Federated Truth Discovery for Mobile Crowdsensing with Privacy-Preserving Trustworthiness Assessment

Leye Wang[1,2], Guanghong Fan[1,2], Xiao Han[3,*]
[1]Key Lab of High Confidence Software Technologies (Peking University), Ministry of Education, China
[2]School of Computer Science, Peking University, China
[3]School of Information Management and Engineering, Shanghai University of Finance and Economics, China
leyewang@pku.edu.cn, fgh@stu.pku.edu.cn, xiaohan@mail.shufe.edu.cn

Abstract

With the prevalence of smart mobile devices empowered by considerable sensing capabilities, crowdsensing has become one promising way to sense urban phenomena (e.g., traffic and environment) at a large scale. In crowdsensing, a fundamental issue is discovering the truth from participants' noisy sensed data. Traditionally, participants need to upload their raw sensed data with locations for truth discovery, but this may leak participants' private information such as home and work locations. In this paper, we propose a federated truth discovery method that can learn the truth without collecting participants' sensed data and locations. Our method ensures that the obtained truth quality has no performance loss compared to the original truth discovery method if all the participants keep online; even if some participants lose connections unpredictably, our method can still learn the truth based on rest participants' data. Meanwhile, as participants' sensed data are unknown to the server, it is hard for the crowdsensing organizer to justify each participant's sensing trustworthiness. This brings difficulties to crowdsensing management such as participant recruitment and incentive allocation. We further propose a federated ranking mechanism to generate a leader-board for participants' trustworthiness, which can also tolerate participants' connection loss. Both theoretical analysis and real-data empirical evaluations have been done to verify the effectiveness of FedTruthFinder.

1 Introduction

With the popularity of smart mobile devices, such as smartphones, pads, and vehicles, crowdsensing has become one promising paradigm for sensing urban dynamics [3]. A typical crowdsensing process first recruits participants and then asks them to upload the data of interest to the central server. Afterward, the server aggregates the data from participants toward a synthetic sensed result [39, 11]. While each participant's sensed data may include noise, an important issue is how to ensure the accuracy of the aggregated sensed result, often called *truth discovery* [26, 12, 24].

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

*Corresponding author

Many research studies have been devoted to the truth discovery for crowdsensing applications. Generally, most relevant studies model participant trustworthiness and sensed data confidence iteratively to obtain the final aggregated sensed result [36]. The basic idea is that a high-credit participant's sensed data should be assigned with high confidence; a user whose sensed data are more confident should be more trustworthy. While this has been verified to be effective, the iterative computation process needs a central server to collect every participant's raw sensed data, which may bring privacy threats to crowdsensing participants. For example, crowdsensing usually asks participants to do sensing tasks at specific locations, and thus most sensed data are associated with detailed location information [39]. The traditional truth discovery process would inevitably leak crowdsensing participants' visited locations during sensing periods, which may be exploited by malicious third parties to conduct serious location privacy breaches such as physical stalking [19]. With the user privacy and personal data regularization (e.g., General Data Protection Regulation[1]) becoming more and more important nowadays, we believe that a privacy-preserving truth discovery algorithm is urgently required for facilitating more extensive crowdsensing campaigns in practice.

Privacy-preserving truth discovery has been recently studied in crowdsensing. Existing studies are usually based on some hardly realized assumptions such as every participant being always online and/or non-colluding [13, 15, 14][2], or two non-colluding servers (or fog nodes) are required [23, 40, 38, 37]. These assumptions hinder the practical applications of the prior mechanisms. More importantly, almost all the prior studies do not discuss two fundamental issues in privacy-preserving truth discovery.

i) **Participants' Completed Tasks Protection.** Existing mechanisms focus on protecting participants' sensed data, but most assume that participants' completed tasks are known to the crowdsensing server [34, 13, 14, 40]. However, sometimes task information is more sensitive than sensed data. Suppose the tasks are air quality sensing at certain points of interest. If a participant's task completion information is disclosed (e.g., finish a sensing task at the Times Square NYC), then her location is revealed without the need to know her sensed air quality value.

ii) **Participants' Trustworthiness Assessment.** Trustworthiness assessment is a key part of crowdsensing for participant recruitment and incentive design [17], while privacy-preserving truth discovery needs to hide participants' trustworthiness scores for privacy protection. To address the dilemma, a privacy-preserving trustworthiness assessment method needs to be proposed.

In this paper, we aim to design a novel and practical privacy-preserving truth discovery mechanism for crowdsensing to overcome the pitfalls of prior studies. In particular, our design follows the federated learning (FL) paradigm [5, 35]. We thus call our method as *FedTruthFinder*. In general, FL requires user clients to do some local computation (e.g., learning the gradients for updating the parameters of the model) on their devices and then only upload the computation results instead of raw data. For a specific algorithm, the local computation and uploading process usually incorporates certain secure mechanisms (e.g., homomorphic encryption and secure multi-party computation) to ensure that no user privacy is leaked theoretically [2, 10]. Based on FL, we aim to consider the following specific issues in the design of FedTruthFinder.

No Accuracy Loss: We expect that FedTruthFinder will not hurt the accuracy of the aggregated sensed results compared to the centralized truth discovery. Without the loss of accuracy, crowdsensing organizers would be likely to adopt the method in practice.

No Third Party: Crowdsensing involves a central server and a set of participants' clients. To make FedTruthFinder easy to deploy, we do not want to introduce any more third parties which are usually hard to find in reality [1, 27].

Robustness against Unpredictable Connection Loss: While crowdsensing participants travel around the whole city, their device connection with the central server is not always stable. Hence, it is necessary to make FedTruthFinder effective when some participants lose connections suddenly.

[1] `https://gdpr.eu/eu-gdpr-personal-data`

[2] Some participants' relations can be very close such as family members, and thus it is non-reasonable to assume that they will not collude with each other.

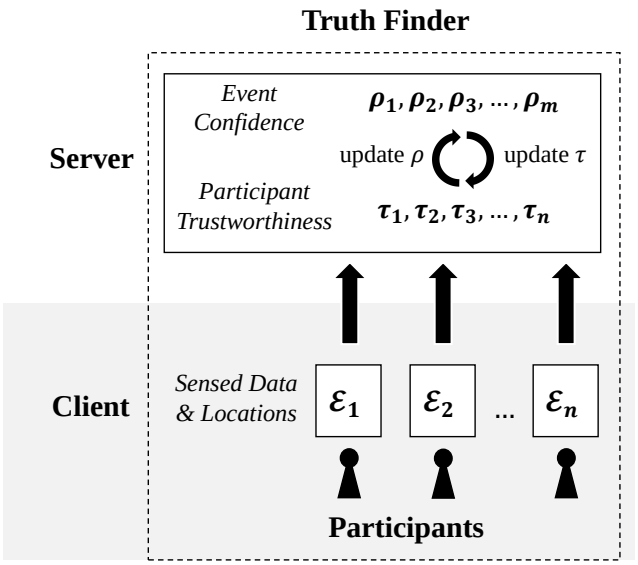


Figure 1: Overview of Iterative Truth Discovery

With the above issues in consideration, this paper makes the following contributions:

(1) To the best of our knowledge, this paper is the first study that addresses the problem of privacy-preserving crowdsensing truth discovery with (i) *participants' completed task protection*, and (ii) *participants' trustworthiness assessment*.

(2) We propose *FedTruthFinder*, a novel privacy-preserving truth discovery mechanism following the federated learning paradigm [35]. FedTruthFinder does not need any third party and can tolerate participants' unpredictable connection loss. The two key components of FedTruthFinder are (i) *federated confidence computation* to learn the probability of a sensed event, and (ii) *federated trustworthiness ranking* to assess participants' sensed data quality.

(3) We have conducted both theoretical analysis and empirical evaluations for FedTruthFinder. In particular, FedTruthFinder can reduce the system failure probability significantly compared to state-of-the-art privacy-preserving truth discovery approaches [1, 34], and achieve good detection accuracy.

2 Preliminary: Truth Discovery

Truth discovery algorithms usually follow an iterative method to calibrate user trustworthiness and data confidence alternatively until convergence [36]. Figure 1 shows the framework of iterative truth discovery methods. In this paper, for clarity, we assume that sensed data is a binary spatial event. That is, for a specific location, the sensed data can be 1 or 0. Our method can be easily extended to multi-class and continuous-value events (see Appendix).

As shown in Figure 1, first, participants upload all of their sensed data and locations $\mathcal{E}_i$ to the central server. The central server would assign an initial trustworthiness score τ_i to each participant u_i (e.g., 0.9 by assuming that 90% of the sensed data are accurate). Then, for each sensed event e_j, the truth discovery algorithm will calculate its confidence ρ_j (i.e., the probability of $e_j = 1$) by considering the users who have sensed e_j as:

$$\rho_j = F_\rho(\mathcal{U}_{j,1}, \mathcal{U}_{j,0}) \tag{73}$$

where $\mathcal{U}_{j,k}$ is the users who have sensed the event e_j with the reported data k; F_ρ is an event confidence calculation function which we will elaborate on later.

With ρ_j for each event e_j, we can then update the trustworthiness score τ_i of each participant u_i by:

$$\tau_i = F_\tau(\mathcal{E}_{i,1}, \mathcal{E}_{i,0}) \tag{74}$$

where $\mathcal{E}_{i,k}$ is the users' sensed event set with the reported data k; F_τ is a user trustworthiness calculation function which we will elaborate on later.

Once τ_i is updated for each user u_i, we can continue updating ρ_j for each event e_j according to Eq. 73, and so on, leading to an alternative updating process for both τ_i and ρ_j. This process can be terminated after a fixed number of iterations or until convergence. Next, we elaborate on the common choices of F_ρ and F_τ in literature.

Sum Function

An intuitive selection of the updating functions of F_ρ and F_τ is the weighted sum:

$$\rho_j = F_\rho(\mathcal{U}_{j,1}, \mathcal{U}_{j,0}) = \frac{\sum_{u_i \in \mathcal{U}_{j,1}} \tau_i}{\sum_{u_i \in \mathcal{U}_{j,1}} \tau_i + \sum_{u_k \in \mathcal{U}_{j,0}} \tau_k} \tag{75}$$

$$\tau_i = F_\tau(\mathcal{E}_{i,1}, \mathcal{E}_{i,0}) = \frac{\sum_{e_j \in \mathcal{E}_{i,1}} \rho_j + \sum_{e_k \in \mathcal{E}_{i,0}} 1 - \rho_k}{|\mathcal{E}_{i,1}| + |\mathcal{E}_{i,0}|} \tag{76}$$

Logistic Function

Another widely used updating function is the Logistic function [36]. Its basic idea is seeing every user independently, so that the probability of event happening, i.e., $e_j = 1$, can be formulated as:

$$\rho_j = 1 - \prod_{u_i \in \mathcal{U}_{j,1}} (1 - \tau_i) \tag{77}$$

As $1 - \tau_i$ may often be small and multiplying many of them may lead to underflow, prior studies proposed to use the logarithm to define a log-trustworthiness score of u_i as [36]:

$$\tau_i^* = -\ln(1 - \tau_i) \tag{78}$$

Similarly, a log-confidence score of event e_j is defined as:

$$\rho_j^* = -\ln(1 - \rho_i) \tag{79}$$

Then, we can infer

$$\rho_j^* = \sum_{u_i \in \mathcal{U}_{j,1}} \tau_i^* \tag{80}$$

The above equation does not consider the users' trustworthiness who report $e_j = 0$, and thus we refine it:

$$\rho_j^* = \sum_{u_i \in \mathcal{U}_{j,1}} \tau_i^* - \sum_{u_k \in \mathcal{U}_{j,0}} \tau_k^* \tag{81}$$

Finally, a logistic function is used to calculate the final confidence ρ_j of event e_j [36]:

$$\rho_j = F_\rho(\mathcal{U}_{j,1}, \mathcal{U}_{j,0}) = (1 + e^{-\rho_j^*})^{-1} \tag{82}$$

τ_i is updated same as Eq. 76.

3 Federated Truth Discovery: Overview and Key Issues

3.1 Overall Design

Figure 2 overviews the workflow of our method *FedTruthFinder*. The general design principle follows the federated learning paradigm [35], which requires the user clients to conduct local computations of their raw

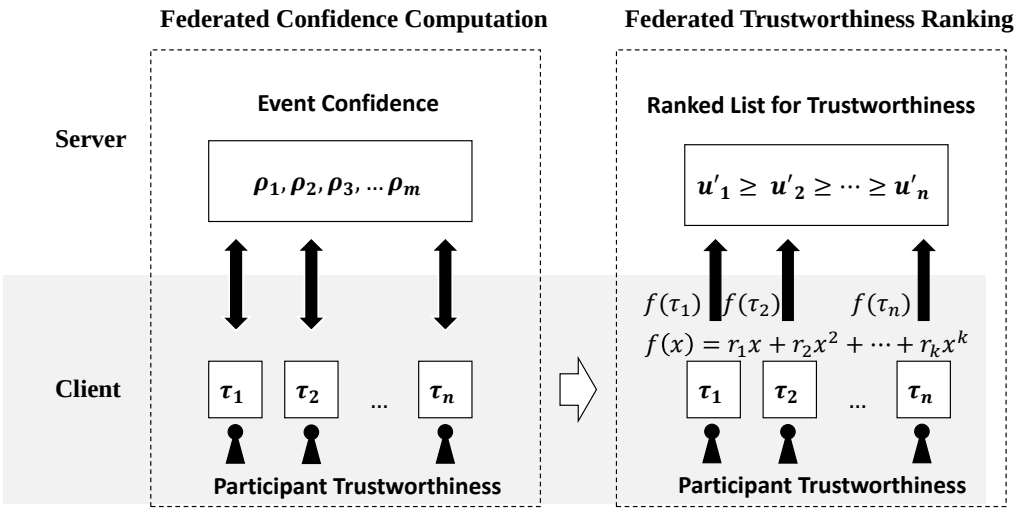


Figure 2: Overview of FedTruthFinder

data and then upload processed data that do not reveal the user's privacy to the server. With these uploaded privacy-preserving data, the server can still find the aggregate truth of the sensed events, the same as the server receiving the raw sensed data and locations from participants.

By analyzing the original truth discovery algorithm in the previous section, we note that there exist two alternative computation processes: (i) *ρ-computation*: updating the confidence ρ_j for each event e_j, and (ii) *τ-computation*: updating the trustworthiness τ_i for each participant u_i. In the two computation processes, τ-computation (e.g., Eq. 76) can be naturally offloaded to each participant u_i's device, as long as the server sends all the current $\rho_j, \forall e_j$ to the participants. However, ρ-computation needs to know each user's sensed data (and locations) and then do aggregation (e.g., sum). This needs a dedicated design to enable the privacy-preserving truth discovery, which will be illustrated in Sec. 4.

Besides, the trustworthiness of each participant's sensed data (i.e., τ_i) is a key metric in crowdsensing organization for participant recruitment and incentive allocation. Hence, we also design a federated privacy-preserving mechanism to rank participants' trustworthiness. Particularly, instead of transferring raw τ_i to the server, we leverage certain security mechanisms to upload $f(\tau_i)$ to the server, while $f(\tau_i)$ keeps the same ranking orders as τ_i. Particularly, in FedTruthFinder, $f(\tau_i) = r_1\tau_i + r_2\tau_i^2 + ... + r_k\tau_i^k$, where $r_i > 0$. In this regard, even though the server cannot know the specific τ_i of each participant u_i, the ranked list of participants according to the trustworthiness can still be learned with $f(\tau_i)$. Specifically, during the whole computation process, the server cannot know r_i, and each participant will also not know all the r_i, so that none of the server or participant can infer other participants' private τ_i. How to compute $f(\tau_i)$ securely will be introduced in Sec. 5.

3.2 Key Issues

Issue 1. Privacy-Preserving ρ-computation: *Suppose there are a set of crowdsensing participants $\mathcal{U}$ and a set of spatial events to sense $\mathcal{E}$, each participant $u_i(\in \mathcal{U})$ with sensed events $\mathcal{E}_{i,1}(\subseteq \mathcal{E})$ and $\mathcal{E}_{i,0}(\subseteq \mathcal{E})$ corresponding to the sensed value being 1 and 0, respectively. How to calculate confidence ρ_j for each event $e_j(\in \mathcal{E})$ while every participant u_i will not leak $\mathcal{E}_{i,1}$, $\mathcal{E}_{i,0}$, and $\mathcal{E}_{i,1} \cup \mathcal{E}_{i,0}$ to the server and other participants?*

Some factors need to be carefully considered:

(1) **Computation**: In ρ-computation, the value to share is a complicated equation instead of a single value, and the equation may even be varied depending on the truth discovery algorithm implementation (e.g., Eq. 75 or

Eq. 82).

(2) **Network Connection**: In crowdsensing, the network connections of mobile participants may not be always stable. Hence, our mechanism should tolerate the scenario when a few participants lose connections.

(3) **No Leakage of Task Completion**: For protecting participants' privacy, not only the sensed data but also the completed tasks (i.e., $\mathcal{E}_{i,1} \cup \mathcal{E}_{i,0}$) should not be disclosed.

Issue 2. Secure Trustworthiness Ranking: *Suppose there are a set of crowdsensing participants $\mathcal{U}$ and each participant $u_i (\in \mathcal{U})$ has a private trustworthiness score τ_i. How to rank participants according to τ_i while every participant u_i will not leak τ_i to the server and other participants?*

Addressing this issue also needs to consider the unstable network connections of participant clients as aforementioned.

Remark on security definition: In this work, we assume that the crowdsensing server and participants are *semi-honest (honest-but-curious)*: they will follow our designed protocol and not maliciously modify the inputs or outputs; however, the server and the participants will try their best to infer others' data from the data that they have received. Besides, our mechanism can defend against *collusion attacks* to a certain extent (i.e., some participants may collude with each other), which we will elaborate on later.

4 Federated Truth Computation

We first introduce a basic scheme for ρ-computation in a federated manner assuming no connection loss. Then, we improve the scheme to be against the participants' unpredictable connection loss.

4.1 Basic Scheme of ρ-Computation with SSS

In this section, we propose our basic scheme for the ρ-computation problem with the federated learning paradigm leveraging secret sharing. Given a spatial crowdsensing event e_j, we first consider ρ-computation with the sum function, i.e., Eq. 75.

4.1.1 ρ-computation with the sum function.

Our process includes three steps as follows:

Step 1 (Share Dispatching). Each participant u_i dispatches d_{ij} and s_{ij} with secret sharing to all the n participants ($\mathcal{U} = \{u_1 \cdots u_n\}$), where

$$d_{ij} = \begin{cases} \tau_i & e_j \in \mathcal{E}_{i,1} \\ 0 & e_j \in \mathcal{E} \setminus \mathcal{E}_{i,1} \end{cases} \tag{83}$$

$$s_{ij} = \begin{cases} \tau_i & e_j \in \mathcal{E}_{i,1} \cup \mathcal{E}_{i,0} \\ 0 & e_j \in \mathcal{E} \setminus (\mathcal{E}_{i,1} \cup \mathcal{E}_{i,0}) \end{cases} \tag{84}$$

Specifically, d_{ij} is divided into n shares

$$\{d_{ij}^1, \cdots, d_{ij}^n\}$$

where $d_{ij}^1, \cdots, d_{ij}^{n-1}$ are random numbers and

$$d_{ij}^n = d_{ij} - \sum_{k=1}^{n-1} d_{ij}^k$$

Hence, $\sum_{k=1}^{n} d_{ij}^k = d_{ij}$. Then, u_i sends d_{ij}^k to u_k. Similarly, s_{ij} is split to n shares

$$\{s_{ij}^1, \cdots, s_{ij}^n = s_{ij} - \sum_{k=1}^{n-1} s_{ij}^k\}$$

and u_k receives s_{ij}^k from u_i.

Step 2 (Client Summation). For each event e_j, an participant u_k uploads $\hat{d}_j^k = \sum_{i=1}^{n} d_{ij}^k$ and $\hat{s}_j^k = \sum_{i=1}^{n} s_{ij}^k$ to the server.

Step 3 (Server Aggregation). After receiving $\hat{d}_j^k$ and $\hat{s}_j^k$ from $\forall u_k \in \mathcal{U}$, the server can add them together:

$$d_j = \sum_{k=1}^{n} \hat{d}_j^k = \sum_{k=1}^{n}\sum_{i=1}^{n} d_{ij}^k = \sum_{i=1}^{n}\sum_{k=1}^{n} d_{ij}^k = \sum_{i=1}^{n} d_{ij} \tag{85}$$

$$s_j = \sum_{k=1}^{n} \hat{s}_j^k = \sum_{k=1}^{n}\sum_{i=1}^{n} s_{ij}^k = \sum_{i=1}^{n}\sum_{k=1}^{n} s_{ij}^k = \sum_{i=1}^{n} s_{ij} \tag{86}$$

Then, ρ_j is computed as:

$$\rho_j = \frac{d_j}{s_j} \tag{87}$$

4.1.2 ρ-computation with the logistic function.

The computation process with the logistic function is not much different from the one with the summation function. Actually, for the event e_j, we only need to modify d_{ij} to:

$$d_{ij} = \begin{cases} -\ln(1-\tau_i) & e_j \in \mathcal{E}_{i,1} \\ \ln(1-\tau_i) & e_j \in \mathcal{E}_{i,0} \\ 0 & e_j \in \mathcal{E} \setminus (\mathcal{E}_{i,1} \cup \mathcal{E}_{i,0}) \end{cases} \tag{88}$$

Besides, we do not need s_{ij}, so every participant u_k only receives d_{ij}^k from other $u_i \in \mathcal{U}$. Finally, in Step 3, we can compute ρ_j as:

$$\rho_j = (1 + e^{-d_j})^{-1} \tag{89}$$

Remark on our novelty. In our ρ-computation for an event e_j, the participant u_i who has not sensed e_j also needs to upload data to the server, e.g., $d_{ij} = s_{ij} = 0$ for the sum function. As d_{ij} and s_{ij} are sent by secret shares, the other participants and the server would not know whether u_i senses e_j or not. In comparison, prior studies usually assume that participants send only the data of their sensed events, which may disclose user privacy from event information (e.g., event locations) [13, 15, 14, 40, 41, 38].

4.2 Connection Robustness Improvement

The basic scheme can learn ρ_j in an ideal environment when all the participants are always online. In practice, participants move around and their network connections are often sporadic. This inspires us to make three improvements to the basic scheme design.

4.2.1 Bias-avoidance adaptive truth updating.

While participants may drop during the iterative truth discovery process due to bad connections, the original event confidence updating function would be ineffective. Specifically, if u_i loses the connection at the k^{th} iteration, u_i's data would not be considered in the ρ-computation (e.g., Eq. 75) from then on. This leads to unreliable ρ_j as the finally alive participants dominate the results. To address this pitfall, we propose an adaptive updating function for k^{th} iteration's $\rho_{j,k}$ as,

$$\rho_{j,k} = w_k F_\rho + (1 - w_k)\rho_{j,k-1} \tag{90}$$

where F_ρ is an original event confidence updating function (e.g., sum and logistic). With Eq. 90, the data contribution of the participants who drop at the k^{th} iteration can still be kept (by $\rho_{j,k-1}$) to avoid the truth bias toward alive participants. w_k can be set as,

$$w_k = (\frac{|\mathcal{U}_{alive,k}|}{|\mathcal{U}|})^\alpha \tag{91}$$

where $\mathcal{U}_{alive,k}$ is the alive participants at the k^{th} iteration. When alive participants decrease with more iterations, w_k becomes smaller, reflecting that fewer participants should occupy lower weights. From our experiments, we find that $\alpha = 3$ is a proper setting.

4.2.2 Server-coordinated communication structure.

In Sec. 4.1, we assume that one participant u_i can establish a secure communication channel with every other participant u_k so as to transfer the secret share d_{ij}^k and s_{ij}^k. Hence, each participant needs to establish $n-1$ channels with others. Considering the sporadic property of the mobile connections, this may not be easy for a participant to keep so many channels stable in practice. To alleviate this issue, we convert the peer-to-peer communication structure to a server-coordinated one. In the server-coordinated structure, every participant first transmits all the data to the server, and then the server dispatches the desired data to the corresponding participant. In this way, each participant needs to establish *only one* secure channel to the server.

To ensure that the transmitted data will not be directly observed by the server, we leverage a public-key encryption system to encrypt the data before the transmission. In particular, each participant u_i first generates a pair of keys, the public key pk_i and the private key sk_i. The public key pk_i is sent to all the other participants (e.g., through the server) at the beginning of the crowdsensing campaign. For details, readers can refer to [1].

Then, for computing ρ_j, each participant u_i first transmits $E_i = \{\textit{Encrypt}(d_{ij}^k, pk_k)|k = 1 \cdots n\}$ to the server.[3] After receiving n participants' E_i, the server re-organizes the received data and sends $\hat{E}_k = \{\textit{Encrypt}(d_{ij}^k, pk_k)|i = 1 \cdots n\}$ to each participant u_k. Afterward, u_k decrypts the received data with her private key, obtains $\{d_{ij}^k|i = 1 \cdots n\}$, and then uploads $\hat{d}_j^k = \sum_i d_{ij}^k$ to the server. The server can then recover $d_j = \sum_i d_{ij}$ from participants' uploaded $\hat{d}_j^k$ and computes ρ_j accordingly.

4.2.3 (t, n)-Shamir secret sharing (SSS)

While the server-coordinated communication structure reduces the burden of secure channel establishing for mobile participants. It may still fail if a user loses the connection during the campaign and cannot link back. For example, suppose a participant u_i has sent E_i to the server and then quit the crowdsensing campaign (e.g., u_i's mobile device runs out of battery). Then, in Step 3, the server will not be able to receive $\hat{d}_j^i$ from u_i, and thus cannot recover d_j.

To address this pitfall, in practical deployment, we can leverage the threshold secret sharing method proposed by Shamir [21], namely (t, n)-*Shamir secret sharing (SSS)*. With (t, n)-SSS, the server only needs to receive

[3] If s_{ij} is needed (e.g., for the sum function), it can be encoded in E_i same as d_{ij}.

t ($t \leq n$) participants' $\hat{d}_j^i$ for recovering d_j. In particular, to leverage (t, n)-SSS to dispatch d_{ij}, we first create a $(t-1)$-polynomial:

$$D_{ij}(x) = d_{ij} + a_{ij1}x + a_{ij2}x^2 + \cdots + a_{ijt-1}x^{t-1} \tag{92}$$

where $a_{ij1}, \cdots, a_{ijt-1}$ are random numbers selected by u_i. Then, u_i dispatches $D_{ij}(k)$ to u_k. If we obtain more than t participants' $D_{ij}(k)$, according to linear algebra, we can infer d_{ij}.

Similar to Step 2 of our basic scheme, $\hat{d}_j^k = \sum_i D_{ij}(k)$ is uploaded to the server by each u_k. Then, in Step 3, after receiving more than t participants' $\hat{d}_j^k$, the server will be able to infer $d_j = \sum_i d_{ij}$ according to the *additive homomorphism property* of SSS [21].

Remark on our novelty. In the truth discovery part, the key advantage of our mechanism beyond literature is its robustness against connection loss. Preliminary privacy-preserving truth discovery papers rarely consider the connection loss issue [15]. Some work tries to deal with drop-out users by letting alive participants send extra information [38, 31]; however, when the connection condition is so bad that certain alive participants again lose connections during the extra information communication, this process would be uncontrolled and time-consuming. Recent work also adopts SSS to enhance connection robustness [34]. The basic idea is using the double-masking secure aggregation algorithms proposed by [1], and every participant needs *two* connections to do event confidence computation for one iteration. In comparison, our mechanism only needs every participant to connect *once* for one iteration of computation. Our numerical experiments (Sec. 6.1) will show that this connection reduction can lead to a significantly difference in the algorithm success probability (e.g., increasing the success probability from 1% to 99% under certain conditions).

4.3 Theoretical Analysis

4.3.1 Correctness

The process to calculate ρ_i in FedTruthFinder follows the original algorithm shown in Sec. 2. Hence, we can obtain the same aggregate truth results as the original algorithm, as long as the SSS scheme is valid. In this regard, the correctness of our algorithm is theoretically guaranteed.

4.3.2 Robustness to Connection Loss & Security

Setting t to a small value allows our mechanism to tolerate more users dropping the campaign due to connection losses. Meanwhile, a small t reduces the security level of our mechanism — if t participants collude with each other, they can recover the other participants' sensed data and locations, leading to privacy leakage.

Theorem 4.1. If there are $\leq n - t$ participants losing the connection in one iteration of ρ-computation, the server can learn the event confidence ρ_j.

Theorem 4.2. If $t'(< t)$ semi-honest users collude with each other, they cannot infer any other users' secret information.

The two theorems hold based on the property of (t, n)-SSS.

4.3.3 Complexity

We analyze the algorithm from both communication and computation complexity perspectives. Particularly, since the participant clients are more sensitive to the communication and computation overhead, our current analysis focuses on the client side, while the server part analysis is similar.

Communication Complexity - $O(nn_e)$. For each participant client, she needs to transfer n share pieces of the secret to the other participants and receive the corresponding shares from every other participant, so the complexity is $O(n)$ for one event. Suppose there are n_e events, the total communication complexity is $O(nn_e)$.

Computation Complexity - $O(nn_e)$. Each client needs to do two local computation processes. The first process is to generate the random coefficients for (t, n)-SSS and calculate the secret shares sent to all the other participants (Step 1), which is $O(nn_e)$. The second process is to do local summation (Step 2), which is also $O(nn_e)$. Hence, the total computation complexity is $O(nn_e)$.

5 Federated Trustworthiness Rank

While FedTruthFinder learns the integrated event truth in a privacy-preserving manner, it brings a challenge in justifying participants' trustworthiness. For example, to incentivize the crowdsensing participants, it is a common strategy to pay the high-trustworthy participants (i.e., high-quality sensing results) with higher incentives. However, in FedTruthFinder, the sensing quality of each participant, i.e., the trustworthiness score τ_i is kept at each participant side and unknown to the server. Hence, how to assess participants' trustworthiness is required and challenging for FedTruthFinder.

In this section, we first illustrate a concrete case to describe that τ_i cannot be directly known to the server, otherwise the server may infer which event u_i has sensed. As τ_i cannot be known to the server, we then design a secure ranking algorithm to let the server know every participant u_i's ranking position of τ_i among all the participants without leaking τ_i. Based on the ranked positions, the crowdsensing organizer can enable certain trustworthiness-aware incentive mechanisms, e.g., rewarding high-position participants with bonus, which can incentivize participants to compete with each other to get more high-quality sensed data [20].

5.1 Privacy Leakage by Trustworthiness τ_i

Here, we illustrate an example to show the risk of revealing τ_i to the server for leaking participant u_i's privacy.

Without the loss of generality, we assume that u_1's $\tau_1 = 0.9$, and other u_i's $\tau_i < 0.9$ $(i \neq 1)$. Suppose that one event e_j's $\rho_j = 0.9$ after truth discovery, then we can easily infer that u_1 has sensed the event e_j and the sensed result is 1. This reveals the fact that u_1 has visited the location of e_j, leaking u_1's location privacy.

Hence, participants cannot directly upload their τ_i to the server for incentive allocation. Next, we design a privacy-preserving method to enable trustworthiness-aware incentive allocation.

5.2 Secure Trustworthiness Leader-board

While revealing τ_i may leak participants' private information, we propose a secure ranking algorithm to learn a leader-board regarding participants' trustworthiness for facilitating trustworthiness-aware incentive allocation.

Secure ranking algorithms have been studied for decades; however, prior studies cannot be directly applied in our scenario for two reasons. First, the communication overheads are usually high. Second, prior studies mostly assume that all the network connections are stable for all the parties, but this is unrealistic for crowdsensing.

Our secure ranking algorithm generally follows the design of [22]. However, the original design [22] cannot tolerate any participants to lose the network connections. We thus enhance it to ensure that the ranking algorithm can still work when certain participants lose connections. The major steps of our federated trustworthiness leader-board generation mechanism are:

Step 1. First, we categorize all the participants into $(2t + 1)$ groups, and thus each group includes $n/(2t + 1)$ participants. We denote $gid(u)$ to refer to the group ID of participant u.

Step 2. For each user u_i, she shares τ_i, τ_i^2, ... , τ_i^{2t+1} with $(t + 1, 2t + 1)$-SSS to all the user groups. Specifically, a user u_j will receive the share piece regarding $gid(u_j)$, denoted as $\tau_{i_1}(gid(u_j))$, $\tau_{i_2}(gid(u_j))$, ... $\tau_{i_{2t+1}}(gid(u_j))$ for τ_i, τ_i^2, ... , τ_i^{2t+1}, respectively.

Step 3. For each user group g_k, it generates a random number $r_k(> 0)$ and shares r_k with $(t + 1, 2t + 1)$-SSS to all the user groups. That is, u_j will receive r_k's share regarding $gid(u_j)$, denoted as $r_k(gid(u_j))$.

Step 4. For each participant u_j, she calculates the following number with the $\tau_{i_k}(gid(u_j))$ received from u_i:

$$h_i(gid(u_j)) = \lambda(gid(u_j)) \sum_{k=1}^{2t+1} r_k(gid(u_j))\tau_{i_k}(gid(u_j)) \tag{93}$$

$$= \lambda(gid(u_j))\gamma(gid(u_j)) \tag{94}$$

where

$$\begin{pmatrix} 1 & 1 & 1^2 & ... & 1^{2t} \\ 1 & 2 & 2^2 & ... & 2^{2t} \\ ... & ... & ... & ... & ... \\ 1 & 2t+1 & (2t+1)^2 & ... & (2t+1)^{2t} \end{pmatrix}^{-1}$$

$$= \begin{pmatrix} \lambda(1) & \lambda(2) & \lambda(3) & ... & \lambda(2t+1) \\ ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... \end{pmatrix}$$

Step 5. For each user group, we randomly select one participant u_j to share $\{h_i(gid(u_j))|i \in [1,n]\}$ with $(t+1,n)$-SSS to all the n participants. Each user u_k's received shares from all the groups are denoted as $\{h_i(g,k)|i \in [1,n], g \in [1,2t+1]\}$.

Step 6. For each participant u_k, she computes:

$$h'_i(k) = \sum_{g=1}^{2t+1} h_i(g,k), \quad \forall i \in [1,n] \tag{95}$$

Each u_k sends $\{h'_i(k)|i \in [1,n]\}$ to the server.

Step 7. After receiving at least $t+1$ participants' $\{h'_i(k)|i \in [1,n]\}$, the server can recover:

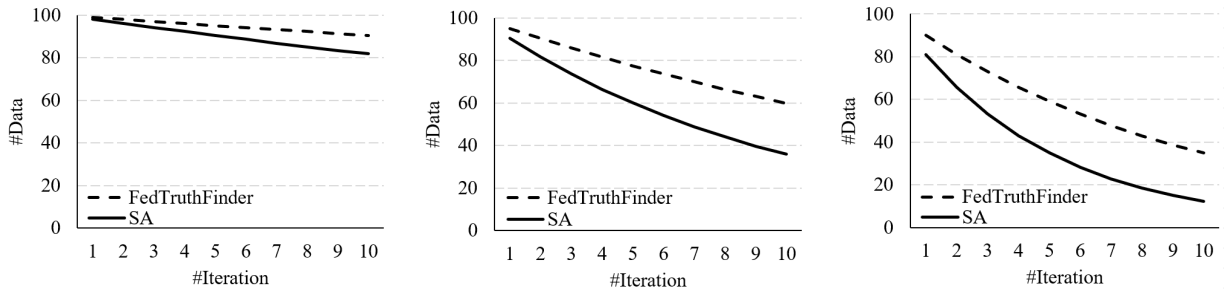
$$h_i = \sum_{k=1}^{2t+1} r_k \tau_i^k, \quad \forall i \in [1,n] \tag{96}$$

Step 8. The server ranks u_i according to h_i and the ranked list is the leader-board regarding trustworthiness τ_i.

Note that same as ρ-computation, we do not need to establish the peer-to-peer communication channels between every two participant clients and can use the crowdsensing server for coordination. To avoid redundancy, readers can refer to Sec. 4.2.2 for details.

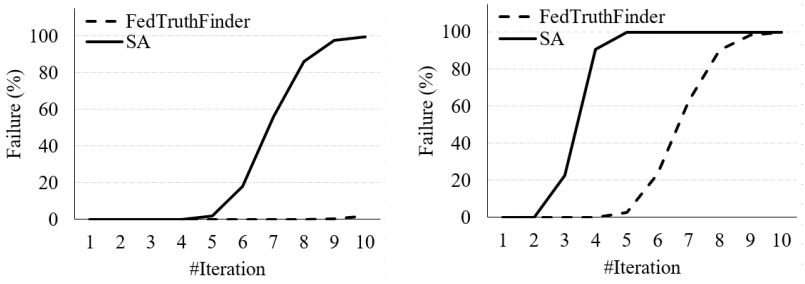
Remark on our novelty. The key improvement of our secure ranking algorithm compared to [22] is the enhanced robustness against participants' connection loss. In [22], every participant holds a r_i and we will randomly select $2t+1$ participants to share their r_i (Step 3) and h_i (Step 5). This process is easy to break if a selected online user (Step 3) loses the connection in Step 5. Our proposed algorithm first constructs user groups so that we only need at least one participant online in each group for both Step 3 and 5, reducing the failure possibility incurred by connection loss. It is worth noting that this algorithm can not only rank crowdsensing participants' trustworthiness, but also be applied to many other applications when privacy-preserving data ranking is needed under unstable network connections.

Remark on the ranked measurements. In the previous algorithm description, we suppose that τ_i needs to be ranked. In practice, crowdsensing organizers can use the same secure ranking mechanism to rank other key measurements of participants (e.g., the number of sensed events) to design better incentive mechanisms or participant recruitment strategies.



(a) $p_l = 0.01$ (b) $p_l = 0.05$ (c) $p_l = 0.1$

Figure 3: Number of data for each event's truth discovery by iterations.



(a) $p_l = 0.05$ (b) $p_l = 0.1$

Figure 4: Failure probability of truth discovery.

5.3 Theoretical Analysis

All the proofs are illustrated in Appendix.

5.3.1 Correctness

We first prove the correctness of our algorithm.

Lemma 5.1. $\sum_{k=1}^{2t+1} r_k(x)\tau_{i_k}(x)$ can be represented as:

$$h_i + a_{i1}x + a_{i2}x^2 + ... + a_{i2t}x^{2t}$$

where $h_i = \sum_{k=1}^{2t+1} r_k \tau_i^k$. [22]

Theorem 5.1. With $t+1$ participants' $h_i'(k)$, we can recover h_i.

Theorem 5.2. Ranking h_i is equivalent to ranking τ_i.

5.3.2 Robustness to Connection Loss

We analyze how our secure ranking algorithm can tolerate connection losses. We assume that before Step 2, there is no user connection loss.[4]

[4]If u_i loses the connection in Step 2 and cannot share τ_i^k with SSS, then there is no way to rank u_i's position because the server has no u_i's information.

Theorem 5.3. To finish Step 3-5, there needs at least one user online for each group. Suppose that every user has p_l probability to lose connection and there are n users, the success probability $\geq (1 - p_l^{\lfloor n/(2t+1) \rfloor})^{2t+1}$.

Theorem 5.4. To finish Step 6-8, $\geq t + 1$ users need to be online.

5.3.3 Security

Here, we analyze the security of our mechanism.

Theorem 5.5 If there are no more than t collusive participants, then these participants cannot recover all the other users' τ_i.

5.3.4 Complexity

We analyze the algorithm from communication and computation complexity perspectives for participant clients.

Communication Complexity - $O(tn)$. In Step 2, the communication overhead of one participant to send $\tau_i, \tau_i^2, ..., \tau_i^{2t+1}$ is $O(t^2)$, while each user received data is $O(tn)$. In Step 3, the complexity is $O(t)$. In Step 5, for sending data, the complexity is $O(n)$; for receiving data, the complexity is $O(tn)$. In Step 7, the complexity is $O(n)$. Combing them together, the communication complexity of the whole process is $O(tn)$ as $t < n$.

Computation Complexity - $O(tn)$. The main computation processes of each client include (1) calculating secret shares for $\tau_i, \tau_i^2, ..., \tau_i^{2t+1}$ with $(t + 1, 2t + 1)$-SSS in Step 2, which is $O(t^2)$, (2) calculating secret shares of r_k in Step 3, which is $O(t)$, (3) computing h_i in Step 4, which is $O(tn)$, and (4) calculating h_i' in Step 6, which is $O(tn)$. Hence, the final computation complexity is $O(tn)$.

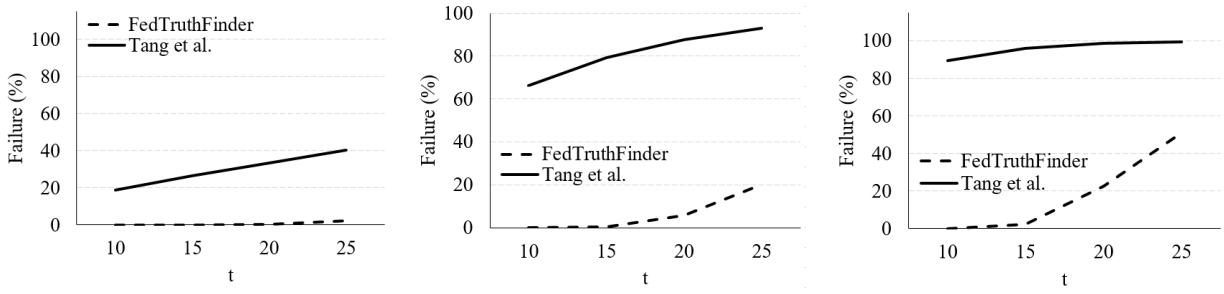
6 Evaluation

6.1 Numerical Analysis for Connection Loss

We have theoretically proven that our algorithm can learn the event confidence and trustworthiness ranking like the original centralized algorithms. This experiment then focuses on how the connection loss would impact FedTruthFinder quantitatively, since the unstable mobile network connection is a key characteristic for mobile crowdsensing. A practical mechanism should be able to fight against the unpredictable connection loss. In general, participants' connection loss may bring two types of negative impacts to the iterative truth discovery algorithm.

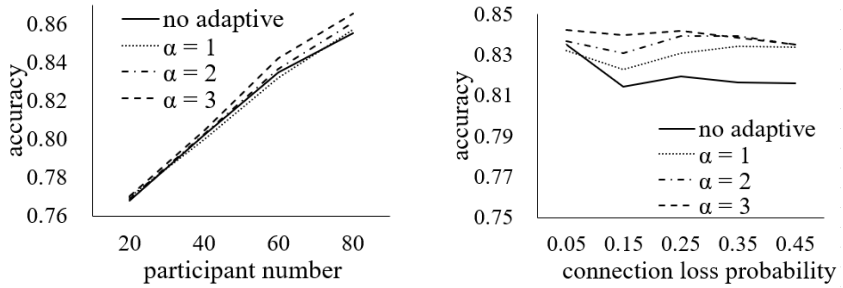
- **A small number of sensed data for truth discovery.** While FedTruthFinder can learn an aggregate truth as long as more than t participants are online, the data sources for the truth would be decreased. This would also affect the performance of the learned truth.
- **Possible failure of the whole algorithm.** If a large number of participants lose the connection and only fewer than t participants remain online, then the whole running process of FedTruthFinder would fail and no result can be learned.

Specifically, we conduct the numerical analysis for two parts of FedTruthFinder respectively, i.e., event confidence computation and participant trustworthiness ranking. We vary the probability of one participant losing the connection (denoted as p_l). If $p_l = 0.01$, a participant has 1% probability of dropping out of the crowdsensing campaign due to one-time connection loss. Then, if a participant needs to connect to the server for n times, it has $1 - (1 - p_l)^n$ probability to lose the connection. In the experiment, we test $p_l = 0.01/0.05/0.1$ to represent good/moderate/bad connection scenarios.



(a) $p_l = 0.01$ (b) $p_l = 0.05$ (c) $p_l = 0.1$

Figure 5: Failure probability of trustworthiness ranking.



(a) Participants number (b) Connection loss

Figure 6: Detection accuracy of FedTruthFinder.

6.1.1 Event Confidence Computation

To compare with FedTruthFinder, we consider the state-of-the-art way to do iterative truth discovery with an SSS-based secure aggregation (SA) protocol [1, 34], denoted as *SA*, which can also tolerate a certain level of participant connection loss. In brief, SA leverages a double-masking method to ensure that the truth discovery can run when some users lose the connection. However, not like FedTruthFinder which only needs a one-time connection for each participant to finish one iteration of ρ-computation, SA needs a two-time connection (double-masking).

Figure 3 shows the number of sensed data for truth discovery in each iteration for FedTruthFinder and SA (the total number of data is set to 100). Literature has shown that the number of iterations for truth discovery is often smaller than 10 [36] and thus we set the number of iterations up to 10. FedTruthFinder can always obtain more sensed data than SA as FedTruthFinder needs fewer connections. Especially, when the network connection condition is bad ($p_l = 0.1$), the performance improvement of FedTruthFinder over SA is more significant.

Figure 4 shows the algorithm failure probability (i.e., fewer than t users are online) for FedTruthFinder and SA (we set the number of participants to 100 and t to 50; we do not plot $p_l = 0.01$ as both methods are successful almost all the time). FedTruthFinder can significantly reduce the failure probability compared to SA. For example, when the network connection quality is moderate ($p_l = 0.05$), FedTruthFinder has around 99% probability to finish successfully for 10 iterations; however, SA has only around 1% probability. For the bad connection scenario ($p_l = 0.1$), SA will fail with more than 20% probability from iteration 3, while FedTruthFinder can keep working well until iteration 6. This reveals that, even if both FedTruthFinder and SA cannot finish all the ten iterations due to a bad network connection condition, FedTruthFinder can run a larger number of iterations, making the truth more reliable.

6.1.2 Participant Trustworthiness Rank

As none of the prior studies have addressed the privacy-preserving trustworthiness ranking problem, we cannot directly find a baseline method to compare. Meanwhile, our proposed trustworthiness ranking algorithm is inspired by the basic idea from [22] while significantly enhancing the capability to tolerate participants' connection loss. To this end, we compare FedTruthFinder and [22] when certain participants lose connections.

In federated trustworthiness ranking, t is the key parameter related to how many user groups are created, which significantly impacts the algorithm success probability (Theorem 5.3). Suppose the total number of users is 100, we set $t = 10/15/20/25$. The algorithm failure probability is shown in Figure 5. With the increase of t, the failure probability of FedTruthFinder rises. This fits our expectation as a larger t means that more user groups are generated and the user number per group is reduced. As FedTruthFinder needs at least one user online for each group, smaller user number per group means that the robustness against connection loss is weakened, leading to higher failure probability. Compared to [22], our algorithm significantly increases the success probability when connection is unstable. When $p_l = 0.1$ and $t = 10$, the failure probability of our ranking algorithm is 0.04%, but [22] is 96.18%. Hence, our algorithm could be an appropriate choice for ranking crowdsensing participants' trustworthiness scores considering the unstable mobile network connection environment.

6.2 Evaluation of Traffic Light Detection

We also test FedTruthFinder for traffic light detection, a representative crowdsensing task [16, 25]. We focus on the truth discovery accuracy and the runtime efficiency of FedTruthFinder, which has not been evaluated in the previous numerical analysis.

6.2.1 Data and Tasks

To evaluate FedTruthFinder on the traffic light detection task, we leverage a real-life open dataset including taxis' trajectories. Specifically, the dataset contains time-stamped GPS trajectories from 536 taxis in San Francisco, U.S. in one month of 2008 [18]. Following [16], we manually label 96 traffic light detection event positions using the Street View of Google Maps (Figure 7). Then, we randomly select some taxis as participants; their trajectories in the dataset are used to simulate their activities — if a taxi stops around an event's location, it may report the data. The report error rate (indicating trustworthiness) of each taxi is randomized in $[0, 0.5]$. The default participant number is 60 and the connection loss probability is 0.05. The event confidence function is set to 'logistic' as it performs better than 'sum'. t in SSS is set to half of the total participant number. To increase the randomness, each taxi randomly reports 20% of the events, and then each setting of the experiment is repeated by 50 times.

6.2.2 Experiment Platform

Our platform is an Alibaba cloud server with CPU of Intel Xeon Platinum 8163 (12 cores, 2.5GHz) and 24GB memory. The operating system is Ubuntu 20.04. FedTruthFinder is implemented by *Rust* 1.56. *Docker*[5] is adopted to simulate the crowdsensing server and participants.

6.2.3 Truth Discovery Accuracy

Figure 6(a) and 6(b) plot the accuracy regarding the number of participants and connection loss probability, respectively. Specifically, we compare FedTruthFinder with and without the adaptive truth updating technique (Sec. 4.2.1). For the adaptive updating, we try $\alpha = 1/2/3$ (Eq. 91), and find $\alpha = 3$ performs the best. The adaptive updating ($\alpha = 3$) can consistently improve the accuracy with different participant numbers and connection losses. Specifically, with more participants and higher connection losses, the improvement is more significant. When the

[5]https://www.docker.com/

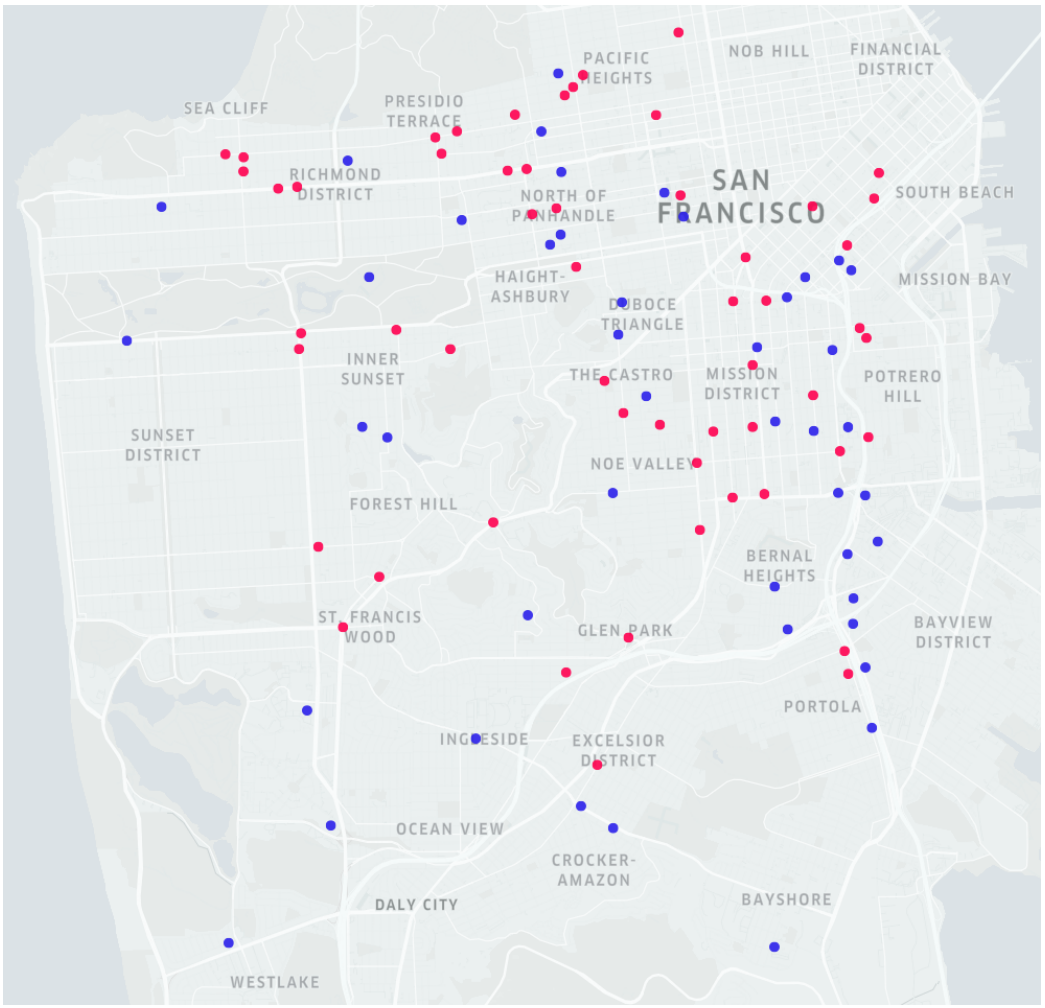


Figure 7: Traffic light event locations (red: true; blue: false). The red points represent the true event locations (i.e, with traffic lights) and the blue points mean the false event locations.

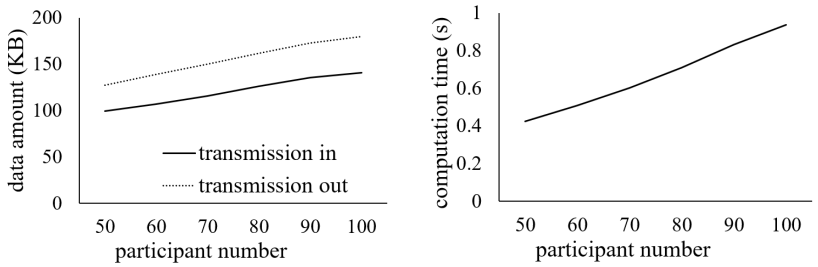
connection loss probability increases, the accuracy decreases gradually. This again verifies the effectiveness of FedTruthFinder over SA [34] — FedTruthFinder reduces the communication times per truth discover iteration compared to SA, which is conceptually equivalent to the reduction of connection losses in practice.

6.2.4 Runtime Efficiency

Figure 8(a) and 8(b) record each participant's data transmission amount and computation time, respectively. Note that the computation time is mostly spent in the truth finding step, while the trustworthiness ranking takes only $\sim$0.01s. The results show that the data transmission amount and computation time are both small, verifying the practicality of FedTruthFinder.

7 Related Work

Truth discovery is a traditional research direction as we may often receive diverse and even conflicting information about one event [7]. In the pioneering research [36], authors discuss the truth discovery problem when there



(a) Data transmission (b) Domputation time

Figure 8: Runtime efficiency of FedTruthFinder.

Table 15: Comparison of our work and representative related work. (NTP: No Third Party, AT: Assess Trustworthiness, CA: Collusion Attacks)

	Privacy Protection		**NTP**	**AT**	**Connection Loss**		**CA**
	Sensed Data	*Completed Tasks*			*Fault Tolerance*	*Bias Avoidance*	
[15]	√	×	×	×	×	×	×
[40]	√	×	×	×	√	×	×
[14]	√	×	√	×	√	×	×
[34]	√	×	√	×	√	×	√
[41]	√	×	√	×	√	×	√
[38]	√	×	×	×	√	×	×
Our Work	√	√	√	√	√	√	√

are many conflicting facts about one subject on different websites. Besides information from websites, truth discovery is also important in many other areas such as social sensing [26] and crowdsourcing [6, 33].

Mobile crowdsensing [39], as a particular type of crowdsourcing that needs workers to do location-based sensing tasks, would also face the truth discovery problem [24]. Meanwhile, privacy protection is also an important issue to consider in crowdsensing, especially for location privacy [4, 30, 27, 32, 29, 28]. Most prior research focuses on protecting crowdsensing participants' location privacy in task allocation [27, 32, 28] or for particular crowdsensing tasks such as missing data inference [30, 29].

Recently, some studies investigate the privacy-preserving truth discovery in crowdsensing [13, 15, 14, 38, 37, 34]. One research direction is applying data perturbation methods such as differential privacy to participants' sensed data [8, 9], but these methods degrade the truth finding accuracy. Another research direction follows the federated learning [35] paradigm that participants' raw data will not be directly sent to the server with certain encryption techniques, while the aggregation results (i.e., detected truths) can be accurately learned. However, the existing privacy-preserving truth discovery methods usually suffer from certain assumptions which may not stand in reality, e.g., online/non-concluding participants [13, 15, 14], or third-party non-concluding servers [38, 37]. Moreover, no prior work considers hiding participants' completed tasks or tracking participants' trustworthiness in a privacy-preserving manner, which has been addressed by our work.

Table 15 summarizes the characteristics of our work and representative related work published in top venues recently. In particular, our work is the **first** privacy-preserving crowdsensing truth discovery research that considers (i) *providing a feasible solution to participant trustworthiness assessment* when the trustworthiness scores are not revealed, and (ii) *hiding participants' completed tasks* to provide stronger privacy protection. Moreover, when dealing with the connection loss during the iterative crowdsensing truth discovery process, our work (i) proposes an adaptive event confidence updating function to reserve the data contributions of drop-out participants to avoid the truth bias toward alive participants, and (ii) designs an SSS-based scheme to defend

against participants' collusion attacks while ensuring the high communication efficiency.

8 Conclusion

In this paper, we propose *FedTruthFinder*, a crowdsensing federated truth discovery mechanism that can not only find aggregate truth from multiple participants' sensed data, but also rank participants' trustworthiness in a privacy-preserving manner. The primary characteristic of FedTruthFinder is its capability to tolerate network connection loss of participants in both event confidence calculation and participant trustworthiness ranking. As a byproduct, our proposed federated ranking algorithm can also serve other applications when the privacy-preserving data ranking is needed and the network connections are unstable. Following most related papers, this work assumes participants to be semi-honest; in the future, we would explore the more challenging scenario that participants may behave maliciously.

9 Appendix

9.1 Theoretical Proof

Proof of Lemma 5.1. It is clear that,

$$\sum_{k=1}^{2t+1} r_k(0)\tau_{i_k}(0) = \sum_{k=1}^{2t+1} r_k \tau_i^k \tag{97}$$

Besides, both $r_k(x)$ and $\tau_{i_k}(x)$ are t-degree polynomials, and thus the degree of $\sum_k r_k(x)\tau_{i_k}(x)$ is $2t$. □

Proof of Theorem 5.1. With Lemma 5.1, for N $(= 2t+1)$ groups, $\gamma(gid(u_j)) = \sum_{k=1}^{2t+1} r_k(gid(u_j))\tau_{i_k}(gid(u_j))$ (Step 4) is:

$$\begin{pmatrix} 1 & 1 & 1^2 & ... & 1^{2t} \\ 1 & 2 & 2^2 & ... & 2^{2t} \\ ... & ... & ... & ... & ... \\ 1 & N & N^2 & ... & N^{2t} \end{pmatrix} \begin{pmatrix} h_i \\ a_{i1} \\ ... \\ a_{i2t} \end{pmatrix} = \begin{pmatrix} \gamma(1) \\ \gamma(2) \\ ... \\ \gamma(N) \end{pmatrix}$$

then,

$$\begin{pmatrix} h_i \\ a_{i1} \\ ... \\ a_{i2t} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1^2 & ... & 1^{2t} \\ 1 & 2 & 2^2 & ... & 2^{2t} \\ ... & ... & ... & ... & ... \\ 1 & N & N^2 & ... & N^{2t} \end{pmatrix}^{-1} \begin{pmatrix} \gamma(1) \\ \gamma(2) \\ ... \\ \gamma(N) \end{pmatrix}$$

so,

$$h_i = \sum_{g=1}^{N} \lambda(g)\gamma(g)$$

In Step 5, $h_i(g) = \lambda(g)\gamma(g)$ is shared with $(t+1, n)$-SSS to all the participants from every group $g \in [1, 2t+1]$. Hence, according to the additive homomorphism property of SSS [21], we can easily recover h_i by receiving $t+1$ participants' $h_i'(k) = \sum_{g=1}^{2t+1} h_i(g, k)$. □

Proof of Theorem 5.2. As $\tau_i > 0$ and $r_k > 0$, $h_i = \sum_k r_k \tau_i^k$ will keep the same ranking as τ_i. □

Proof of Theorem 5.3. For Step 3 to 5, if there is at least one user in every group, then the process can continue. So the probability of failure incurred by one specific group g is all the users in g losing the connections,

i.e., $p^{n_g} \leq p_l^{\lfloor n/(2t+1) \rfloor}$ (n_g is the user number in g). So for g, the probability of at least one user online $\geq 1 - p_l^{\lfloor n/(2t+1) \rfloor}$. With $2t+1$ groups, the success probability $\geq (1 - p_l^{\lfloor n/(2t+1) \rfloor})^{2t+1}$. □

Proof of Theorem 5.4. This is based on the property of $(t+1, n)$-SSS in Step 5. □

Proof of Theorem 5.5. In Step 2, $\tau_i^k (k = 1...2t+1)$ is shared with $(t+1, 2t+1)$-SSS. So, if t participants collude, they can get at most $t \cdot (2t+1)$ equations when t participants are from t groups. However, the number of unknown parameters (including τ_i and t random coefficients for sharing each τ_i^k) is $t \cdot (2t+1) + 1$. Hence, these t collusive participants cannot recover other participants' τ_i. □

9.2 Mechanism Extension to Multi-class and Continuous-value Events

Multi-class Events. For a multi-class event (m classes), we can see it as m binary events, so that our method can be directly applied.

Continuous-value Events. For continuous-value events, following the literature, we may adopt other proper event confidence and participant trustworthiness updating functions such as CRH [34, 41]. Specifically, suppose that the discovered truth sensed value of a continuous event e_j is ρ_j, and u_i's sensed data of e_j is $\hat{\rho}_{ij}$, then the event truth (confidence) and participant trustworthiness updating functions can be:

$$\rho_j = \frac{\sum_{u_i \in \mathcal{U}_{e_j}} \tau_i \cdot \hat{\rho}_{ij}}{\sum_{u_i \in \mathcal{U}_{e_j}} \tau_i} \tag{98}$$

$$\tau_i = \log(\sum_{u_i \in \mathcal{U}} \sum_{e_j \in \mathcal{E}_{u_i}} \frac{(\rho_j - \hat{\rho}_{ij})^2}{|\mathcal{E}_{u_i}|}) - \log(\sum_{e_j \in \mathcal{E}_{u_i}} \frac{(\rho_j - \hat{\rho}_{ij})^2}{|\mathcal{E}_{u_i}|}) \tag{99}$$

where $\mathcal{U}_{e_j}$ is the set of users who sense e_j, and $\mathcal{E}_{u_i}$ is the set of events that u_i has sensed. For ρ-computation, following Sec. 4.1, we can just adapt d_{ij} and s_{ij} according to Eq. 98 (the participant $u_i \notin \mathcal{U}_{e_j}$ can still send $d_{ij} = s_{ij} = 0$ to protect her task completion information). For τ-computation, Eq. 99 requires $\sum_{u_i \in \mathcal{U}} \sum_{e_j \in \mathcal{E}_{u_i}} \frac{(\rho_j - \hat{\rho}_{ij})^2}{|\mathcal{E}_{u_i}|}$, which can be done with the same SSS-based method as ρ-computation. In particular, each participant u_i can send $\sum_{e_j \in \mathcal{E}_{u_i}} \frac{(\rho_j - \hat{\rho}_{ij})^2}{|\mathcal{E}_{u_i}|}$ by secret shares, and then the server can compute the sum in a privacy-preserving manner. In a word, for continuous-value events, our mechanism can still work without revealing each participant's raw sensed data and completed tasks.

References

[1] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. CCS, 2017.

[2] D. Chai, L. Wang, K. Chen, and Q. Yang. Secure federated matrix factorization. IEEE Intelligent Systems, 2020.

[3] R. K. Ganti, F. Ye, and H. Lei. Mobile crowdsensing: current state and future challenges. IEEE Communications Magazine, 49(11):32–39, 2011.

[4] X. Han, L. Wang, and W. Fan. Is hidden safe? location protection against machine-learning prediction attacks in social networks. MIS Quarterly, 45(2):821–858, 2021.

[5] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. arXiv:1610.05492, 2016.

[6] H. Li, B. Zhao, and A. Fuxman. The wisdom of minority: discovering and targeting the right group of workers for crowdsourcing. In WWW, pages 165–176, 2014.

[7] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han. A survey on truth discovery. SIGKDD Explor. Newsl., 17(2):1–16, Feb. 2016.

[8] Y. Li, C. Miao, L. Su, J. Gao, Q. Li, B. Ding, Z. Qin, and K. Ren. An efficient two-layer mechanism for privacy-preserving truth discovery. KDD, 2018.

[9] Y. Li, H. Xiao, Z. Qin, C. Miao, L. Su, J. Gao, K. Ren, and B. Ding. Towards differentially private truth discovery for crowd sensing systems. ICDCS, pages 1156–1166, 2020.

[10] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang. A secure federated transfer learning framework. IEEE Intelligent Systems, 35(4):70–82, 2020.

[11] H. Ma, D. Zhao, and P. Yuan. Opportunities in mobile crowd sensing. IEEE Communications Magazine, 52(8):29–35, 2014.

[12] C. Meng, W. Jiang, Y. Li, J. Gao, L. Su, H. Ding, and Y. Cheng. Truth discovery on crowd sensing of correlated entities. In Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, pages 169–182, 2015.

[13] C. Miao, W. Jiang, L. Su, Y. Li, S. Guo, Z. Qin, H. Xiao, J. Gao, and K. Ren. Cloud-enabled privacy-preserving truth discovery in crowd sensing systems. In SenSys, 2015.

[14] C. Miao, W. Jiang, L. Su, Y. Li, S. Guo, Z. Qin, H. Xiao, J. Gao, and K. Ren. Privacy-preserving truth discovery in crowd sensing systems. ACM Transactions on Sensor Networks, 15:1 – 32, 2019.

[15] C. Miao, L. Su, W. Jiang, Y. Li, and M. Tian. A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems. INFOCOM, pages 1–9, 2017.

[16] R. W. Ouyang, M. Srivastava, A. Toniolo, and T. J. Norman. Truth discovery in crowdsourced detection of spatial events. IEEE transactions on knowledge and data engineering, 28(4):1047–1060, 2015.

[17] D. Peng, F. Wu, and G. Chen. Data quality guided incentive mechanism design for crowdsensing. IEEE Transactions on Mobile Computing, 17:307–319, 2018.

[18] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from `https://crawdad.org/epfl/mobility/20090224`, Feb. 2009.

[19] V. Primault, A. Boutet, S. B. Mokhtar, and L. Brunie. The long road to computational location privacy: A survey. IEEE Communications Surveys & Tutorials, 21:2772–2793, 2019.

[20] S. Reddy, D. Estrin, M. Hansen, and M. Srivastava. Examining micro-payments for participatory sensing data collections. UbiComp, 2010.

[21] A. Shamir. How to share a secret. Communications of the ACM, 22(11):612–613, 1979.

[22] C. Tang, G. Shi, and Z. Yao. Secure multi-party computation protocol for sequencing problem. SCIENTIA SINICA Informationis, 41(7):789–797, 2011.

[23] X. Tang, C. Wang, X. Yuan, and Q. Wang. Non-interactive privacy-preserving truth discovery in crowd sensing applications. INFOCOM, pages 1988–1996, 2018.

[24] D. Wang, T. Abdelzaher, and L. M. Kaplan. Surrogate mobile sensing. IEEE Communications Magazine, 52:36–41, 2014.

[25] D. Wang, L. Kaplan, T. Abdelzaher, and C. C. Aggarwal. On credibility estimation tradeoffs in assured social sensing. IEEE Journal on Selected Areas in Communications, 31(6):1026–1037, 2013.

[26] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. On truth discovery in social sensing: A maximum likelihood estimation approach. In IPSN, pages 233–244, 2012.

[27] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma. Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation. WWW, 2017.

[28] L. Wang, D. Yang, X. Han, D. Zhang, and X. Ma. Mobile crowdsourcing task allocation with differential-and-distortion geo-obfuscation. IEEE Transactions on Dependable and Secure Computing, 18:967–981, 2021.

[29] L. Wang, D. Zhang, D. Yang, B. Y. Lim, X. Han, and X. Ma. Sparse mobile crowdsensing with differential and distortion location privacy. IEEE Transactions on Information Forensics and Security, 15:2735–2749, 2020.

[30] L. Wang, D. Zhang, D. Yang, B. Y. Lim, and X. Ma. Differential location privacy for sparse mobile crowdsensing. In ICDM, pages 1257–1262. IEEE, 2016.

[31] T. Wang, C. Lv, C. Wang, F. Chen, and Y. Luo. A secure truth discovery for data aggregation in mobile crowd sensing. Secur. Commun. Networks, 2021:2296386:1–2296386:15, 2021.

[32] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, and H. Qi. Personalized privacy-preserving task allocation for mobile crowdsensing. IEEE Transactions on Mobile Computing, 18:1330–1341, 2019.

[33] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In NeurIPS, 2009.

[34] G. Xu, H. Li, S. Liu, M. Wen, and R. Lu. Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. IEEE Transactions on Vehicular Technology, 68:3854–3865, 2019.

[35] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology, 10(2):12, 2019.

[36] X. Yin, J. Han, and S. Y. Philip. Truth discovery with multiple conflicting information providers on the web. IEEE Transactions on Knowledge and Data Engineering, 20(6):796–808, 2008.

[37] C. Zhang, C. Xu, L. Zhu, Y. Li, C. Zhang, and H. Wu. An efficient and privacy-preserving truth discovery scheme in crowdsensing applications. Computers & Security, 97:101848, 2020.

[38] C. Zhang, L. Zhu, C. Xu, X. Liu, and K. Sharif. Reliable and privacy-preserving truth discovery for mobile crowdsensing systems. IEEE Transactions on Dependable and Secure Computing, 18:1245–1260, 2021.

[39] D. Zhang, L. Wang, H. Xiong, and B. Guo. 4w1h in mobile crowd sensing. IEEE Communications Magazine, 52(8):42–48, 2014.

[40] Y. Zheng, H. Duan, and C. Wang. Learning the truth privately and confidently: Encrypted confidence-aware truth discovery in mobile crowdsensing. IEEE Transactions on Information Forensics and Security, 13:2475–2489, 2018.

[41] Y. Zheng, H. Duan, X. Yuan, and C. Wang. Privacy-aware and efficient mobile crowdsensing with truth discovery. IEEE Transactions on Dependable and Secure Computing, 17:121–133, 2020.