satisfies the inequality (3) as the certified security level $m^*$. However, we found that the certified security level $m^*$ derived based on inequality (3) 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 (3). 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):

$$\frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} > 2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{4}$$

When an $m$ satisfies inequality (3), the $m$ also satisfies inequality (4), 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 (3) is smaller than or equals the certified security level derived based on inequality (4). Note that when $\underline{p_y} = p_y$ and $\overline{p}_z = p_z$, both (3) and (4) reduce to (2) as the label probabilities are integer multiplications of $\frac{1}{\binom{n}{k}}$. The following theorem formally summarizes our certified security level.

**Theorem 1.** *Suppose we are given $n$ clients $\mathbf{C}$, an arbitrary base federated learning algorithm $\mathcal{A}$, a subsample size $k$, and a testing example $\mathbf{x}$. $y$ and $z$ are the labels that have the largest and second largest label probabilities for $\mathbf{x}$ in our ensemble global model $h$. $\underline{p_y}$ 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:*

$$\max_{i \neq y} p_i = p_z \leq \overline{p}_z \leq \underline{p_y} \leq p_y. \tag{5}$$

*Then, $h$ provably predicts $y$ for $\mathbf{x}$ when at most $m^*$ clients in $\mathbf{C}$ become malicious, i.e., we have $h(\mathbf{C}', \mathbf{x}) = h(\mathbf{C}, \mathbf{x}) = y, \forall \mathbf{C}', M(\mathbf{C}') \leq m^*$, where $m^*$ is the largest integer $m$ ($0 \leq m \leq n - k$) that satisfies inequality (4).*

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 (5). 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 $M(\mathbf{C}') > m^*$, there exists a base federated learning algorithm $\mathcal{A}^*$ that satisfies (5) but $h(\mathbf{C}', \mathbf{x}) \neq y$ or there exist ties.*

## 3 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 $\hat{m}_t^*$ 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, 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 [12] to estimate a lower bound $\underline{p_{\hat{y}_t}}$ of $p_{\hat{y}_t}$ with a confidence level $1-\alpha$. Specifically, we have $\underline{p_{\hat{y}_t}} = \mathcal{B}\left(\alpha; N_{\hat{y}_t}, N - N_{\hat{y}_t} + 1\right)$, where $\mathcal{B}(q; v, w)$ is the $q$th 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 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 $\alpha$ 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 1 in Appendix 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 $\mathbf{x}_t$, SEARCHLEVEL finds the certified security level $\hat{m}_t^*$ via finding the largest integer $m$ that satisfies (4). For example, SEARCHLEVEL can simply start $m$ from 0 and iteratively increase it by one until finding $\hat{m}_t^*$.

**Probabilistic guarantees:** 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 our algorithm 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 has probabilistic guarantees for its outputted certified security levels. However, in the following theorem, we prove the probability that our algorithm returns an incorrect certified security level for at least one testing example is at most $\alpha$.

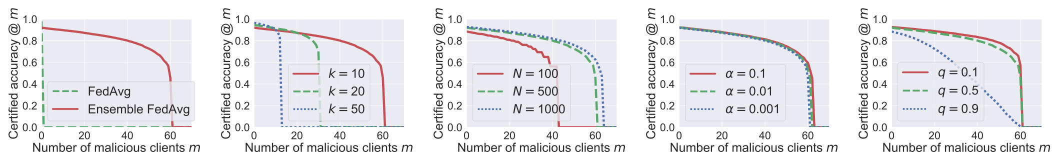
**Theorem 3.** *The probability that our algorithm (formally described in Algorithm 1 in Appendix) returns an incorrect certified security level for at least one testing example in $\mathcal{D}$ is bounded by $\alpha$, which is equivalent to: $Pr(\cap_{\mathbf{x}_t \in \mathcal{D}}(h(\mathbf{C}', \mathbf{x}_t) = \hat{y}_t, \forall \mathbf{C}', M(\mathbf{C}') \leq \hat{m}_t^* | \hat{y}_t \neq ABSTAIN)) \geq 1 - \alpha$.*

## 4 Experiments

### 4.1 Experimental Setup

**Datasets, model architectures, and base algorithm:** We use MNIST [20] and Human Activity Recognition (HAR) [3] datasets. We focus on MNIST in the main body, while the dataset description and experimental results for HAR are shown in Appendix B. We use MNIST to simulate federated learning scenarios. 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 $l$th 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. We consider a convolutional neural network (CNN) architecture (shown in Appendix) for MNIST. We use the popular FedAvg [29] as the base federated learning algorithm. We set the number of global iterations to 3,000 and in each global iteration, the clients train the local models for 5 local iterations using SGD with batch size 32 and learning rate 0.001.

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



(a) Comparison with FedAvg (b) Impact of $k$ (c) Impact of $N$ (d) Impact of $\alpha$ (e) Impact of $q$

**Figure 1: Experimental results of ensemble FedAvg on the MNIST dataset.**

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 1, 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 $\alpha$. Unless otherwise mentioned, we adopt the following default settings for them: $N = 500$, $\alpha = 0.001$, and $k = 10$ for MNIST.

### 4.2 Experimental Results

**Single-global-model FedAvg vs. ensemble FedAvg:** Figure 1(a) compares single-global-model FedAvg and ensemble FedAvg with respect to certified accuracy. 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 [8]. However, the certified accuracy of ensemble FedAvg reduces to 0 when up to 61 clients (6.1%) are malicious. 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.

**Impact of $k$, $N$, and $\alpha$:** Figure 1(b), 1(c), and 1(d) show the impact of $k$, $N$, and $\alpha$, 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 $\alpha$ increases. This is because training more global models or a larger $\alpha$ allows our algorithm 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 1(e) 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.

## 5 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 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.

## References

[1] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *NeurIPS*, pages 4613–4623, 2018.

[2] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *NeurIPS*, pages 1709–1720, 2017.

[3] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *ESANN*, 2013.

[4] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *AISTATS*, pages 2938–2948. PMLR, 2020.

[5] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In *NeurIPS*, pages 8635–8645, 2019.

[6] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *ICML*, pages 560–569, 2018.

[7] Arjun Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *ICML*, 2019.

[8] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NeurIPS*, 2017.

[9] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *CCS*, pages 1175–1191, 2017.

[10] Lingjiao Chen, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Draco: Byzantine-resilient distributed training via redundant gradients. In *ICML*, pages 903–912, 2018.

[11] Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *POMACS*, 2017.

[12] Charles J Clopper and Egon S Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934.

[13] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In *USENIX Security*, 2020.

[14] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[15] Jenny Hamer, Mehryar Mohri, and Ananda Theertha Suresh. Fedboost: Communication-efficient algorithms for federated learning. In *ICML*, 2020.

[16] Yufei Han and Xiangliang Zhang. Robust federated learning via collaborative machine teaching. In *AAAI*, pages 4075–4082, 2020.

[17] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *CCS*, pages 603–618, 2017.

[18] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[19] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. In *NeurIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[20] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *Available: http://yann. lecun. com/exdb/mnist*, 1998.

[21] Kangwook Lee, Maximilian Lam, Ramtin Pedarsani, Dimitris Papailiopoulos, and Kannan Ramchandran. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory*, 64(3):1514–1529, 2017.

[22] Liping Li, Wei Xu, Tianyi Chen, Georgios B Giannakis, and Qing Ling. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *AAAI*, volume 33, pages 1544–1551, 2019.

[23] Qinbin Li, Zeyi Wen, and Bingsheng He. Practical federated gradient boosting decision trees. In *AAAI*, pages 4642–4649, 2020.

[24] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *ICLR*, 2020.

[25] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *ICLR*, 2020.

[26] Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. In *ICML*, 2020.

[27] Fenglin Liu, Xian Wu, Shen Ge, Wei Fan, and Yuexian Zou. Federated learning for vision-and-language grounding problems. In *AAAI*, pages 11572–11579, 2020.

[28] Grigory Malinovsky, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local sgd to local fixed point methods for federated learning. In *ICML*, 2020.

[29] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[30] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE S&P*, pages 691–706. IEEE, 2019.

[31] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. In *ICML*, 2018.

[32] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *ICML*, pages 4615–4625, 2019.

[33] Xingchao Peng, Zijun Huang, Yizhe Zhu, and Kate Saenko. Federated adversarial domain adaptation. In *ICLR*, 2020.

[34] Shashank Rajput, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Detox: A redundancy-based framework for faster and more robust gradient aggregation. In *NeurIPS*, pages 10320–10330, 2019.

[35] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *ICML*, 2020.

[36] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[37] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *NeurIPS*, pages 4424–4434, 2017.

[38] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. Powersgd: Practical low-rank gradient compression for distributed optimization. In *NeurIPS*, pages 14236–14245, 2019.

[39] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *ICLR*, 2020.

[40] Yansheng Wang, Yongxin Tong, and Dingyuan Shi. Federated latent dirichlet allocation: A local differential privacy based framework. In *AAAI*, 2020.

[41] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *NeurIPS*, pages 1509–1519, 2017.

[42] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *ICLR*, 2020.

[43] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *UAI*, 2019.

[44] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Defending against saddle point attack in byzantine-robust distributed learning. In *ICML*, pages 7074–7084, 2019.

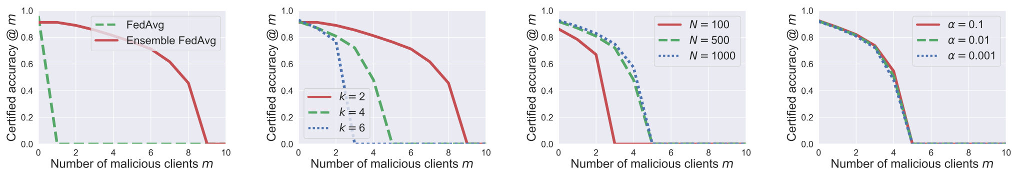
[45] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2018.

[46] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *ICML*, pages 7252–7261, 2019.

[47] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *NeurIPS*, pages 14747–14756, 2019.

**Algorithm 1** Computing Predicted Label and Certified Security Level

```
**Input:** $\mathbf{C}$, $\mathcal{A}$, $k$, $N$, $\mathcal{D}$, $\alpha$.
**Output:** Predicted label and certified security level for each testing example in $\mathcal{D}$.
$g_1, g_2, \cdots, g_N \leftarrow \textsc{Sample\&Train}(\mathbf{C}, \mathcal{A}, k, N)$
**for** $\mathbf{x}_t$ **in** $\mathcal{D}$ **do**
    $\text{counts}[i] \leftarrow \sum_{l=1}^{N} \mathbb{I}(g_l(\mathbf{x}_t) = i), i \in \{1, 2, \cdots, L\}$
    /* $\mathbb{I}$ 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**
        $\hat{m}_t^* \leftarrow \textsc{SearchLevel}(\underline{p_{\hat{y}_t}}, \overline{p}_{\hat{z}_t}, k, |\mathbf{C}|)$
    **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 $\hat{m}_1^*, \hat{m}_2^*, \cdots, \hat{m}_d^*$
```



(a) Comparison with FedAvg (b) Impact of $k$ (c) Impact of $N$ (d) Impact of $\alpha$

**Figure 2: Experimental results of ensemble FedAvg on the HAR dataset.**

| Layer | Size |
|---|---|
| Input | $28 \times 28 \times 1$ |
| Convolution + ReLU | $5 \times 5 \times 20$ |
| Max Pooling | $2 \times 2$ |
| Convolution + ReLU | $5 \times 5 \times 50$ |
| Max Pooling | $2 \times 2$ |
| Fully Connected + ReLU | 512 |
| Softmax | 10 |

**Table 1: The CNN architecture for MNIST.**

## A Conference Submission Information

This work has been accepted by AAAI-21.

## B HAR Dataset

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.

### B.1 Experimental Setup

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 set the number of global iterations to 5,000 and in each global iteration, the clients train the local models for 5 local iterations using SGD with batch size 32 and learning rate 0.001. By default, we set $N = 500$, $\alpha = 0.001$, and $k = 2$ for HAR.

### B.2 Experimental Results

Figure 2 shows the results on the HAR dataset. Note that under the 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. Moreover, we set $k = 4$ when exploring the impact of $N$ (i.e., Figure 2(c)) and $\alpha$ (i.e., Figure 2(d)) since the default setting $k = 2$ gives deterministic certified accuracy, making $N$ and $\alpha$ not relevant. Our observations are similar to those for the MNIST dataset.

## C More Related Work

In federated learning, the first category of studies [37, 25, 39, 27, 33] aim to design federated learning methods that can learn more accurate global models and/or analyze their convergence properties. For instance, FedMA [39] 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 [19, 29, 41, 2, 21, 36, 6, 38, 46, 32, 39, 23, 26, 15, 35, 28] 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 [9, 14, 17, 30, 47, 32, 40, 24] 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. [37] 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 [4, 43, 7, 22, 5, 42, 13, 16] 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 [8], trimmed mean [45], coordinate-wise median [45], and Bulyan [31]. 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., [13]. 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 Byzantine-robust aggregation rules DRACO [10] and DETOX [34] used majority vote in each iteration of aggregating clients' model updates when learning a global model. They require the server to distribute the same batch of training data to multiple clients, among which majority vote can be performed. They are applicable to distributed learning with centralized training data, where the server can control the distribution of data on clients. However, they are not applicable to federated learning, where the training data are decentralized on clients and the server cannot control them.

## D Proof of Theorem 1

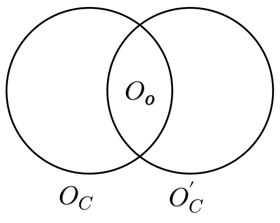


**Figure 3: Illustration of $\mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$.**

We first define a subsample of $k$ clients from $\mathbf{C}$ as $\mathcal{S}(\mathbf{C}, k)$. Then, we define the space of all possible subsamples from $\mathbf{C}$ as $\mathbf{O}_C = \{\mathcal{S}(\mathbf{C}, k)\}$ and the space of all possible subsamples from $\mathbf{C}'$ as $\mathbf{O}'_C = \{\mathcal{S}(\mathbf{C}', k)\}$. Let $\mathbf{O}_o = \{\mathcal{S}(\mathbf{C} \cap \mathbf{C}', k)\} = \mathbf{O}_C \cap \mathbf{O}'_C$ denote the space of all possible subsamples from the set of normal clients $\mathbf{C} \cap \mathbf{C}'$, and $\mathbf{O} = \{\mathcal{S}(\mathbf{C} \cup \mathbf{C}', k)\} = \mathbf{O}_C \cup \mathbf{O}'_C$ denote the space of all possible subsamples from either $\mathbf{C}$ or $\mathbf{C}'$. Figure 3 illustrates $\mathbf{O}_C$, $\mathbf{O}'_C$, and $\mathbf{O}_o$. We use a random variable $\mathbf{X}$ to denote a subsample $\mathcal{S}(\mathbf{C}, k)$ and $\mathbf{Y}$ to denote a subsample $\mathcal{S}(\mathbf{C}', k)$ in $\mathbf{O}$. We know that $\mathbf{X}$ and $\mathbf{Y}$ have the following probability distributions:

$$\Pr(\mathbf{X} = s) = \begin{cases} \frac{1}{\binom{n}{k}}, & \text{if } s \in \mathbf{O}_C \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

$$\Pr(\mathbf{Y} = s) = \begin{cases} \frac{1}{\binom{n}{k}}, & \text{if } s \in \mathbf{O}'_C \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Recall that given a set of clients $s$, the base federated learning algorithm $\mathcal{A}$ learns a global model. For simplicity, we denote by $\mathcal{A}(s, \mathbf{x})$ the predicted label of a testing example $\mathbf{x}$ given by this global model. We have the following equations:

$$p_y = \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y) \tag{8}$$

$$= \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in \mathbf{O}_o) \cdot \Pr(\mathbf{X} \in \mathbf{O}_o) + \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)), \tag{9}$$

$$p'_y = \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y) \tag{10}$$

$$= \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in \mathbf{O}_o) \cdot \Pr(\mathbf{Y} \in \mathbf{O}_o) + \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)). \tag{11}$$

Note that we have:

$$\Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in \mathbf{O}_o) = \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in \mathbf{O}_o), \tag{12}$$

$$\Pr(\mathbf{X} \in \mathbf{O}_o) = \Pr(\mathbf{Y} \in \mathbf{O}_o) = \frac{\binom{n-m}{k}}{\binom{n}{k}}, \tag{13}$$

where $m$ is the number of malicious clients. Therefore, we know:

$$\Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in \mathbf{O}_o) \cdot \Pr(\mathbf{X} \in \mathbf{O}_o) = \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in \mathbf{O}_o) \cdot \Pr(\mathbf{Y} \in \mathbf{O}_o). \tag{14}$$

By subtracting (9) from (11), we obtain:

$$p'_y - p_y = \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)). \tag{15}$$

Similarly, we have the following equation for any $i \neq y$:

$$p'_i - p_i = \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = i | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = i | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) \cdot \Pr(\mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)). \tag{16}$$

Therefore, we can show:

$$p'_y - p'_i = p_y - p_i + (p'_y - p_y) - (p'_i - p_i) \tag{17}$$

$$\begin{aligned} &= p_y - p_i \\ &+ [\Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = i | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o))] \cdot \Pr(\mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) \\ &- [\Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = i | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o))] \cdot \Pr(\mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)). \end{aligned} \tag{18}$$

Note that we have:

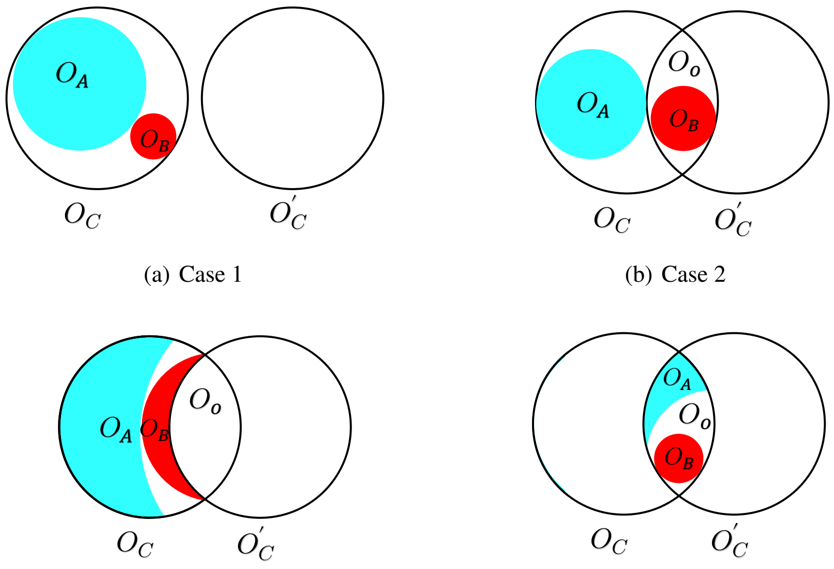
$$\Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = y | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{Y}, \mathbf{x}) = i | \mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) \geq -1, \tag{19}$$

$$\Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = y | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) - \Pr(\mathcal{A}(\mathbf{X}, \mathbf{x}) = i | \mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) \leq 1, \tag{20}$$

$$\Pr(\mathbf{Y} \in (\mathbf{O}'_C - \mathbf{O}_o)) = \Pr(\mathbf{X} \in (\mathbf{O}_C - \mathbf{O}_o)) = 1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{21}$$

Therefore, based on (18) and that $p_y$ and $p_i$ are integer multiplications of $\frac{1}{\binom{n}{k}}$, we have the following:

$$p'_y - p'_i \geq p_y - p_i + (-1) \cdot \left[1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}\right] - \left[1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}\right] \tag{22}$$



(a) Case 1 (b) Case 2

(c) Case 3 (d) Case 4

**Figure 4: Illustration of $\mathbf{O}_C, \mathbf{O}'_C, \mathbf{O}_o, \mathbf{O}_A$, and $\mathbf{O}_B$ in the four cases.**

$$= p_y - p_i - \left[2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}}\right] \tag{23}$$

$$= \frac{\lceil p_y \cdot \binom{n}{k} \rceil}{\binom{n}{k}} - \frac{\lfloor p_i \cdot \binom{n}{k} \rfloor}{\binom{n}{k}} - 2\left[1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}\right] \tag{24}$$

$$\geq \frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}} - \frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}} - 2\left[1 - \frac{\binom{n-m^*}{k}}{\binom{n}{k}}\right] \tag{25}$$

$$> 0, \tag{26}$$

which indicates $h(\mathbf{C}', \mathbf{x}) = y$.

## E Proof of Theorem 2

We prove Theorem 2 by constructing a base federated learning algorithm $\mathcal{A}^*$ such that the conditions in (5) are satisfied but $h(\mathbf{C}', \mathbf{x}) \neq y$ or there exist ties. We follow the definitions of $\mathbf{O}, \mathbf{O}_C, \mathbf{O}'_C, \mathbf{O}_o, \mathbf{X}$, and $\mathbf{Y}$ in the previous section. Next, we consider four cases (Figure 4 illustrates them).

**Case 1:** $m \geq n - k$.

In this case, we know $\mathbf{O}_o = \emptyset$. Let $\mathbf{O}_A \subseteq \mathbf{O}_C$ and $\mathbf{O}_B \subseteq \mathbf{O}_C$ such that $|\mathbf{O}_A| = \lceil \underline{p_y} \cdot \binom{n}{k} \rceil$, $|\mathbf{O}_B| = \lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor$, and $\mathbf{O}_A \cap \mathbf{O}_B = \emptyset$. Since $\underline{p_y} + \overline{p}_z \leq 1$, we have:

$$|\mathbf{O}_A| + |\mathbf{O}_B| = \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor \tag{27}$$

$$\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor (1 - \underline{p_y}) \cdot \binom{n}{k} \right\rfloor \tag{28}$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \tag{29}$$

$$= \binom{n}{k} = |\mathbf{O}_C|. \tag{30}$$

Therefore, we can always find such a pair of disjoint sets $(\mathbf{O}_A, \mathbf{O}_B)$. Figure 4(a) illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C$, and $\mathbf{O}'_C$. We can construct $\mathcal{A}^*$ as follows:

$$\mathcal{A}^*(s, \mathbf{x}) = \begin{cases} y, & \text{if } s \in \mathbf{O}_A \\ z, & \text{if } s \in \mathbf{O}_B \cup \mathbf{O}'_C \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{31}$$

We can show that such $\mathcal{A}^*$ satisfies the following probability properties:

$$p_y = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{32}$$

$$p_z = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z. \tag{33}$$

Therefore, $\mathcal{A}^*$ satisfies the probability conditions in (5). However, we have:

$$p'_z = \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = z) = 1, \tag{34}$$

which indicates $h(\mathbf{C}', \mathbf{x}) = z \neq y$.

**Case 2:** $m^* < m < n - k$, $0 \leq \underline{p_y} \leq 1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, and $0 \leq \overline{p}_z \leq \frac{\binom{n-m}{k}}{\binom{n}{k}}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_C - \mathbf{O}_o$ such that $|\mathbf{O}_A| = \lceil \underline{p_y} \cdot \binom{n}{k} \rceil$. Let $\mathbf{O}_B \subseteq \mathbf{O}_o$ such that $|\mathbf{O}_B| = \lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor$. Figure 4(b) illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. We can construct a federated learning algorithm $\mathcal{A}^*$ as follows:

$$\mathcal{A}^*(s, \mathbf{x}) = \begin{cases} y, & \text{if } s \in \mathbf{O}_A \\ z, & \text{if } s \in \mathbf{O}_B \cup (\mathbf{O}'_C - \mathbf{O}_o) \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{35}$$

We can show that such $\mathcal{A}^*$ satisfies the following probability conditions:

$$p_y = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{36}$$

$$p_z = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, \tag{37}$$

which indicates $\mathcal{A}^*$ satisfies (5). However, we have:

$$\begin{aligned} p'_y - p'_z &= \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = y) - \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = z) && (38) \\ &= 0 - \frac{|\mathbf{O}_B| + |\mathbf{O}'_C - \mathbf{O}_o|}{|\mathbf{O}'_C|} && (39) \\ &= -\frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} - 1 + \frac{\binom{n-m}{k}}{\binom{n}{k}} && (40) \\ &< 0, && (41) \end{aligned}$$

which implies $h(\mathbf{C}', \mathbf{x}) \neq y$.

**Case 3:** $m^* < m < n - k$, $0 \leq \underline{p_y} \leq 1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, and $\frac{\binom{n-m}{k}}{\binom{n}{k}} \leq \overline{p}_z \leq 1 - \underline{p_y}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_C - \mathbf{O}_o$ and $\mathbf{O}_B \subseteq \mathbf{O}_C - \mathbf{O}_o$ such that $|\mathbf{O}_A| = \lceil \underline{p_y} \cdot \binom{n}{k} \rceil$, $|\mathbf{O}_B| = \lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor - \binom{n-m}{k}$, and $\mathbf{O}_A \cap \mathbf{O}_B = \emptyset$. Note that $|\mathbf{O}_C - \mathbf{O}_o| = \binom{n}{k} - \binom{n-m}{k}$, and we have:

$$\begin{aligned} |\mathbf{O}_A| + |\mathbf{O}_B| &= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} && (42) \\ &\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor (1 - \underline{p_y}) \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} && (43) \end{aligned}$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor \binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \right\rfloor - \binom{n-m}{k} \tag{44}$$

$$= \binom{n}{k} - \binom{n-m}{k}. \tag{45}$$

Therefore, we can always find a pair of such disjoint sets $(\mathbf{O}_A, \mathbf{O}_B)$. Figure 4(c) illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. We can construct an algorithm $\mathcal{A}^*$ as follows:

$$\mathcal{A}^*(s, \mathbf{x}) = \begin{cases} y, & \text{if } s \in \mathbf{O}_A \\ z, & \text{if } s \in \mathbf{O}_B \cup \mathbf{O}'_C \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{46}$$

We can show that such $\mathcal{A}^*$ satisfies the following probability conditions:

$$p_y = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{47}$$

$$p_z = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B| + |\mathbf{O}_o|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, \tag{48}$$

which are consistent with the probability conditions in (5). However, we can show the following:

$$p'_z = \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = z) = 1, \tag{49}$$

which gives $h(\mathbf{C}', \mathbf{x}) = z \neq y$.

**Case 4:** $m^* < m < n - k$, $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}} < \underline{p_y} \leq 1$, and $0 \leq \overline{p}_z \leq 1 - \underline{p_y} < \frac{\binom{n-m}{k}}{\binom{n}{k}}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_o$ and $\mathbf{O}_B \subseteq \mathbf{C}_o$ such that $|\mathbf{O}_A| = \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k}$, $|\mathbf{O}_B| = \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor$, and $\mathbf{O}_A \cap \mathbf{O}_B = \emptyset$. Note that $|\mathbf{O}_o| = \binom{n-m}{k}$, and we have:

$$|\mathbf{O}_A| + |\mathbf{O}_B| = \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor \tag{50}$$

$$\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left\lfloor (1 - \underline{p_y}) \cdot \binom{n}{k} \right\rfloor \tag{51}$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left\lfloor \binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \right\rfloor \tag{52}$$

$$= \binom{n-m}{k}. \tag{53}$$

Therefore, we can always find such a pair of disjoint sets $(\mathbf{O}_A,\ \mathbf{O}_B)$. Figure 4(d) illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. Next, we can construct an algorithm $\mathcal{A}^*$ as follows:

$$\mathcal{A}^*(s, \mathbf{x}) = \begin{cases} y, & \text{if } s \in \mathbf{O}_A \cup (\mathbf{O}_C - \mathbf{O}_o) \\ z, & \text{if } s \in \mathbf{O}_B \cup (\mathbf{O}'_C - \mathbf{O}_o) \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{54}$$

We can show that $\mathcal{A}^*$ has the following properties:

$$p_y = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A| + |\mathbf{O}_C - \mathbf{O}_o|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{55}$$

$$p_z = \Pr(\mathcal{A}^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, \tag{56}$$

which implies $\mathcal{A}^*$ satisfies the probability conditions in (5). However, we also have:

$$p'_y - p'_z = \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = y) - \Pr(\mathcal{A}^*(\mathbf{Y}, \mathbf{x}) = z) \tag{57}$$

$$= \frac{|\mathbf{O}_A|}{|\mathbf{O}'_C|} - \frac{|\mathbf{O}_B| + |\mathbf{O}'_C - \mathbf{O}_o|}{|\mathbf{O}'_C|} \tag{58}$$

$$= \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k}}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} + \binom{n}{k}}{\binom{n}{k}} \tag{59}$$

$$= \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} - \left[ 2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}} \right]. \tag{60}$$

Since $m > m^*$, we have:

$$\frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \le \left[ 2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}} \right]. \tag{61}$$

Therefore, we have $p'_y - p'_z \le 0$, which indicates $h(\mathbf{C}', \mathbf{x}) \neq y$ or there exist ties.

To summarize, we have proven that in any possible cases, Theorem 2 holds, indicating that our derived certified security level is tight.

## F Proof of Theorem 3

Based on the Clopper-Pearson method, for each testing example $\mathbf{x}_t$, we have:

$$\Pr(\underline{p_{\hat{y}_t}} \le \Pr(\mathcal{A}(\mathcal{S}(\mathbf{C}, k), \mathbf{x}_t) = \hat{y}_t) \wedge \overline{p}_{\hat{z}_t} \ge \Pr(\mathcal{A}(\mathcal{S}(\mathbf{C}, k), \mathbf{x}_t) = i), \forall i \neq \hat{y}_t) \ge 1 - \frac{\alpha}{d}. \tag{62}$$

Therefore, for a testing example $\mathbf{x}_t$, if our Algorithm 1 does not abstain for $\mathbf{x}_t$, the probability that it returns an incorrect certified security level is at most $\frac{\alpha}{d}$. Formally, we have the following:

$$\Pr((\exists \mathbf{C}', M(\mathbf{C}') \le \hat{m}^*_t, h(\mathbf{C}', \mathbf{x}_t) \neq \hat{y}_t) | \hat{y}_t \neq \text{ABSTAIN}) \le \frac{\alpha}{d}. \tag{63}$$

Therefore, we have the following:

$$\Pr(\cap_{\mathbf{x}_t \in \mathcal{D}}((\forall \mathbf{C}', M(\mathbf{C}') \le \hat{m}^*_t, h(\mathbf{C}', \mathbf{x}_t) = \hat{y}_t) | \hat{y}_t \neq \text{ABSTAIN})) \tag{64}$$

$$= 1 - \Pr(\cup_{\mathbf{x}_t \in \mathcal{D}}((\exists \mathbf{C}', M(\mathbf{C}') \le \hat{m}^*_t, h(\mathbf{C}', \mathbf{x}_t) \neq \hat{y}_t) | \hat{y}_t \neq \text{ABSTAIN})) \tag{65}$$

$$\ge 1 - \sum_{\mathbf{x}_t \in \mathcal{D}} \Pr((\exists \mathbf{C}', M(\mathbf{C}') \le \hat{m}^*_t, h(\mathbf{C}', \mathbf{x}_t) \neq \hat{y}_t) | \hat{y}_t \neq \text{ABSTAIN}) \tag{66}$$

$$\ge 1 - d \cdot \frac{\alpha}{d} \tag{67}$$

$$= 1 - \alpha. \tag{68}$$

We have (66) from (65) based on the Boole's inequality.