Data Poisoning Attacks and Defenses to Crowdsourcing Systems

Minghong Fang[1,2], Minghao Sun[1], Qi Li[1], Neil Zhenqiang Gong[3], Jin Tian[1], Jia Liu[2]
[1]Iowa State University [2]The Ohio State University [3]Duke University

ABSTRACT

A key challenge of big data analytics is how to collect a large volume of (labeled) data. Crowdsourcing aims to address this challenge via aggregating and estimating high-quality data (e.g., sentiment label for text) from pervasive clients/users. Existing studies on crowdsourcing focus on designing new methods to improve the aggregated data quality from unreliable/noisy clients. However, the security aspects of such crowdsourcing systems remain underexplored to date. We aim to bridge this gap in this work. Specifically, we show that crowdsourcing is vulnerable to *data poisoning attacks*, in which malicious clients provide carefully crafted data to corrupt the aggregated data. We formulate our proposed data poisoning attacks as an optimization problem that maximizes the error of the aggregated data. Our evaluation results on one synthetic and two real-world benchmark datasets demonstrate that the proposed attacks can substantially increase the estimation errors of the aggregated data. We also propose two defenses to reduce the impact of malicious clients. Our empirical results show that the proposed defenses can substantially reduce the estimation errors of the data poisoning attacks.

CCS CONCEPTS

• **Security and privacy** → **Systems security**.

KEYWORDS

Data poisoning attacks, crowdsourcing, truth discovery

ACM Reference Format:
Minghong Fang[1,2], Minghao Sun[1], Qi Li[1], Neil Zhenqiang Gong[3], Jin Tian[1], Jia Liu[2]. 2021. Data Poisoning Attacks and Defenses to Crowdsourcing Systems . In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3450066

1 INTRODUCTION

Background and Motivation: A well-known challenge for big data analytics is that its success highly relies on a large amount of (labeled) data. Crowdsourcing aims to address this challenge by significantly reducing the labeling cost. However, since individuals may make mistakes, a common practice is to hire multiple workers for the same task and then obtain high-quality aggregated data. Specifically, a data requester (called *central server* in this paper) has a set of items/tasks. The server distributes the items to some *workers*; each worker provides a value for each item that is allocated to him/her; and the server estimates an *aggregated value* for each item using the workers' values. For instance, the items could be some text documents and the server aims to estimate a numeric value (e.g., a number between -100 and 100) for each document.

Although crowdsourcing holds a great potential to solve the labeling challenges in big data analytics, a critical challenge that affecting its future large-scale adoption stems from the fact that the data provided by workers in many crowdsourcing applications are *noisy* and *unreliable*. To extract high-quality information from unreliable and noisy values provided by workers in crowdsourcing, a widely adopted approach is the so-called truth discovery methods [10, 12, 18, 19, 25–29, 34, 35, 40, 41, 48–50]. Generally speaking, the truth discovery methods cover a family of algorithms that perform *weighted* aggregation of worker information based on the quality of each worker. Specifically, in many real-world crowdsourcing applications, some workers may provide biased or incorrect values for items due to various reasons, e.g., lack of effort, lack of expertise, etc. To handle unreliable and noisy worker information, state-of-the-art truth discovery methods (e.g., the conflict resolution on heterogeneous data (CRH) [26, 29], Gaussian truth model (GTM) [49], etc.) jointly estimate workers' reliability measured by certain uncertainty metrics (e.g., variance, etc.) and use these reliability metrics as weights in aggregating the values provided by the workers for each item. The rationale behind the truth discovery methods is simple: if a worker does not have a large deviation from the majority of the workers very often, then this worker is more likely to be reliable. Further, if a piece of information is provided by reliable workers, then this information is more likely to be correct and should be assigned a larger weight in the aggregation.

It is worth noting that, to date, most algorithms in the truth discovery family are based on the assumption that all workers are benign and the unreliable values from the workers are caused by unavoidable randomness in the nature. Unfortunately, in the presence of *malicious workers* who could provide *carefully crafted values* to the server (aka *data poisoning attacks*), recent studies have found that existing truth discovery methods could perform rather poorly (see, e.g., [31, 32]). We note that, although these results exposed the vulnerability of truth discovery methods, they remain mostly limited to crowdsourcing applications with categorical labels (i.e., discrete labels in multiple-choice surveys, etc.). So far, research results on data poisoning attacks and defense for crowdsourcing with *continuous* labels are still quite limited. However, crowdsourcing applications with continuous labeling are prevalent in practice (for example, the temperature values in the crowdsourcing-based weather reporting are continuous). In light of the growing importance of crowdsourcing applications, there is a compelling need to investigate and understand the data poisoning attacks and defense for continuous-labeled crowdsourcing systems.

Our work: In this paper, we aim to bridge this gap and show that truth discovery methods are vulnerable to data poisoning attacks, in which an attacker injects malicious workers to a crowdsourcing system and the malicious workers provide carefully crafted values

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450066

to corrupt the truth discovery system. In particular, our data poisoning attacks can increase the estimation errors of the aggregated values substantially.

Toward this end, we formulate our data poisoning attacks for truth discovery methods as an optimization problem, whose objective function is to maximize the estimation errors of the aggregated values for the attacker-chosen targeted items and whose variables are the values provided by the malicious workers. In particular, our optimization problem is bi-level, which is NP-hard and challenging to solve exactly. We address the challenge via iteratively solving the upper-level and lower-level subproblems in our bi-level optimization problem via a projected gradient ascent method.

We evaluate our data poisoning attacks using one synthetic dataset and two well-known benchmark datasets in the crowdsourcing community. For instance, in one benchmark dataset called Emotion, the central server aims to estimate the sentiment values (ranging from -100 to 100) for 700 documents from 38 workers, where each document is allocated to 10 workers. To show the effectiveness of our data poisoning attacks on the truth discovery algorithms, we use two state-of-the-art methods from this family called *Conflict Resolution on Heterogeneous Data (CRH)* and *Gaussian Truth Model (GTM)* as examples. We show that our attacks can substantially increase the average estimation error. For instance, on the sentiment estimation dataset, our attack can increase the average estimation error of the sentiment values to 93.69 when 10% of workers are malicious under the CRH model.

We also propose two defense mechanisms to mitigate our data poisoning attacks, namely *Median-of-Weighted-Average (MWA)* and *Maximize Influence of Estimation (MIE)*. In the MWA defense, the server first partitions the workers who provide values for a given item into groups, computes the weighted average in each group, and then estimates the median of the weighted average among the groups as the final aggregated value for the item. Note that in MWA, we considers all workers to estimate the aggregated values, though the impact of the malicious workers is mitigated by robust aggregation. By contrast, in MIE, we use an influence function to identify the potential malicious workers and remove them before estimating the aggregated values. Our empirical results show that our defenses can substantially reduce the estimation errors of our data poisoning attacks.

Our contributions in this paper are summarized as follows:

- We propose data poisoning attacks to crowdsourcing, which can be formulated as a bi-level optimization problem. Due to the NP-hardness of the problem, we propose an efficient algorithm that achieves competitive results.
- We evaluate our attacks on three datasets and show that our attacks can increase the estimation errors substantially.
- We propose two defenses to mitigate our attacks. Our experimental results demonstrate that the proposed defenses can effectively reduce the estimation errors.

2 PRELIMINARIES AND RELATED WORK

In this section, we first provide an overview and some necessary preliminaries of crowdsourcing and truth discovery methods, using two state-of-the-art truth discovery methods called *Conflict Resolution on Heterogeneous Data (CRH)* [26, 29] and *Gaussian Truth Model (GTM)* [49] as examples. Then, we provide an overview on data poisoning attacks, which put our work in comparative perspectives. Table 1 lists the key notation used in this paper.

Table 1: Summary of key notation.

Notation	Definition
$\mathcal{I}/\lvert\mathcal{I}\rvert$	Set/number of items
$\mathcal{U}/\widetilde{\mathcal{U}}$	Set of normal/malicious workers
$\mathcal{M}$	Set of all workers, $\mathcal{M} = \mathcal{U} \cup \widetilde{\mathcal{U}}$
$x_i^u/x_i^{\tilde{u}}$	Value of normal/malicious worker $u/\tilde{u}$ on item i
$x_i^\star/\widehat{x}_i^*$	Aggregated value for item i before/after attack
$w_u/w_{\tilde{u}}$	Weight of normal/malicious worker $u/\tilde{u}$
$\sigma_u^2/\sigma_{\tilde{u}}^2$	Variance of normal/malicious worker $u/\tilde{u}$
$\mathcal{U}_i/\widetilde{\mathcal{U}}_i$	Set of normal/malicious workers who provide values for item i
$\mathcal{I}_u/\mathcal{I}_{\tilde{u}}$	Set of items rated by normal/malicious worker $u/\tilde{u}$

2.1 The Truth Discovery Methods: A Primer

In this subsection, we provide an overview on the family of truth discovery methods for crowdsourcing. In most crowdsourcing systems, there is a *central server* performing data aggregation and there are some clients called *workers*. We denote the set of workers as $\mathcal{U}$. The server has a set of items $\mathcal{I}$ and aims to estimate a certain *value* for each item based on the input from the workers. In this work, we focus on the cases where the value to be estimated is *continuous*. For instance, the items could be a set of text documents and the server aims to estimate a numeric value for each document from the workers. The server assigns each item to a subset of workers. We denote by $\mathcal{I}_u$ the set of items that are allocated to worker u. Moreover, we denote by x_i^u the value that the worker u provides for item i, where $u \in \mathcal{U}$ and $i \in \mathcal{I}_u$.

To find reliable information among unreliable data, a naive approach is majority voting or taking the average of the values provided by workers. A major limitation of these methods is that they do not take the quality/reliability of workers into consideration. In practice, the quality of different workers varies. To address this challenge, the truth discovery approaches are proposed to automatically jointly estimate the quality of workers while performing information aggregation. The rationale behind the truth discover methods is to characterize the reliability of a worker as a weight. If a worker has a smaller weight, then all of its provided values are less reliable. To illustrate the basic idea of the truth discovery methods, in what follows, we use two state-of-the-art algorithms in this family that are widely adopted in crowdsourcing systems as concrete examples: i) conflict resolution on heterogeneous data (CRH) [26, 29] and ii) Gaussian truth model (GTM) [49].

2.1.1 The Conflict Resolution on Heterogeneous Data Model (CRH). CRH, a state-of-the-art truth discovery method, jointly estimates the aggregated values of items and the weights of workers. In particular, CRH formulates the estimations of the aggregated values and worker weights as the following optimization problem:

$$\begin{aligned} \min_{X^\star, W} f(X^\star, W) &= \sum\nolimits_{u\in\mathcal{U}} w_u \sum\nolimits_{i\in\mathcal{I}_u} d(x_i^u, x_i^\star) \\ \text{s.t.} \quad & \sum\nolimits_{u\in\mathcal{U}} \exp(-w_u) = 1, \end{aligned} \tag{1}$$

where $X^\star = \{x_i^\star\}_{i\in\mathcal{I}}$ is the set of aggregated values for all the items, $W = \{w_u\}_{u\in\mathcal{U}}$ is the set of weights for all the workers, w_u

Algorithm 1 The CRH framework [26, 29].

Input: Values from workers x_i^u for $u \in \mathcal{U}, i \in \mathcal{I}_u$.
Output: Aggregated values $X^\star$ and worker weights W.
1: Server initializes the worker weights.
2: **while** the convergence condition is not satisfied **do**
3: Server updates the aggregated value of each item according to Eq. (2).
4: Server updates the weight of each worker according to Eq. (3).
5: **end while**
6: **return** $X^\star$ and W.

is weight for worker u, $d(\cdot)$ is a function to measure the distance between a worker's value of an item and the item's aggregated value, which reflects the reliability of this particular worker. In our experiments, we use the square distance function. CRH solves the optimization problem by iteratively alternating between the following two steps:

Step 1 (Estimate the aggregated values): In this step, the workers' weights W are fixed, and the aggregated value for item i is updated as follows:

$$x_i^\star = \frac{\sum_{u \in \mathcal{U}_i} w_u x_i^u}{\sum_{u \in \mathcal{U}_i} w_u}, \tag{2}$$

where $\mathcal{U}_i$ is the set of workers who provide values for item i.

Step 2 (Update worker weights): Next, the aggregated values $X^\star$ are fixed, and the weight of worker u is updated as follows:

$$w_u = \log\left(\frac{\sum_{k \in \mathcal{U}} \sum_{i \in \mathcal{I}_k} d(x_i^k, x_i^\star)}{\sum_{i \in \mathcal{I}_u} d(x_i^u, x_i^\star)}\right). \tag{3}$$

It can be seen from (3) that, the smaller the distance $d(x_i^u, x_i^\star)$, the larger the weight of worker u (i.e., more reliable). We can use the block coordinate descent method [3] to iteratively update the above two-step procedure until some convergence criterion is met. Algorithm 1 shows the CRH framework. In this paper, we assume that all workers are given equal initial weights in the CRH method.

2.1.2 The Gaussian Truth Model (GTM). GTM model is a Bayesian probabilistic model designed for numeric data in truth discovery. In GTM, the reliability of a worker are captured by a variance parameter. Intuitively, a worker with larger variance is more likely to provide inaccurate values that deviate more from the truth. The GTM model first normalizes all input values to its z-scores, then tries to solve the following optimization problem:

$$\min_{X^\star, \Omega} f(X^\star, \Omega) \propto -\sum_{u \in \mathcal{U}} \left(2(\alpha+1)\log\sigma_u + \frac{\beta}{\sigma_u^2}\right) - \sum_{i \in \mathcal{I}} \frac{(x_i^\star - \mu_0)^2}{2\sigma_0^2} - \sum_{i \in \mathcal{I}} \sum_{u \in \mathcal{U}_i} \left(\log\sigma_u + \frac{(x_i^u - x_i^\star)^2}{2\sigma_u^2}\right), \tag{4}$$

where $\Omega = \{\sigma_u^2\}_{u \in \mathcal{U}}$ is the set of variances for all the workers, σ_u^2 is the variance of worker u, μ_0 and σ_0^2 are prior parameters and α and β are hyper-parameters. The GTM model leverages the EM algorithm [11] which contains the following expectation step (E-step) and maximization step (M-step) to iteratively update aggregated values and variance parameters of workers':

E-step (Estimate the aggregated values): In this step, the workers' variances are fixed, and the aggregated value for item i is computed by solving $\frac{\partial f}{x_i^\star} = 0$, which yields:

$$x_i^\star = \frac{\frac{\mu_0}{\sigma_0^2} + \sum_{u \in \mathcal{U}_i} \frac{x_i^u}{\sigma_u^2}}{\frac{1}{\sigma_0^2} + \sum_{u \in \mathcal{U}_i} \frac{1}{\sigma_u^2}}. \tag{5}$$

M-step (Update worker variances): In this step, the aggregated values $X^\star$ are fixed, and the variance of worker u can be calculated by solving $\frac{\partial f}{\sigma_u^2} = 0$, which yields:

$$\sigma_u^2 = \frac{2\beta + \sum_{i \in \mathcal{I}_u} (x_i^u - x_i^\star)^2}{2(\alpha+1) + |\mathcal{I}_u|}, \tag{6}$$

where $|\mathcal{I}_u|$ is the number of values provided by worker u. The EM algorithm alternates between the above two steps iteratively until some convergence criterion is satisfied. The GTM algorithmic framework is similar to Algorithm 1, and we omit it here for briefly.

2.2 Data Poisoning Attacks: An Overview

Generally speaking, data poisoning attacks refer to manipulating data to corrupt certain computational results based on those data. For instance, in machine learning, a classifier is learnt using a training dataset; and a data poisoning attack can carefully forge the training dataset to corrupt the learnt classifier [1, 5, 7, 22, 36, 44]. In data poisoning attacks to recommender systems [9, 16, 17, 24, 47], an attacker can inject fake users with carefully crafted rating values to a recommender system such that the recommender system makes recommendations as the attacker desires, e.g., recommending an attacker-chosen item to many normal users. In federated learning, an attacker can inject malicious workers with misleading training samples to corrupt the learnt global model [2, 4, 6, 15, 46]. Different computations (e.g., machine learning, recommender system, and federated learning) often require different data poisoning attacks to optimize the attack effectiveness. The most relevant to ours are [31, 32, 38], where the authors proposed efficient attack algorithms that reduce the effectiveness of crowdsourcing systems with strategic malicious workers. However, the proposed attack models focus on categorical data and do not consider the potential defense deployed by the server.

3 PROBLEM FORMULATION

In this section, we first introduce our threat model, including the attacker's goal, capability, and knowledge. Then, we formulate the attacker's goal mathematically.

3.1 Threat Model

Attacker's goal: Given a targeted crowdsourcing system, the goal of the attacker is to maximize the estimation errors of the aggregated values for some attacker-chosen targeted items. This attack is well motivated in real-world crowdsourcing systems. For instance, an attacker may be interested in manipulating the real-time navigation crowdsourcing system such that the system provides misleading or even life-threatening directions to users. An attacker could also attack a competitor's system to gain competitive advantages.

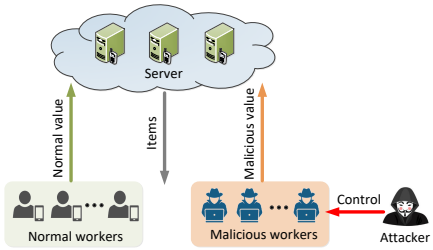

Figure 1: Crowdsourcing systems under attack.

Attacker's capability and knowledge: In our threat model, we assume the attacker is able to inject some malicious workers into the crowdsourcing system and launch attacks by carefully crafting their values, as shown in Figure 1. This threat model is practical because a crowdsourcing system is essentially a distributed system and an attacker can inject malicious workers into it, which is more realistic than modifying existing data of normal workers.

The attacker can have different degrees of knowledge of the targeted crowdsourcing system. In particular, we consider two cases, *full knowledge* and *partial knowledge*. In the full knowledge scenario, the attacker has full knowledge of the aggregation method and all values provided by normal workers. We note that, although appearing to be a strong assumption, the full-knowledge setting is *not* uncommon in practice since all data and the sources of the data could be public in crowdsourcing. For instance, in the weather forecast integration task, the workers in different monitoring stations collect the local weather data and upload their data to the crowdsourcing platform (e.g., website). These data are available to all workers and each worker can see weather information at other locations and know where the data comes from.

In the partial knowledge scenario, the attacker knows the aggregation method but only knows the values of a subset of normal workers. For instance, the attacker may compromise some normal workers via bribing them, compromising their computer systems, and/or stealing their credentials.

3.2 Formulating Data Poisoning Attacks

We formulate our data poisoning attacks as an optimization problem, which maximizes the estimation errors of the targeted items' aggregated values. Suppose that $\mathcal{T}$ is the set of attacker-chosen targeted items and α fraction of workers are malicious. Specifically, we let $\mathcal{U}$ and $\widetilde{\mathcal{U}}$ denote the sets of normal and malicious workers, respectively. Then, we have $|\widetilde{\mathcal{U}}| = \left\lfloor \frac{\alpha|\mathcal{U}|}{1-\alpha} \right\rfloor$. We let $x_t^{\tilde{u}}$ denote the value that a malicious worker $\tilde{u} \in \widetilde{\mathcal{U}}$ provides on item $t \in \mathcal{T}$. Then, the attacker's goal is to find an optimal value for each malicious worker to rate each targeted item, such that after injecting those malicious workers into the crowdsourcing system, the distance of the aggregated values before and after our attack is maximized. We let $x_t^\star$ and $\widehat{x}_t^*$ denote the aggregated values for item t before and after the poisoning attack, respectively. Then, we can formulate our attack as follows:

$$\underset{\{x_t^{\tilde{u}}\}_{t\in\mathcal{T},\tilde{u}\in\widetilde{\mathcal{U}}}}{\text{Maximize}} \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} d(\widehat{x}_t^*, x_t^\star) \tag{7}$$

$$\text{s.t. } |\widetilde{\mathcal{U}}| = \left\lfloor \frac{\alpha|\mathcal{U}|}{1-\alpha} \right\rfloor, \tag{8}$$

where $d(\cdot)$ is a distance function that measures the estimation error of an item introduced by our attack. In our experiments, we use the square distance function. The objective function $\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} d(\widehat{x}_t^*, x_t^\star)$ measures the average estimation error for the targeted items introduced by the attack. Note that our problem formulation in Eq. (7) can be applied to any truth discovery method. Solving the optimization problem for a given truth discovery method (e.g., CRH, GTM, etc.) leads to a specific data poisoning attack to this particular method.

4 OUR ATTACKS

In this section, we introduce our data poisoning attacks in the full-knowledge and partial-knowledge scenarios, respectively. Note that our attacks are essentially to solve the optimization problem in Eq. (7) under these two settings.

4.1 Full-Knowledge Attack

In the full-knowledge scenario, the attacker knows the values provided by the normal workers. Note that the attacker's goal in Eq. (7) is to maximize the deviation of the truth discovery's output before and after the attack, while the server's goal of Eq. (1) or Eq. (4) is to estimate the aggregated value for each item. Moreover, the malicious workers' values should be within the normal range to avoid outlier detection. Considering these two goals together, we can instantiate the optimization problem in Eq. (7) for the CRH and GTM models as the following bi-level optimization problem for the attacker:

$$\begin{aligned} &\underset{\{x_t^{\tilde{u}}\}_{t\in\mathcal{T},\tilde{u}\in\widetilde{\mathcal{U}}}}{\text{Maximize}} \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} d(\widehat{x}_t^*, x_t^\star) \\ &\text{s.t. } x_t^{\tilde{u}} \in \left[x_t^{\min}, x_t^{\max}\right], \forall \tilde{u} \in \widetilde{\mathcal{U}}, \forall t \in \mathcal{T}, \\ &\{\widehat{X}^*, \Lambda^*\} = \underset{\widehat{X}^\star,\Lambda}{\arg\min} f(\widehat{X}^\star, \Lambda), \end{aligned} \tag{9}$$

where $x_t^{\min}$ and $x_t^{\max}$ are the minimum and maximum values of item t provided by normal workers, respectively; Λ^* are the after-attack weight or variance parameters for all workers. We remind here that Problem (9) is a general attack framework, and can be applied to any truth discovery method. To be specific, for the CRH model, we just substitute $f(\widehat{X}^\star, W)$ in Eq. (1) to $f(\widehat{X}^\star, \Lambda)$ in Eq. (9) and let $W = \Lambda$; for the GTM model, we substitute $f(\widehat{X}^\star, \Omega)$ in Eq. (4) to $f(\widehat{X}^\star, \Lambda)$ in Eq. (9) and let $\Omega = \Lambda$. From Eq. (9), we can see that the upper-level problem is to determine the optimal fake values for malicious workers, and the lower-level problem is to estimate the aggregated value for each item. The lower-level problem can be considered a constraint of the upper-level problem. Bi-level optimization is NP-hard in general [20]. In our bi-level formulation, although the upper-level problem is relatively simple, the lower-level problem is highly non-linear and non-convex. In this paper, we propose

the following two-step iterative method to solve the above bi-level optimization problem:

Step 1 (Update the aggregated values and worker weights/variances): In this step, the attacker fixes the malicious workers' values, then solves the lower-level optimization problem to obtain aggregated values and worker weights/variances $\{\widehat{X}^*, \Lambda^*\}$. As we have discussed above, this step can be done for the CRH model by solving Eqs. (2)-(3) or for the GTM model by solving Eqs. (5)-(6).

Step 2 (Update malicious workers' values): In this step, we let $\widetilde{X} = \{x_t^{\tilde{u}}\}_{t\in\mathcal{T},\tilde{u}\in\widetilde{\mathcal{U}}}$ denote the values of all malicious workers, and define the following objective function:

$$\mathcal{L}(\widetilde{X}) = \sum\nolimits_{t\in\mathcal{T}} d(\widehat{x}_t^*, x_t^\star). \tag{10}$$

When taking the malicious workers into consideration, we can compute the term $\widehat{x}_t^*$ in Eq. (10) as followings for the CRH model:

$$\widehat{x}_i^* = \frac{\sum_{u\in\mathcal{U}_i} w_u x_i^u + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_i} w_{\tilde{u}} x_i^{\tilde{u}}}{\sum_{u\in\mathcal{U}_i} w_u + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_i} w_{\tilde{u}}}. \tag{11}$$

For the GTM model, $\widehat{x}_t^*$ can be computed as:

$$\widehat{x}_i^* = \frac{\frac{\mu_0}{\sigma_0^2} + \sum_{u\in\mathcal{U}_i} \frac{x_i^u}{\sigma_u^2} + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_i} \frac{x_i^{\tilde{u}}}{\sigma_{\tilde{u}}^2}}{\frac{1}{\sigma_0^2} + \sum_{u\in\mathcal{U}_i} \frac{1}{\sigma_u^2} + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_i} \frac{1}{\sigma_{\tilde{u}}^2}}. \tag{12}$$

For malicious worker $v \in \widetilde{U}$, the gradient of the objective function $\mathcal{L}(\widetilde{X})$ with respect to x_t^v can be computed as follows:

$$\nabla_{x_t^v}\mathcal{L}(\widetilde{X}) = \sum_{t'\in\mathcal{T}} \frac{\partial d(\widehat{x}_{t'}^*, x_{t'}^\star)}{\partial x_t^v} = \sum_{t'\in\mathcal{T}} \frac{\partial d(\widehat{x}_{t'}^*, x_{t'}^\star)}{\partial \widehat{x}_{t'}^*} \cdot \frac{\partial \widehat{x}_{t'}^*}{\partial x_t^v}. \tag{13}$$

Here, if we adopt the square distance function $d(\widehat{x}_{t'}^*, x_{t'}^\star) = (\widehat{x}_{t'}^* - x_{t'}^\star)^2$, then we have:

$$\frac{\partial d(\widehat{x}_{t'}^*, x_{t'}^\star)}{\partial \widehat{x}_{t'}^*} = 2(\widehat{x}_{t'}^* - x_{t'}^\star). \tag{14}$$

For the CRH model, from Eq. (11), the gradient $\frac{\partial \widehat{x}_{t'}^*}{\partial x_t^v}$ can be calculated as:

$$\frac{\partial \widehat{x}_{t'}^*}{\partial x_t^v} = \begin{cases} \frac{w_v}{\sum_{u\in\mathcal{U}_t} w_u + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_t} w_{\tilde{u}}}, & t' = t, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

For the GTM model, according to Eq. (12), the gradient $\frac{\partial \widehat{x}_{t'}^*}{\partial x_t^v}$ can be calculated as:

$$\frac{\partial \widehat{x}_{t'}^*}{\partial x_t^v} = \begin{cases} \frac{1}{\sigma_v^2\left(\frac{1}{\sigma_0^2} + \sum_{u\in\mathcal{U}_t} \frac{1}{\sigma_u^2} + \sum_{\tilde{u}\in\widetilde{\mathcal{U}}_t} \frac{1}{\sigma_{\tilde{u}}^2}\right)}, & t' = t, \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

After obtaining $\nabla_{x_t^v}\mathcal{L}(\widetilde{X})$, we then can use the projected gradient ascent method to update the value of malicious worker v as follows:

$$x_t^v[r+1] = \text{Proj}_{[x_t^{\min}, x_t^{\max}]}\left(x_t^v[r] + \eta_r \cdot \nabla_{x_t^v}\mathcal{L}(\widetilde{X})\right), \tag{17}$$

where r is the r-th iteration, $\text{Proj}_{[x_t^{\min}, x_t^{\max}]}(\cdot)$ is the projection operator onto the range $[x_t^{\min}, x_t^{\max}]$, and η_r is the step size in the r-th iteration. In this paper, we assume that the attacker initializes the workers with equal weights for the CRH model and equal variances for the GTM model. Algorithm 2 summaries our full-knowledge attack algorithm for the CRH model. The full-knowledge attack algorithm for the GTM model is similar to Algorithm 2, and we omit it here to avoid repetitiveness.

Algorithm 2 Full-knowledge data poisoning attack.

Input: Values from all normal workers x_i^u for $u \in \mathcal{U}, i \in \mathcal{I}_u$.
Output: Malicious workers' values.

1: The attacker initializes the workers' weights.
2: The attacker estimates the aggregated values before attack by iteratively solving Eqs. (2)-(3).
3: **while** the convergence condition is not satisfied **do**
4: The attacker computes the aggregated values and workers' weights $\{\widehat{X}^*, W^*\}$ by iteratively solving Eqs. (2)-(3).
5: The attacker estimates the gradient $\nabla_{x_t^v}\mathcal{L}(\widetilde{X})$ according to Eq. (13).
6: The attacker updates malicious workers' values according to Eq. (17).
7: **end while**
8: **return** x_t^v for $v \in \widetilde{U}, t \in \mathcal{T}$.

4.2 Partial-Knowledge Attack

In the previous section, we showed that the attacker can launch efficient data poisoning attacks to CRH and GTM truth discovery methods when the attacker has full knowledge of the targeted system. However, this could be a restrictive assumption in practice. In this section, we consider a weaker assumption that the attacker only observes part of the values of normal workers on the targeted items. Further, we note that in the partial-knowledge attack, the attacker only needs access to the values of normal workers for the targeted items, i.e., the attacker does not need to know the values of normal workers for non-targeted items.

4.2.1 Aggregated Values Estimation with Bootstrapping. In the partial-knowledge attack, for a given targeted item, it is hard for the attacker to estimate the aggregated value accurately since he only has access to part of the values provided by normal workers. To address this challenge, we leverage the Bootstrapping [8, 13, 14, 21] technique to obtain more accurate before-attack estimated aggregated values for targeted items. Bootstrapping is a classic re-sampling method to estimate a sample distribution. The basic idea of Bootstrapping is to independently sample with replacement from an observed dataset with the same sample size, and perform estimation among these resampled data.

We let $\mathcal{S}_t \in \mathcal{U}_t$ denote the set of normal workers whose observations on the targeted item $t \in \mathcal{T}$ can be accessed by the attacker. In our attack model, once the normal workers' weights are held fixed, the attacker uses the Bootstrapping method to obtain B estimated aggregated value for item t. To be specific, the b-th estimation $\widehat{x}_t^b$, $1 \le b \le B$, can be calculated by the following two steps:

Step 1 (Workers Bootstrapping): The attacker randomly samples a set of normal workers $\mathcal{S}_t^b$ from $\mathcal{S}_t$ with replacement, such that $|\mathcal{S}_t^b| = |\mathcal{S}_t|$.

Step 2 (Value Estimation): The attacker computes the aggregated value for item t in the b-th estimation $\widehat{x}_t^b$ according to Eq. (2) for

Algorithm 3 Partial-knowledge data poisoning attack.

Input: Part of values on the targeted items provided by normal workers in set $\mathcal{S}_t$, $t \in \mathcal{T}$, B.
Output: Malicious workers' values.

The attacker initializes the workers' weights.
//Estimate the aggregated values before attack.
while the convergence condition is not satisfied **do**
  for each $t \in \mathcal{T}$ **do**
    for $b = 1, \cdots, B$ **do**
      The attacker first bootstraps $\mathcal{S}_t^b$ from $\mathcal{S}_t$, then computes $\widehat{x}_t^b$ according to Eq. (2).
    end for
    The attacker computes $\widehat{x}_t^{boot}$ according to Eq. (18).
  end for
  The attacker updates the weight of each normal worker according to Eq. (3).
end while
//Update malicious workers' values.
while the convergence condition is not satisfied **do**
  The attacker computes the aggregated values and workers' weights $\{\widehat{X}^*, W^*\}$ by iteratively solving Eqs. (2)-(3).
  The attacker estimates the gradient $\nabla_{x_t^v} \mathcal{L}(\widetilde{X})$ according to Eq. (13).
  The attacker updates malicious workers' values according to Eq. (17).
end while
return x_t^v for $v \in \widetilde{U}, t \in \mathcal{T}$.

the CRH model or Eq. (5) for the GTM model based on the sampled values provided by workers in set $\mathcal{S}_t^b$.

After the attacker repeats the above two-step procedure B times, the attacker obtains B estimated aggregated values for item t. Then, the final before-attack estimated aggregated value $\widehat{x}_t^{boot}$ can be computed as:

$$\widehat{x}_t^{boot} = \frac{1}{B} \sum_{b=1}^{B} \widehat{x}_t^b. \tag{18}$$

After the attacker uses the Bootstrapping technique to estimate the before-attack value of each item, the attacker then uses the projected gradient ascent method to update malicious workers' value. Algorithm 3 summaries our partial-knowledge attack algorithm for the CRH model, and the partial-knowledge attack algorithm for the GTM model follows a similar procedure. Note that we do not leverage the Bootstrapping method to estimate the after-attack value, since the majority of value of normal workers may be drawn from a certain distribution, while value of malicious workers do not necessarily fit the distribution.

4.2.2 Convergence in Distribution. In this section, we show that for the CRH model, the aggregated value estimated by the Bootstrapping technique converges in distribution to the aggregated value computed by all the observed values at once. We discuss it for the targeted item t, and it can be applied to other targeted items. Note that in this section, we do not assume that the values provided by workers are independent and identically distributed.

We assume that the value of normal worker u on item t follows a normal distribution, i.e., $x_t^u \sim N\left(x_t^\star, \sigma_u^2\right)$, where the variance σ_u^2 measures the quality of values provided by worker u, $u \in \mathcal{S}_t$. We let $X_{\mathcal{S}_t}$ denote the values provided by workers in set $\mathcal{S}_t$, and let $\widehat{\theta}\left(X_{\mathcal{S}_t}\right)$ denote the estimator that the attacker uses. From Eq. (2), the estimator $\widehat{\theta}\left(X_{\mathcal{S}_t}\right)$ of item t can be computed as $\widehat{\theta}\left(X_{\mathcal{S}_t}\right) = \frac{\sum_{u \in \mathcal{S}_t} w_u x_t^u}{\sum_{u \in \mathcal{S}_t} w_u}$. Since $x_t^u \sim N\left(x_t^\star, \sigma_u^2\right)$, we have $\mathbb{E}\left[\widehat{\theta}\left(X_{\mathcal{S}_t}\right)\right] = x_t^\star$, $\mathrm{Var}\left(\widehat{\theta}\left(X_{\mathcal{S}_t}\right)\right) = \frac{\sum_{u \in \mathcal{S}_t} w_u^2 \sigma_u^2}{\left(\sum_{u \in \mathcal{S}_t} w_u\right)^2}$. We further define $\widehat{\mathrm{Var}}\left(\widehat{\theta}\left(X_{\mathcal{S}_t}\right)\right) \stackrel{\text{def}}{=} \frac{\sum_{u \in \mathcal{S}_t} w_u^2 \widehat{\sigma}_u^2}{\left(\sum_{u \in \mathcal{S}_t} w_u\right)^2}$, where $\widehat{\sigma}_u^2 = \frac{\sum_{t \in \mathcal{I}_u} \left(x_t^u - \widehat{x}_t^{boot}\right)^2}{|\mathcal{I}_u| - 1}$ and $\widehat{x}_t^{boot}$ can be computed by Eq. (18). To measure the error between $\widehat{\theta}\left(X_{\mathcal{S}_t}\right)$ and $x_t^\star$, we construct a statistic Q as follows:

$$Q = \frac{\widehat{\theta}\left(X_{\mathcal{S}_t}\right) - x_t^\star}{\left[\widehat{\mathrm{Var}}\left(\widehat{\theta}\left(X_{\mathcal{S}_t}\right)\right)\right]^{1/2} / \sqrt{|\mathcal{S}_t|}}. \tag{19}$$

Since the distribution of Q is usually unknown *a priori*, the attacker could leverage the Bootstrapping strategy to approximate Q. Note that in the b-th estimation of $\widehat{x}_t^b$, the attacker randomly samples a set of workers $\mathcal{S}_t^b$ in $\mathcal{S}_t$. We let $X_{\mathcal{S}_t^b}$ and $\widehat{\theta}\left(X_{\mathcal{S}_t^b}\right)$ denote the values provided by workers from set $\mathcal{S}_t^b$ and the estimator computed based on $X_{\mathcal{S}_t^b}$, respectively. Then the attacker could approximate the distribution of Q as follows:

$$\widehat{Q}_b = \frac{\widehat{\theta}\left(X_{\mathcal{S}_t^b}\right) - \widehat{\theta}\left(X_{\mathcal{S}_t}\right)}{\left[\widehat{\mathrm{Var}}\left(\widehat{\theta}\left(X_{\mathcal{S}_t^b}\right)\right)\right]^{1/2} / \sqrt{|\mathcal{S}_t|}}. \tag{20}$$

Note that $\widehat{\theta}\left(X_{\mathcal{S}_t^b}\right)$ and $\widehat{\mathrm{Var}}\left(\widehat{\theta}\left(X_{\mathcal{S}_t^b}\right)\right)$ can be computed on values $X_{\mathcal{S}_t^b}$. Theorem 1 states that $\widehat{Q}_b$ converges to Q in distribution under the CRH model.

Theorem 1. *Assume that $x_t^u \sim N\left(x_t^\star, \sigma_u^2\right)$, where $u \in \mathcal{S}_t$. Let Q and $Q^\star$ be defined as (19) and (20), respectively. Then, for any real number q, we have that:*

$$\lim_{|\mathcal{S}_t| \to \infty} \left\| \mathbb{P}^\star(\widehat{Q} \le q) - \mathbb{P}(Q \le q) \right\| = 0,$$

where $\mathbb{P}^\star$ stands for the probability computed based on the bootstrapping sample distribution.

Proof. For the targeted item t, the set of workers whose values that can be observed by the attacker is $\mathcal{S}_t$, with the set size $|\mathcal{S}_t|$. Let $G_u(\cdot)$ denote the distribution of a sample x_t^u. Since x_t^u follows from a normal distribution, we have $x_t^u \sim N\left(x_t^\star, \sigma_u^2\right) = G_u\left(x_t^u\right)$. As shown in [30, 45], we need to prove the following conditions to prove Theorem 1:

I) There exists a non-lattice distribution H with mean zero and variance one, and a sequence $k_{|\mathcal{S}_t|}$ with $\frac{k_{|\mathcal{S}_t|}}{\log |\mathcal{S}_t|} \to \infty$, such that $k_{|\mathcal{S}_t|}$ of the population G_u, $u \in \mathcal{S}_t$, are of the form $G_u(x) = H\left(\frac{x - \mu_u}{\sigma_u}\right)$ with σ_u and $u \in \mathcal{S}_t$, bounded away from 0;

II) There exists an $M_1 > 0$ such that $\mathbb{E}\left[|x_t^u|^{3+\delta_1}\right] \leq M_1 < \infty$ for some $\delta_1 > 0$;

III) $\liminf_{|\mathcal{S}_t|\to\infty} \xi^2 > 0$ and $\frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}(\mu_u - \bar{\mu})^2 = o\left(|\mathcal{S}_t|^{-1/2}\right)$;

IV) H is continuous and there exists an $M_2 > 0$ such that for some $\delta_2 > 0$, we have $\mathbb{E}\left[|x_t^u|^{6+\delta_2}\right] \leq M_2 < \infty$,

where $\mu_u = x_t^\star$, $\xi^2 = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}\sigma_u^2$, $\bar{\mu} = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}\mu_u$.

Proof of I: We let H be the standard normal distribution, i.e., $H = N(0,1)$. Then H is a non-lattice distribution since any continuous distribution is non-lattice. If we let $k_{|\mathcal{S}_t|} = |\mathcal{S}_t|$ and $G_u(x) = H\left(\frac{x-\mu_u}{\sigma_u}\right)$, then we have $\frac{|\mathcal{S}_t|}{\log|\mathcal{S}_t|} \to \infty$ as $|\mathcal{S}_t|$ increases.

Proof of II: According to the moments of a normal distribution, we have $\mathbb{E}\left[|x_t^u|^k\right] = \sigma_u^k \frac{2^{\frac{k}{2}}\Gamma\left(\frac{k+1}{2}\right)}{\sqrt{\pi}}$, where k is any non-negative integer, $\Gamma(\cdot)$ is the gamma function. If we let $\delta_1 = 1$, we have $\mathbb{E}\left[|x_t^u|^4\right] = \sigma_u^4 \frac{4\Gamma\left(\frac{5}{2}\right)}{\sqrt{\pi}} \overset{(a)}{=} 3\delta_u^4 = M_1 \leq \infty$, where (a) follows from $\Gamma\left(\frac{5}{2}\right) = \frac{3}{4}\sqrt{\pi}$, which can be shown by the Legendre duplication formula that $\Gamma(z)\Gamma(z+\frac{1}{2}) = 2^{1-2z}\sqrt{\pi}\Gamma(2z)$ with $z = 2$.

Proof of III: $\sigma_u^2 > 0$, $\mu_u = \bar{\mu}$, which completes the proof.

Proof of IV: Since $H = N(0,1)$, then H is continuous. If we let $\delta_2 = 2$, then we have $\mathbb{E}\left[|x_t^u|^{6+\delta_2}\right] = \sigma_u^8 \frac{2^{\frac{8}{2}}\Gamma\left(\frac{9}{2}\right)}{\sqrt{\pi}} = 105\delta_u^8 = M_2 < \infty$.

Since all four conditions above are satisfied, we obtain:

$$\mathbb{P}(Q \leq q) = \Phi(q) + \frac{\beta_1}{6\beta_2\sqrt{|\mathcal{S}_t|}}\left(2q^2+1\right)\phi(q) + o\left(|\mathcal{S}_t|^{-1/2}\right),$$

$$\mathbb{P}^\star(\widehat{Q} \leq q) = \Phi(q) + \frac{\beta_3}{6\beta_4\sqrt{|\mathcal{S}_t|}}\left(2q^2+1\right)\phi(q) + o\left(|\mathcal{S}_t|^{-1/2}\right),$$

where $\Phi(q) = \frac{1}{2\pi}\int_{-\infty}^{q} e^{-t^2/2}dt$, $\phi(\cdot)$ is the derivative of $\Phi(\cdot)$ (i.e., $\phi(\cdot) = \Phi'(\cdot)$), $\beta_1 = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}\mathbb{E}_{G_u}\left[\left(x_t^u - \mu_u\right)^3\right]$, $\beta_2 = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}\sigma_u^3$, $\beta_3 = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t}\left(x_t^u - \rho\right)^3$, $\beta_4 = \zeta^3$, $\rho = \frac{1}{|\mathcal{S}_t|}\sum_{u\in\mathcal{S}_t} x_t^u$, $\zeta = \sqrt{(1/|\mathcal{S}_t|)\sum_{u\in\mathcal{S}_t}\left(x_t^u - \rho\right)^2}$. Proof of II and III also show that $\beta_3 - \beta_1 \to 0$ as $|\mathcal{S}_t| \to \infty$, so we have $\mathbb{P}^\star(\widehat{Q} \leq q) = \mathbb{P}(Q \leq q) + O_p\left(|\mathcal{S}_t|^{-1/2}\right)$ [1]. The proof is complete. □

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 Datasets. We first use a synthetic dataset to demonstrate the effectiveness of the proposed attack methods. In this synthetic dataset, there are 50,000 values in total on 4,000 items generated by 500 workers. We assume that the value of worker u on item i follows a normal distribution $x_i^u \sim N\left(\mu_i, \sigma_u^2\right)$, where μ_i is the ground truth of item i, σ_u^2 is the reliability of worker u. In the experiment, μ_i and σ_u^2 are generated from uniform distributions Uniform(20, 30) and Uniform(0, 30), respectively.

To further demonstrate the advantages of the proposed attack methods, we also conduct experiments on two real-world continuous datasets, which are widely used for evaluating crowdsourcing systems. The first real-world dataset is *Emotion* [37], where the workers in this dataset need to assign a value from the interval [-100, 100] to some texts, indicating the degree of emotion (e.g., surprise) of the text. The second real-world dataset is *Weather* [42], which contains temperature forecast information for 88 major US cities collected from HAM weather [43], Weather Underground (Wunderground) [39], and World Weather Online (WWO) [33]. The statistics of the three datasets are shown in Table 2.

Table 2: Dataset statistics.

Dataset	# Workers	# Items	# Values
Synthetic	500	4,000	50,000
Emotion	38	700	7,000
Weather	152	7,568	936,989

5.1.2 Attack Variants. We test and compare two variants of our proposed attack models, namely:

Full-knowledge attack: The attacker in this attack model is able to inject a set of malicious workers into the crowdsourcing systems. The attacker has full knowledge of the targeted system and sets values of the injected malicious workers according to Algorithm 2.

Partial-knowledge attack: The attacker in this attack model is able to inject a set of malicious workers into the crowdsourcing systems. The attacker has partial knowledge of the targeted system and sets values of malicious workers according to Algorithm 3.

5.1.3 Comparison of Attacks. To demonstrate the effectiveness of our proposed attacks, we compare our attack methods with the following methods.

Random attack: In this attack, for targeted item t, each malicious worker randomly assigns a number from the range $\left[x_t^{\min}, x_t^{\max}\right]$ as the value for item t, where $x_t^{\min}$ and $x_t^{\max}$ are the minimum and maximum values on item t provided by normal workers, respectively.

Maximum attack: In this attack model, for targeted item t, each malicious worker provides the maximum value $x_t^{\max}$ as the value for item t.

5.1.4 Evaluation Metric. In order to measure the effectiveness of different attack models, we use the average estimation error defined in Eq. (7) as our evaluation metric. Since the goal of the attack is to maximize the error of the aggregation results after attack, the larger the estimation error, the better the attack model.

5.1.5 Parameter Setting. Assume the attack size is α (i.e., the number of malicious workers is α fraction of the number of the total workers, $\alpha = \frac{|\widetilde{\mathcal{U}}|}{|\mathcal{U}|+|\widetilde{\mathcal{U}}|}$), and we can inject $\left\lfloor\frac{\alpha|\mathcal{U}|}{1-\alpha}\right\rfloor$ malicious workers into the crowdsourcing systems. Then for a targeted item i, we random select $\left\lfloor\frac{\alpha|\mathcal{U}_i|}{1-\alpha}\right\rfloor$ out of $\left\lfloor\frac{\alpha|\mathcal{U}|}{1-\alpha}\right\rfloor$ malicious workers to attack item i. In this setting, it is guaranteed that for each targeted item, the majority workers are normal workers.

Unless stated otherwise, we use the following default parameter setting: We randomly select some items as targeted items and each targeted item is rated by at least 10 workers. The numbers of targeted items are set to 400, 60 and 100 for Synthetic, Emotion and Weather datasets, respectively. We let $B = 500$. We repeat each

[1] $X_n = O_p(Y_n)$ means $X_n/\|Y_n\|$ is bounded in probability, where X_n and Y_n are random sequences taking values in any normed vector spaces.

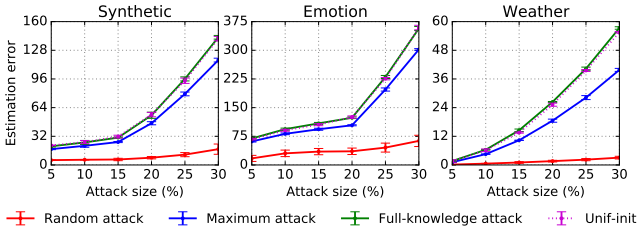

Figure 2: Estimation error with respect to different attack sizes when attacking the CRH model.

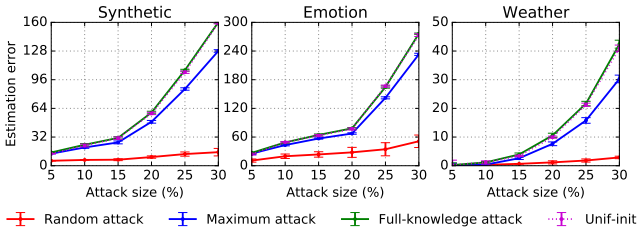

Figure 3: Estimation error with respect to different attack sizes when attacking the GTM model.

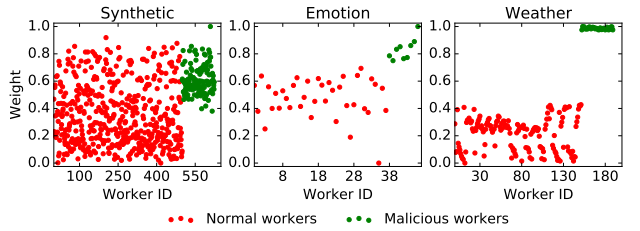

Figure 4: The weights of normal and malicious workers under maximum attack when attacking the CRH model.

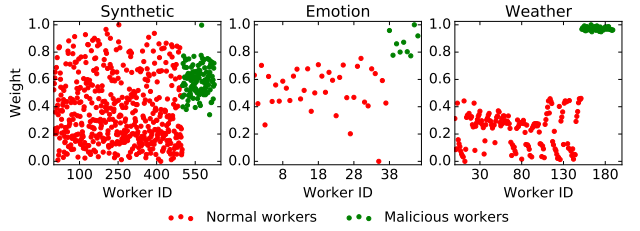

Figure 5: The weights of normal and malicious workers under full-knowledge attack when attacking the CRH model.

experiment for 50 trials and report the average results. All distance functions used in the experiments are squared distance.

5.2 Full-Knowledge Attack Evaluation

For the optimization-based attack strategy, we first consider the full-knowledge attack, where the attacker knows the aggregation algorithm used in crowdsourcing systems (the CRH and GTM methods) and all values provided by normal workers.

Impacts of the attack size: Figures 2-3 show the average estimation errors of different attacks as the attack size (percentage of

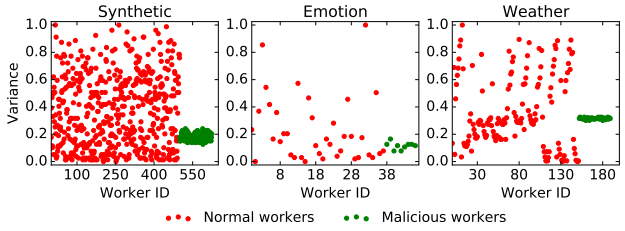

Figure 6: The variances of normal and malicious workers under maximum attack when attacking the GTM model.

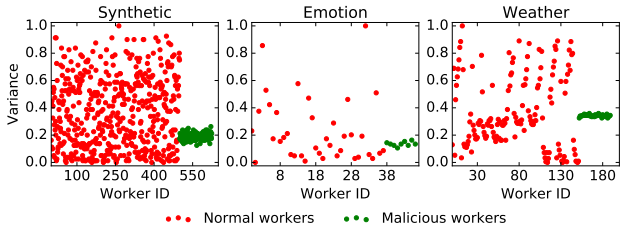

Figure 7: The variances of normal and malicious workers under full-knowledge attack when attacking the GTM model.

malicious workers) increases on three datasets, where the bar is standard deviation. "Unif-init" in Figures 2-3 means for our full-knowledge attack algorithm, the server initializes the CRH/GTM model uniformly at random, and the initial weights/variances are drawn from the uniform distribution Uniform(2, 3). The attacker also initializes the attack model uniformly at random, but the initial weights/variances are drawn from the uniform distribution Uniform(1, 3). First, we observe that our attack is effective in terms of inducing large estimation errors. For instance, in the Emotion dataset, the attacker increases the estimation error to 93.69 by injecting 10% of malicious workers for the CRH model. Second, our proposed attack outperforms the baselines. The reasons are as follows: First, random and maximum attacks are general attack models and not optimized for CRH-based nor GTM-based truth discovery methods. Thus, their attack performance are not satisfactory. Second, our proposed attack model takes the malicious workers' reliability into consideration. Specifically, our attack model abandons some targeted items when there is little chance to increase the aggregation error. Thus, the malicious workers behave similarly with the majority of normal workers. By doing so, the crowdsourcing system may consider the malicious workers as normal workers and increase/decreases their weights/variances, which indirectly increases/decreases these malicious workers' weights/variances on other targeted items. We also find that our attack increases the aggregation error significantly when we inject more malicious workers. By contrast, random attack only slightly increases the estimation error. Another interesting finding is that even though the server and attacker adopt different ways to initialize the workers and not all workers' initial weights/variances are equal, it does not affect the effectiveness of our proposed attack model. From Figures 2-3, we observe that the standard deviations are very small, so we report the average results in the remaining experiments.

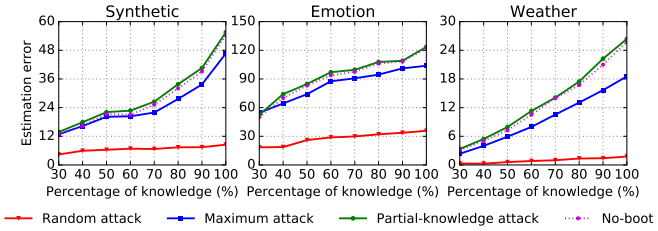

Figure 8: Estimation error with respect to the percentage of knowledge known by the attacker when attacking the CRH model.

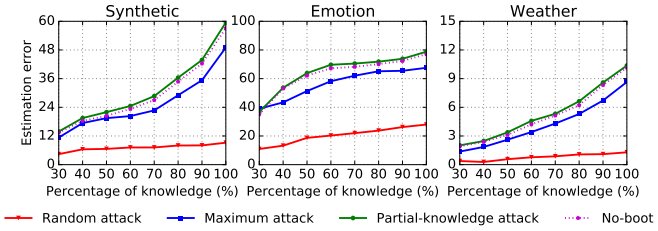

Figure 9: Estimation error with respect to the percentage of knowledge known by the attacker when attacking the GTM model.

Comparisons between weights/variances of normal workers and malicious workers: The CRH/GTM model uses the weights or variances to capture the workers' quality. The key intuition of the CRH/GTM is that a worker should be assigned with a higher weight or lower variance if his values are closer to the estimated results. In this experiment, we investigate the weight and variance distributions for both normal and malicious workers, the attack size is set to 20%. As CRH and GTM models use different ways to measure the reliability of workers, we leverage min-max normalization technique to normalize reliability scores (weights and variances) into the range [0, 1]. The results are shown in Figures 4-7. From Figure 5 and Figure 7, we find that the malicious workers generated by our proposed full-knowledge attack have higher weights or smaller variances comparing with the normal workers. This means that the malicious workers successfully blend into normal workers and it is hard to distinguish normal and malicious workers based on the weights/variances under our attack strategy.

5.3 Partial-Knowledge Attack Evaluation

The amount of values can be accessed by the attacker is another important factor in the attack. Figures 8-9 show the results when the attacker only observes a portion of the values provided by normal workers on targeted items, where the attack size is set to be 20%. The percentage of knowledge in Figures 8-9 represents the fraction of values provided by normal workers that can be observed by the attacker given a targeted item. "No-boot" means the attacker also sets values of malicious workers according to Algorithm 3. However, instead of leveraging the Bootstrapping technique to estimate the before-attack values, the attacker estimates the before-attack values using all observed values at once (without Bootstrapping). Note that in the partial-knowledge attack, the attacker generates the values of malicious workers based only on the observed data. We find that as the attacker has access to more data provided by normal workers, the estimation error increases (i.e., better attack performance). We also find that our method achieves better attack performance than the baselines in most cases. The reason is that our proposed partial-knowledge attack uses the Bootstrapping technique to combine estimated values from multiple bootstrapped values, rather than using all known values at once. This leads to a more accurate value estimation that further slightly enhances the attack performance.

6 DEFENSES

In this section, we propose two defense mechanisms to mitigate the impacts of poisoning attacks on crowdsourcing systems. The basic idea in our defense mechanism design is to arm the crowdsourcing systems with malicious workers detection capability.

6.1 Median-of-Weighted-Average Defense

Although the CRH and GTM models aim to provide robust aggregated results by assigning a larger weight or smaller variance to a worker if this worker's values are closer to the aggregated results, both the CRH and GTM models remain vulnerable to adversarial attacks. To defend potential data poisoning attacks, we design a defense strategy that satisfies two goals: 1) similar to the CRH and GTM models, the server takes the quality of workers into account, and 2) the server should be resilient to potential poisoning attacks. To achieve these goals, we propose the Median-of-Weighted-Average (MWA) defense. In our MWA defense, the server is not aware whether the crowdsourcing system is being attacked.

Since GTM model can only handle continuous labels, while CRH model can deal with both categorical and continuous labels. Thus, in the MWA defense, the server uses a weight parameter to capture a worker's reliability and updates the weights of workers the same way as the CRH model, i.e., weights are updated according to Eq. (3). However, instead of updating the values based on Eq. (2) directly, the server uses the following three steps to estimate the value for each item: 1) the server first sorts workers in ascending order according to the values provided by workers for this item, then partitions the workers (normal and malicious workers) who observe this item into L groups; 2) the server then computes the weighted average of values in each group; and 3) the server takes the median of L values as the estimated value for this item. For each item $i \in \mathcal{I}$, the estimated value $\widehat{x}_i^*$ can be computed as:

$$\widehat{x}_i^* = \text{Median}\left(\frac{\sum_{u\in\mathcal{M}_i^1} w_u x_i^u}{\sum_{u\in\mathcal{M}_i^1} w_u}, ..., \frac{\sum_{u\in\mathcal{M}_i^L} w_u x_i^u}{\sum_{u\in\mathcal{M}_i^L} w_u}\right), \tag{21}$$

where $\mathcal{M} = \mathcal{U} \cup \widetilde{\mathcal{U}}$ is the set of all workers, $\mathcal{M}_i^l$, $l = 1, ..., L$, is the set of workers who observe item i in the l-th group. The MWA defense is summarized in Algorithm 4. In our proposed defense mechanisms, we also assume that all workers are given equal initial weights.

6.2 Maximize Influence of Estimation Defense

We note that, under the MWA defense, malicious workers still exist in the crowdsourcing systems. In this section, we propose another

Algorithm 4 The Median-of-Weighted-Average (MWA) defense.

Input: Values from all workers x_i^u for $u \in \mathcal{M}, i \in \mathcal{I}$.
Output: Aggregated values X^* and worker weights W.

1: Server initializes the workers' weights.
2: **while** the convergence condition is not satisfied **do**
3: For each item, the server partitions workers who observe this item into L groups, then updates the aggregated value according to Eq. (21).
4: Server updates the weight of each worker according to Eq. (3).
5: **end while**
6: **return** X^* and W.

Algorithm 5 Greedy influential worker selection.

Input: Values from all workers x_i^u for $u \in \mathcal{M}, i \in \mathcal{I}$.
Output: Influential worker set $\mathcal{A}$.

1: Initialize $\mathcal{A} = \emptyset$.
2: **while** $|\mathcal{A}| < \lfloor \alpha |\mathcal{M}| \rfloor$ **do**
3: Select $u = \arg\max_{k \in \mathcal{M} \setminus \mathcal{A}} \varphi(k, \mathcal{I})$.
4: $\mathcal{A} \leftarrow \mathcal{A} \cup \{u\}$.
5: **end while**
6: **return** $\mathcal{A}$.

Algorithm 6 The Maximize Influence of Estimation (MIE) defense.

Input: Values from all workers x_i^u for $u \in \mathcal{M}, i \in \mathcal{I}$.
Output: Aggregated values X^* and worker weights W.

1: Server initializes the workers' weights.
2: Server finds the influential worker set $\mathcal{A}$ according to Algorithm 5.
3: Server removes workers in the set $\mathcal{A}$ from the crowdsourcing systems.
4: **while** the convergence condition is not satisfied **do**
5: Server updates the aggregated value of each item with the remaining workers according to Eq. (2).
6: Server updates the weights of the remaining workers according to Eq. (3).
7: **end while**
8: **return** X^* and W.

defense mechanism to detect the malicious workers and remove them from the crowdsourcing systems. However, this defense mechanism requires a *stronger* assumption that the server knows the crowdsourcing system is being attacked and the goal of the attacker. The server also knows there exists $\lfloor \alpha |\mathcal{M}| \rfloor$ number of malicious workers in the system, but the server does not know which items are being attacked. Here, we propose the Maximize Influence of Estimation (MIE) defense to detect the malicious workers in the targeted systems.

For the MIE defense, we let $\mathbb{I}(\mathcal{A}, \mathcal{I})$ denote the influence of removing workers in the set $\mathcal{A}$ on the estimation over all items in $\mathcal{I}$, where the influence here is defined as the change of estimated value. The server wants to find a set of influential workers that have the largest influence on all items in $\mathcal{I}$. The influence maximization

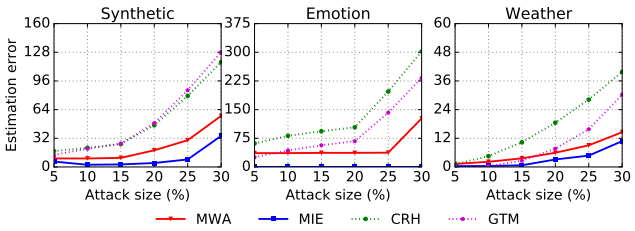

Figure 10: Estimation error of maximum attack when attacking the malicious-worker-aware crowdsourcing systems.

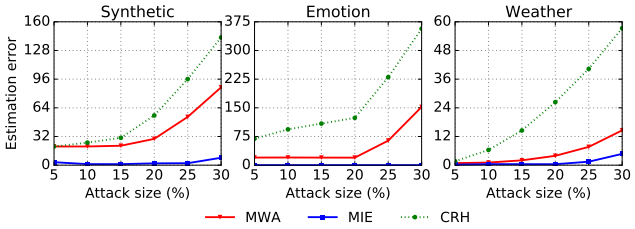

Figure 11: Estimation error of full-knowledge attack when attacking the malicious-worker-aware crowdsourcing systems, where the malicious workers are generated by the full-knowledge attack algorithm for the CRH model.

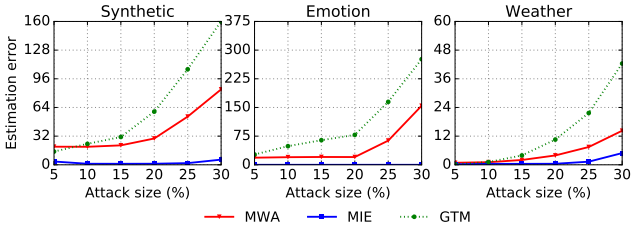

Figure 12: Estimation error of full-knowledge attack when attacking the malicious-worker-aware crowdsourcing systems, where the malicious workers are generated by the full-knowledge attack algorithm for the GTM model.

defense problem can be formulated as:

$$\text{Maximize } \mathbb{I}(\mathcal{A}, \mathcal{I}), \quad \text{subject to } |\mathcal{A}| = \lfloor \alpha |\mathcal{M}| \rfloor, \tag{22}$$

However, this combinatorial influence maximization problem is NP-hard [23] in general. In order to solve Problem (22), we first show how to quantify the influence of one worker, then we will show how to find a subset of $\lfloor \alpha |\mathcal{M}| \rfloor$ workers with the maximum influence. We define $\varphi(u, \mathcal{I})$ as the influence of removing worker $u \in \mathcal{M}$ on the estimation over the targeted crowdsourcing system:

$$\varphi(u, \mathcal{I}) \stackrel{\text{def}}{=} \frac{1}{|\mathcal{I}_u|} \sum\nolimits_{i \in \mathcal{I}} d(\widehat{x}_i^*(\mathcal{M}), \widehat{x}_i^*(\mathcal{M} \setminus \{u\})), \tag{23}$$

where $\widehat{x}_i^*(\mathcal{M})$ represents the after-attack estimated value for item i computed over all workers in set $\mathcal{M} = \mathcal{U} \cup \widetilde{\mathcal{U}}$, the distance function $d(\cdot)$ is squared distance, $|\mathcal{I}_u|$ is the number of items rated by worker u. Therefore, the influence of removing workers from some set $\mathcal{A}$ on the estimation over the targeted system can be defined as the

sum of the influence of individual worker in the set $\mathcal{A}$:

$$\mathbb{I}(\mathcal{A}, \mathcal{I}) \stackrel{\text{def}}{=} \sum_{u \in \mathcal{A}} \varphi(u, \mathcal{I}). \tag{24}$$

We can see that the set influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$ can be naturally computed based on the worker influence $\varphi(u, \mathcal{I})$. Note that even though the worker influence $\varphi(u, \mathcal{I})$ of Eq. (23) shares some similarity with attacker's goal of Eq. (7), they have different meanings. In Eq. (7), the attacker computes the average estimation errors of targeted items before and after attack, while the server in Eq. (23) measures the change of estimated values of all items before and after removing one worker from the crowdsourcing systems.

6.2.1 Approximation Algorithm to Determine $\mathcal{A}$. Although solving Problem (22) is hard, we could design a greedy selection algorithm to approximately find a solution to Eq. (22) by leveraging the submodular property of influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$, which is stated in Theorem 2 as follows:

Theorem 2. *The influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$ is normalized, monotonically non-decreasing and submodular.*

Proof. Define three sets $\mathcal{P}$, $\mathcal{K}$ and $\mathcal{Q}$, where $\mathcal{K} \subseteq \mathcal{P}$ and $\mathcal{Q} = \mathcal{P} \setminus \mathcal{K}$. To simplify the notation, we use $\mathbb{I}(\mathcal{P})$ to denote $\mathbb{I}(\mathcal{P}, \mathcal{I})$. When there is no ambiguity, we let $\mathbb{I}(u)$ denote $\mathbb{I}(\{u\})$ for $u \in \mathcal{M}$. Since $\mathbb{I}(\emptyset) = 0$, the influence function is normalized. We also have $\mathbb{I}(\mathcal{P}) - \mathbb{I}(\mathcal{K}) = \sum_{u \in \mathcal{P}} \mathbb{I}(u) - \sum_{u \in \mathcal{K}} \mathbb{I}(u) = \sum_{u \in \mathcal{P} \setminus \mathcal{K}} \mathbb{I}(u) = \mathbb{I}(\mathcal{Q}) \geq 0$, which shows that influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$ is monotonically non-decreasing. To prove the submodular property, we define an arbitrary set C and we have $\mathbb{I}(\mathcal{P} \cup C) - \mathbb{I}(\mathcal{K} \cup C) = \mathbb{I}((\mathcal{P} \cup C) \setminus (\mathcal{K} \cup C)) = \mathbb{I}(\mathcal{Q} \setminus (\mathcal{Q} \cap C)) \leq \mathbb{I}(\mathcal{Q}) = \mathbb{I}(\mathcal{P}) - \mathbb{I}(\mathcal{K})$. Thus the influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$ is submodular and the proof is complete. □

Based on the submodular property of influence $\mathbb{I}(\mathcal{A}, \mathcal{I})$, we propose a greedy selection method (Algorithm 5) to find an influential worker set $\mathcal{A}$ with $\lfloor \alpha |\mathcal{M}| \rfloor$ workers. To be specific, we first compute the influence of each worker and add the worker with the largest influence to the set $\mathcal{A}$. Then we compute the influence of the remaining workers in the set $\mathcal{M} \setminus \mathcal{A}$, repeat this process until we find $\lfloor \alpha |\mathcal{M}| \rfloor$ workers. Theorem 3 states that Algorithm 5 finds a $(1 - 1/e)$ approximation solution with linear running time complexity.

Theorem 3. *Let $\mathcal{A}$ be an influential worker set returned by Algorithm 5 and $\mathcal{A}^*$ be the optimal influential worker set, respectively. It then holds that $\mathbb{I}(\mathcal{A}, \mathcal{I}) \geq \left(1 - \frac{1}{e}\right) \mathbb{I}(\mathcal{A}^*, \mathcal{I})$.*

Proof. Let $\mathcal{A}^* = \{a_1, a_2, ..., a_{\lfloor \alpha |\mathcal{M}| \rfloor}\}$ be the optimal influential worker set, and $\mathcal{A}_i$ be the worker set after the i-th iteration of Algorithm 5. Thus, we have $\mathbb{I}(\mathcal{A}^*) \stackrel{(a)}{\leq} \mathbb{I}(\mathcal{A}_i \cup \mathcal{A}^*) = \mathbb{I}(\mathcal{A}_i) + \mathbb{I}(\mathcal{A}_i \cup \{a_1\}) - \mathbb{I}(\mathcal{A}_i) + \mathbb{I}(\mathcal{A}_i \cup \{a_1, a_2\}) - \mathbb{I}(\mathcal{A}_i \cup \{a_1\}) + \cdots \stackrel{(b)}{\leq} \mathbb{I}(\mathcal{A}_i) + \mathbb{I}(\mathcal{A}_i \cup \{a_1\}) - \mathbb{I}(\mathcal{A}_i) + \mathbb{I}(\mathcal{A}_i \cup \{a_2\}) - \mathbb{I}(\mathcal{A}_i) + \cdots + \mathbb{I}(\mathcal{A}_i \cup \{a_{\lfloor \alpha |\mathcal{M}| \rfloor}\}) - \mathbb{I}(\mathcal{A}_i) \stackrel{(c)}{\leq} \mathbb{I}(\mathcal{A}_i) + \lfloor \alpha |\mathcal{M}| \rfloor (\mathbb{I}(\mathcal{A}_{i+1}) - \mathbb{I}(\mathcal{A}_i))$, where (a) follows from the monotonically non-decreasing property of influence $\mathbb{I}(\mathcal{A}, t)$; (b) uses the submodular property of influence; and (c) is due to $|\mathcal{A}_i| \leq \lfloor \alpha |\mathcal{M}| \rfloor$. Arranging the terms, we obtain $\mathbb{I}(\mathcal{A}^*) - \mathbb{I}(\mathcal{A}_{i+1}) \leq \left(1 - \frac{1}{\lfloor \alpha |\mathcal{M}| \rfloor}\right) (\mathbb{I}(\mathcal{A}^*) - \mathbb{I}(\mathcal{A}_i))$. Recursively applying the inequality, we have $\mathbb{I}(\mathcal{A}^*) - \mathbb{I}(\mathcal{A}_{\lfloor \alpha |\mathcal{M}| \rfloor}) \leq \left(1 - \frac{1}{\lfloor \alpha |\mathcal{M}| \rfloor}\right)^{\lfloor \alpha |\mathcal{M}| \rfloor} \mathbb{I}(\mathcal{A}^*) \leq \frac{1}{e} \mathbb{I}(\mathcal{A}^*)$. Thus, we have $\left(1 - \frac{1}{e}\right) \mathbb{I}(\mathcal{A}^*) \leq \mathbb{I}(\mathcal{A}_{\lfloor \alpha |\mathcal{M}| \rfloor})$, which completes the proof. □

After using the influence function $\mathbb{I}(\mathcal{A}, \mathcal{I})$ to find the influential worker set $\mathcal{A}$ with $\lfloor \alpha |\mathcal{M}| \rfloor$ workers, the server then removes workers in set $\mathcal{A}$ from the crowdsourcing systems (the server views these workers as malicious workers), and finally estimates the value for each item with the remaining workers. In our MIE defense, we use the CRH model to find influential workers and estimate the aggregated values of items. Our MIE defense is stated in Algorithm 6.

6.3 Defense Evaluation

Figures 10-12 show the average estimation errors of different attacks on the CRH, GTM, MWA and MIE methods, where full-knowledge attacks are considered. The number of groups in MWA defense is set to 5, 4 and 5 for Synthetic, Emotion and Weather datasets, respectively. From Figures 10-12, we observe that both MWA and MIE defenses could mitigate the impacts of malicious workers. MIE achieves a better defense performance compared to MWA. However, MIE and MWA are *not* directly comparable since we assume the server knows the crowdsourcing system is being attacked and the server knows the number of malicious workers exist in the system in MIE. For the MWA defense mechanism, the strategy of dividing workers into groups and computing the median between different groups can only reduce the impact of malicious workers since malicious workers still exist in the system; and if the percentage of malicious workers is high, there would be more malicious workers in each group on average, which leads to less robust weighted average in each group. We also find that even if the server is equipped with malicious workers detection capability, our proposed MWA and MIE defenses may still be vulnerable to poisoning attacks if the percentage of malicious workers is high. For example, the average estimation error of MWA is still 14.57 on the Weather dataset when the attacker injects 30% of malicious workers under the CRH model.

7 CONCLUSION

In this paper, we performed a systematic study on data poisoning attacks and defenses to crowdsourcing systems. We demonstrated that crowdsourcing systems are vulnerable to data poisoning attacks. We proposed an optimization-based data poisoning attack to blend malicious workers into normal workers and increase the estimation errors of the aggregated values for attacker-chosen targeted items. Our attacks are effective under both full-knowledge and partial-knowledge settings. Furthermore, we designed two defense mechanisms to mitigate the impacts of malicious workers. Our results showed that our proposed attacks can increase the estimation errors substantially and our defenses are effective.

ACKNOWLEDGEMENTS

This work is supported in part by NSF grants CAREER CNS-2110259, CIF-2110252, ECCS-1818791, CCF-2110252, CNS-1937786, IIS-2007941, ONR grant ONR N00014-17-1-2417, and a Google Faculty Research Award.

REFERENCES

[1] Scott Alfeld, Xiaojin Zhu, and Paul Barford. 2016. Data poisoning attacks against autoregressive models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[2] Gilad Baruch, Moran Baruch, and Yoav Goldberg. 2019. A little is enough: Circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems*. 8632–8642.

[3] Dimitri P Bertsekas. 1997. Nonlinear programming. *Journal of the Operational Research Society* 48, 3 (1997), 334–334.

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

[5] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012).

[6] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. 2021. FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping. In *NDSS*.

[7] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).

[8] Dehua Cheng and Yan Liu. 2014. Parallel gibbs sampling for hierarchical dirichlet processes via gamma processes equivalence. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 562–571.

[9] Konstantina Christakopoulou and Arindam Banerjee. 2019. Adversarial Attacks on an Oblivious Recommender. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)*. 322–330.

[10] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28, 1 (1979), 20–28.

[11] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39, 1 (1977), 1–22.

[12] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. 2009. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment* 2, 1 (2009), 550–561.

[13] Bradley Efron. 1992. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*. Springer, 569–593.

[14] Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

[15] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2020. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *Usenix Security Symposium*.

[16] Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. 2020. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020*. 3019–3025.

[17] Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. 2018. Poisoning Attacks to Graph-Based Recommender Systems. In *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC)*. ACM, 381–392.

[18] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. 2010. Corroborating information from disagreeing views. In *Proceedings of the third ACM international conference on Web search and data mining*. 131–140.

[19] Daniel A Garcia-Ulloa, Li Xiong, and Vaidy Sunderam. 2017. Truth discovery for spatio-temporal events from crowdsourced data. *Proceedings of the VLDB Endowment* 10, 11 (2017), 1562–1573.

[20] Pierre Hansen, Brigitte Jaumard, and Gilles Savard. 1992. New branch-and-bound rules for linear bilevel programming. *SIAM Journal on scientific and Statistical Computing* 13, 5 (1992), 1194–1217.

[21] Robert V Hogg, Joseph McKean, and Allen T Craig. 2005. *Introduction to mathematical statistics*. Pearson Education.

[22] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. 2018. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 19–35.

[23] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 137–146.

[24] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*. 1885–1893.

[25] Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. 2014. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment* 8, 4 (2014), 425–436.

[26] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. 2014. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 1187–1198.

[27] Xian Li, Xin Luna Dong, Kenneth Lyons, Weiyi Meng, and Divesh Srivastava. 2015. Truth finding on the deep web: Is the problem solved? *arXiv preprint arXiv:1503.00303* (2015).

[28] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2016. A survey on truth discovery. *ACM Sigkdd Explorations Newsletter* 17, 2 (2016), 1–16.

[29] Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2016. Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery. *IEEE Transactions on Knowledge and Data Engineering* 28, 8 (2016), 1986–1999.

[30] Regina Y Liu et al. 1988. Bootstrap procedures under some non-iid models. *The annals of statistics* 16, 4 (1988), 1696–1708.

[31] Chenglin Miao, Qi Li, Lu Su, Mengdi Huai, Wenjun Jiang, and Jing Gao. 2018. Attack under Disguise: An Intelligent Data Poisoning Attack Mechanism in Crowdsourcing. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 13–22.

[32] Chenglin Miao, Qi Li, Houping Xiao, Wenjun Jiang, Mengdi Huai, and Lu Su. 2018. Towards data poisoning attacks in crowd sensing systems. In *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 111–120.

[33] World Weather Online. [Online]. https://www.worldweatheronline.com/

[34] Jeff Pasternack and Dan Roth. 2011. Making better informed trust decisions with generalized fact-finding. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

[35] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11, 4 (2010).

[36] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*. 6103–6113.

[37] Rion Snow, Brendan O'connor, Dan Jurafsky, and Andrew Y Ng. 2008. Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 conference on empirical methods in natural language processing*. 254–263.

[38] Farnaz Tahmasebian, Li Xiong, Mani Sotoodeh, and Vaidy Sunderam. 2020. Crowdsourcing under Data Poisoning Attacks: A Comparative Study. In *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 310–332.

[39] Weather Underground. [Online]. https://www.wunderground.com/

[40] Mengting Wan, Xiangyu Chen, Lance Kaplan, Jiawei Han, Jing Gao, and Bo Zhao. 2016. From truth discovery to trustworthy opinion discovery: An uncertainty-aware quantitative modeling approach. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1885–1894.

[41] Yaqing Wang, Fenglong Ma, Lu Su, and Jing Gao. 2017. Discovering truths from distributed data. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 505–514.

[42] Weather. [Online]. http://lunadong.com/fusionDataSets.htm

[43] HAM weather. [Online]. http://www.hamweather.com/

[44] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. 2015. Is feature selection secure against training data poisoning?. In *International Conference on Machine Learning*. 1689–1698.

[45] Houping Xiao, Jing Gao, Qi Li, Fenglong Ma, Lu Su, Yunlong Feng, and Aidong Zhang. 2016. Towards confidence in the truth: A bootstrapping based truth discovery approach. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1935–1944.

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

[47] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. 2017. Fake Co-visitation Injection Attacks to Recommender Systems. In *NDSS*.

[48] Xiaoxin Yin, Jiawei Han, and S Yu Philip. 2008. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering* 20, 6 (2008), 796–808.

[49] Bo Zhao and Jiawei Han. 2012. A probabilistic model for estimating real-valued truth from conflicting sources. *Proc. of QDB* (2012).

[50] Bo Zhao, Benjamin IP Rubinstein, Jim Gemmell, and Jiawei Han. 2012. A bayesian approach to discovering truth from conflicting sources for data integration. *Proceedings of the VLDB Endowment* 5, 6 (2012), 550–561.