Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

Farnaz Tahmasebian	Date

Robust Crowdsourcing and Federated Learning under Poisoning Attacks

By

Farnaz Tahmasebian

Doctor of Philosophy

Computer Science and Informatics

Li Xiong, Ph.D.

Advisor

Joyce Ho, Ph.D.

Committee Member

Cyrus Shahabi, Ph.D.

Committee Member

Vaidy Sunderam, Ph.D.

Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.

Dean of the Graduate School

Date

Robust Crowdsourcing and Federated Learning under Poisoning Attacks

By

Farnaz Tahmasebian
Ph.D., Emory University, 2021

Advisor: Li Xiong, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the Graduate School
of Emory University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Abstract

Robust Crowdsourcing and Federated Learning under Poisoning Attacks
By Farnaz Tahmasebian

Crowd-based computing can be described in a way that distributes tasks among multiple individuals or organizations to interact with their intelligent or computing devices. Two of the exciting classes of crowd-based computing are crowdsourcing and federated learning, where the first one is crowd-based data collection, and the second one is crowd-based model learning. Crowdsourcing is a paradigm that provides a cost-effective solution for obtaining services or data from a large group of users. It has been increasingly used in modern society for data collection in various domains such as image annotation or real-time traffic reports. Although crowdsourcing is a cost-effective solution, it is an easy target to take advantage of by assembling great numbers of users to artificially boost support for organizations, products, or even opinions. Therefore, deciding to use the best aggregation method that tackles attacks in such applications is one of the main challenges in developing an effective crowdsourcing system. Moreover, the original aggregation algorithm in federated learning is susceptible to data poisoning attacks. Also, the dynamic behavior of this framework in terms of choosing clients randomly in each iteration poses further challenges for implementing the robust aggregating method in federated learning. In this dissertation, we devise strategies that improve the system's robustness under data poisoning attacks when workers intentionally or strategically misbehave.

Robust Crowdsourcing and Federated Learning under Poisoning Attacks

By

Farnaz Tahmasebian
Ph.D., Emory University, 2021

Advisor: Li Xiong, Ph.D.
Co-Advisor:

A dissertation submitted to the Faculty of the Graduate School
of Emory University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Acknowledgments

My first words of appreciation go towards my advisor, Prof. Li Xiong, for her continuous support and her limitless patience, motivation, and guidance throughout my Ph.D. during these years. She guided me through each step, and I have learned many lessons from her, from spotting interesting problems to writing a structured paper. During my Ph.D. years, Li helped me grow as a researcher and as a person. She always encouraged our group to take care of ourselves and choose an activity besides researching and improving our soul and bodies. I learned so much from her about doing research and being a good mentor with care and patience.

Besides my advisor, I would like to thank the rest of my dissertation committee: Prof. Sunderam, Prof. Shahabi, and Dr. Ho, for their constructive and comprehensive feedback on this thesis. Their remarks greatly improved the quality of this thesis.

Special thanks go to people who were my mentors during my academic years and helped me grow, settled down and support me long the way: Layla Pournajaf and Zahra Ghafouri.

I want to thank my fellow graduate students at AIMS, collaborators, and the staff at the computer science department for all of their passionate support and input. Thank you for enriching my Ph.D. studies with insightful discussions, feedback, support, and relaxing chitchats. I want to thank them for being here for me and for all the fun we have had: Layla Pournajaf, Jian Lou, Mani Sotoodeh, Ardavan Afshar, Jinfei Liu, Pengfei Tang, Wengie Wang, Shuacheng Ma, Reza Karimi, Ali Ahmadvand, Ziwei Dong, Payam Karisani, Shabnam Hossein, Daniel Garcia ulloa, Rongmei Lin, Qiuchen Zhang, Jing Ma,Forough Zarean.

I want to thank my parents Shahla and Bahram, who always believed in me and my brothers, Shahram, Iman, and Ehsan, and my sister, Shahrzad, who guide and support me along the way. And last but not least, all this would be impossible without my husband, who join me on this journey. Words cannot describe my gratitude and love for him. I thank him for all the understanding and support that gave me the energy to work and improve.

This research was funded by Air Force Office of Scientific Research (AFOSR) and DDDAS Program under grant FA9550-12-1-0240.

To my dear husband Azad
who have always supported me with his patience and encouragements.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Crowd-based computing can be described in a way that distributes tasks among multiple individuals or organizations to interact with their intelligent or computing devices. Two of the exciting classes of crowd-based computing are crowdsourcing [6, 19, 43, 44] and federated learning [47, 63] where the first one is crowd-based data collection and the second one is crowd-based model learning.

The nature of crowd-based computing associate with uncertainty about the quality of data provided by individuals or organizations. In this thesis, we devise strategies that improve the system's robustness under data poisoning attacks when workers intentionally or strategically misbehave.

1.1.1 Crowd-based data collection: Crowdsourcing

A large amount of labeled data are crucial for machine learning, retrieval, and search. Ideally, ground truth labels are collected from experts, but such sources are often too slow or costly; therefore, crowdsourcing is used as a data annotation method. Crowdsourcing is a crowd-based data collection in which organizations or individuals obtain data or service from a large or relatively open group of users, or *crowd*. It has been increasingly used in modern society for data collection in various domains

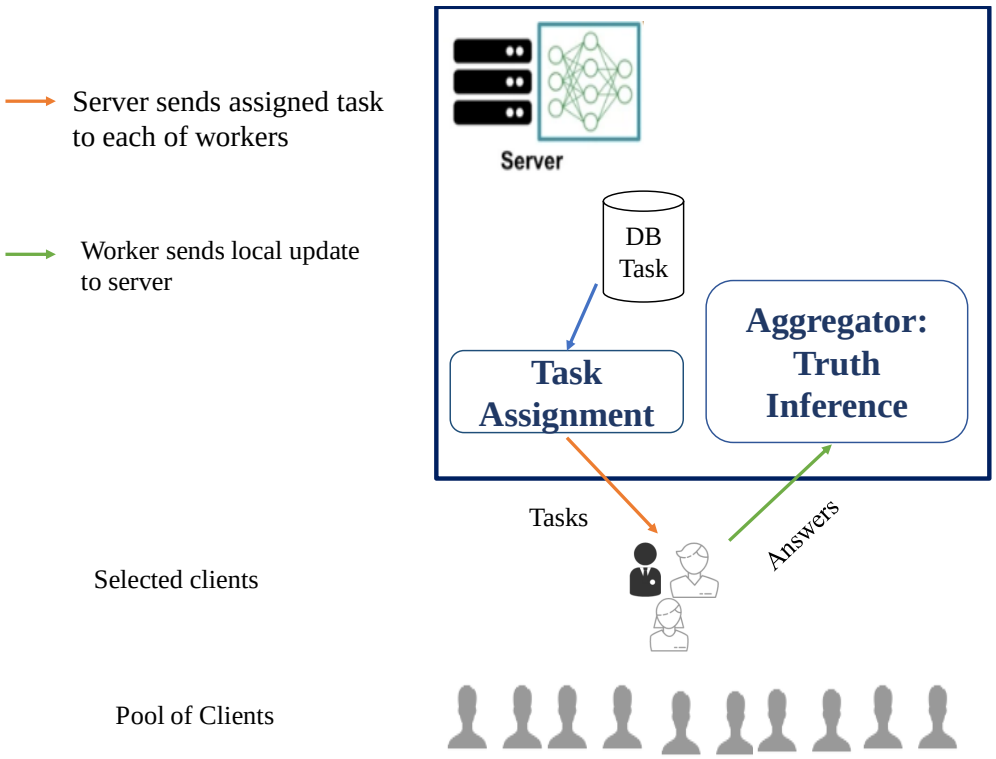


Figure 1.1: Framework of a Crowdsourcing System

such as image annotation or real-time traffic reports. Amazon Mechanical Turk (MTurk) [11, 37] is one of the most pervasive crowdsourcing marketplaces, in which requesters design their tasks and publish those tasks to the MTurk platform. Those tasks are requiring human intelligence, such as labeling objects in an image or flagging inappropriate content.

Another example is Waze [87], navigation and traffic sharing application. Users can report the traffic status at various locations using the app, which is then aggregated to update the traffic condition shown on the map. Figure 1.1 establishes a general framework of crowd-based computing; the essential entities of this system are identified as follows.

1. *Server* is the entity that collect tasks from requesters to infer the correct answer/label for each tasks.

2. *Workers or Clients* are entities that perform tasks and submit the results to the platform.

3. *Tasks* are entities that need contextual knowledge and the cognitive ability of the crowd (workers).

4. *Aggregator* is an entity that collects workers' answers and infers the truth of each task.

5. *Task Assignment* is an entity that assigns tasks to workers.

Truth Inference in Crowdsourcing

A vital component of these crowdsourcing applications is *truth inference* which aims to derive the answers for the tasks by considering all collected answers from workers, e.g., the objects in the image, or real traffic condition of the road, by aggregating the user-contributed data. Truth inference [20, 43, 56, 74] is a challenging task due to the open nature of the crowd. First, the number of available ratings for each task varies significantly among tasks, and thus, missing ratings become a hurdle. Second, the reliability of the workers can vary. For example, in the Waze application, it is quite common for some users not to care to report at all or to report the traffic condition carelessly. Therefore, estimating the level of trust one has in workers' responses and ultimately inferring the truth value for the tasks by aggregating their responses becomes complicated. Finally, the crowdsourcing applications may be subject to *data poisoning attacks* [40, 48, 80] where malicious users may intentionally and strategically manipulate reports to mislead the system to infer the wrong truth for all or a targeted set of tasks. In the Waze example, attackers might want to take the road with the least traffic themselves by deceiving the Waze application into wrongly indicating heavy traffic on that specific road. This misleading in the system can be often achieved via Sybil attacks [16, 22, 95, 101] where an attacker creates a large number of Sybil workers to report wrong answers strategically. Studies have shown [3, 60] these attacks are beneficial towards the organization or individuals that

hire Sybil workers. For example, increasing one-star rating by Sybil workers on Yelp increases the revenue up to 9%.

The simplest method for truth inference is *majority voting*, where the truth of the task will be the one chosen by the majority of the workers assigned to the task. Since workers' reliability is not considered, majority voting may yield inaccurate results in the presence of unreliable or malicious workers. Considerable research has been done on improving the accuracy of truth inference methods, including optimization-based methods [39,53], probabilistic graphical model based methods [19,44,70,85,107], and neural network based methods [29,34,98]. Zheng et al. [107] evaluated truth inference methods under "normal" settings where workers may have varying reliability but do not intentionally or strategically manipulate the answers. These methods construct models that explicitly or implicitly consider workers' credibility, which create some form of defense against unreliable or malicious workers.

We identify two main challenges in truth inference approaches in crowdsourcing. These challenges are 1) *an accurate assessment of the robustness of truth inference techniques requires modeling of realistic threat models and understanding the robustness of existing truth inference methods*, and 2) *enhancing the robustness of existing truth inference methods under such adversarial attacks.*

1.1.2 Crowd-based model learning: Federated Learning

In the conventional machine learning techniques, the data points are collected, and the training process is performed on a server. These traditional learning can be considered as a centralized system where all procedure is performed on the server. However, in federated learning, clients do not have to share their data. Instead, the data points are partitioned across the devices of selected clients. Clients can have varying numbers of data points. In the federated learning system, responsibilities are divided among a server and the clients in a way that a federation of clients takes over a significant amount of work. However, there is still one central entity, a server, coordinating everything.

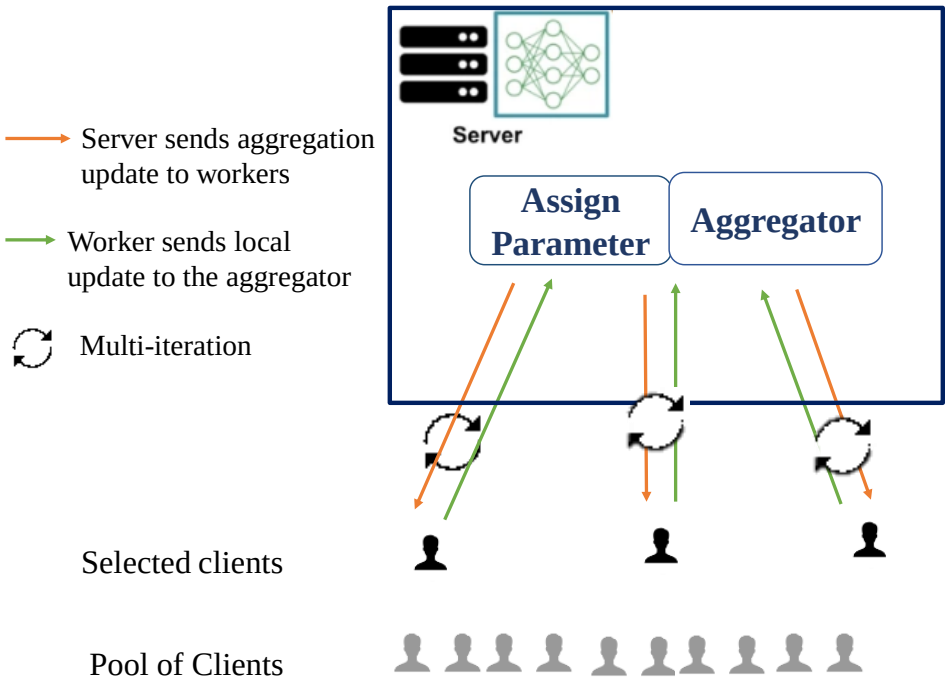


Figure 1.2: Framework of a Federated Learning System

Most of the learning models require access to a large amount of data and various distribution to build a meaningful and sensible model. In reality, organizations have some concerns regarding sharing their valuable datasets with another third party, one of these concerns that get more attention these days is data privacy. For example, hospitals would not share their patient data to train a model in a centralized manner. Still, they can utilize crowd-based computing and collaborate to train a global model leveraging each institution's electronic health records (EHR) without compromising patient privacy through federated learning settings. Federated learning is a crowd-based model learning that enables a global model trained in a decentralized manner while protecting clients' data privacy. In federated learning, clients have the same model as the global server, train the model on their data locally, and upload their updates to the server. The server aggregate these updates to find the final models' parameters [62–64].

Google was the first company that implements federated learning on a large scale to improve the quality of google keyword suggestions by training a global model. Another example is leveraging federated learning in a production level setting to enhance the Firefox search results without collecting the users' actual data [93].

The essential entities of the federated learning are identified as follows.

1. *Server* is the entity that chooses the clients from a pool of clients and iteratively aggregates clients' provided parameters.
2. *Clients Assignment* is an entity that assigns tasks to clients.
3. *Aggregation* is an entity that collects client's results and aggregate them.
4. *Workers or Clients* are entities that hold the local dataset and training model on their local data and submit the results to the platform.
5. *Tasks/Parameters* are entities that define a general machine learning model.

Figure 2.1 shows a general framework of federated learning and offers a graphical representation of the process. The server selects K clients from the pool. They receive the current model and compute updates using their data. Training happens on several devices. Subsequently, the new models are pushed to the central server and averaged to get the new model. Then, the global server pushed the latest model to all selected client's devices, and the process starts again. While requesting updates from all users would lead to more stable model improvements, it would also be costly. Only querying a subset of them makes it more feasible to run many iterations efficiently. This training process is repeated until the model parameters converge, as determined by an appropriate criterion.

One of the key important components in a federated learning system is the aggregation algorithm. The known aggregation algorithm in federated learning is FedAvg [63] that takes the average of locally updated models. However, this traditional aggregation method is vulnerable to data poisoning attacks and byzantine clients,

and just a few clients can compromise the entire performance of the shared model. Considerable research has been proposed different byzantine-robust aggregation algorithm [8, 32, 72, 96]. Some of these byzantine-robust aggregators are successfully defending against some data poisoning attacks in federated learning.

The critical question that remains unanswered is how robust these crowd-based computing models under data poisoning attacks. In crowdsourcing, malicious workers may disguise themselves as ordinary workers by providing reliable answers for specific tasks such that they escape the worker reliability model while providing the wrong answer for other targeted tasks. Thus, it is essential to understand the various types of data poisoning attacks and evaluate how the aggregation methods behave under such attacks to ultimately build a robust framework complementary to other Sybil detection methods. Moreover, a crucial challenge in federated learning is that the small number of malicious clients could hurt the global model performance. Therefore, it is vital to avoid aggregating malicious updates into the global model parameters. Thus, estimating the reliability of the client's updates at each iteration could help to reach a solution against a poisoning attack. One potential approach to estimate clients' reliability is taking advantage of the truth inference approaches, which have been extensively studied in crowdsourcing. Therefore, it is important to assess the various architecture to adopt federated learning under truth inference methods. This work explores deploying a robust federated learning model under adversarial attack.

1.2 Research Contributions

In this dissertation, we propose truth inference approaches for the robustness of crowd-based computing under data poisoning attacks with the motivation mentioned earlier. These approaches provide a robust aggregation framework for a system under the adversarial environment. In Chapter 3, we explore the effect of data poisoning attacks on crowdsourcing problems. We propose a comprehensive data poisoning

attack taxonomy for truth inference in crowdsourcing and systematically evaluate the state-of-the-art truth inference methods under various attack models. This research is accepted at the DBSec conference. Chapter 4 extends our work by presenting two enhanced truth inference methods to address the robustness issues in crowdsourcing systems under attack published at the SMDS conference. In Chapter 5, we extend our approach to build the robust aggregation method in a federated learning system that deals with a dynamic environment and more susceptible to data poising attacks. In a federated learning system, the general machine learning model is trained through multiple rounds of collaboration between a server and clients. Therefore applying a simple truth inference method is not suitable for federated learning. Since each iteration, the server could select different clients to participate in the training process. Also, in each round, each client's behavior could change and affect the performance of the model. The rest of this section highlights the details of our contributions.

1.2.1 Evaluation of Truth Inference Methods under Data Poisoning Attack (Chapter 3)

This chapter proposes an extensive experimental evaluation for existing truth inference methods under an adversarial environment. Many truth inference approaches are proposed in the literature for various tasks and workers models. However, it is still challenging to adopt the most suitable one for the system under a data poisoning attack.

Existing truth inference approaches consider the low-quality workers who are not experts and might randomly answer the tasks. The performance of current inference methods under malicious workers who professionally provide fraudulent answers to the system has not been comprehensively studied yet. Thus, the lack of an extensive evaluation to show each method's strengths and limitations in various attack strategies is felt; therefore, building a framework to evaluate and compare these methods remains a challenge. We summarize our contributions below.

- We present a comprehensive data poisoning attack taxonomy in crowdsourcing. We analyze the attacks along different dimensions, including attack goal (targeted vs. untargeted), adversarial knowledge (black-box vs. white-box), and attack strategy (heuristic vs. optimization-based).
- We design heuristic and optimization-based attacks that can be used against various truth inference methods as part of our evaluation methodology. The heuristic-based attacks assume black-box or no adversarial knowledge and model the worker behavior using a confusion matrix [19] and an additional disguise parameter to hide their malicious behavior. The optimization-based attacks assume white-box or full adversarial knowledge, including the truth inference methods being used and other workers' answers, and are adapted from existing optimization-based attacks [65] while making them more generic so they are applicable to broader types of truth inference methods.
- We propose several metrics to evaluate the robustness or susceptibility of the methods against data poisoning attacks. We experiment on synthetic and real-world datasets and analyze the results over various parameters, including the percentage of malicious workers, different attack parameters, and the sparsity of the crowdsourcing dataset.

1.2.2 Enhanced Truth Inference Method under Data Poisoning Attack (Chapter 4)

Crowdsourcing is widely studied in the literature and is employed in many businesses due to its efficiency, simplicity of use, and effectiveness. One important element of the crowdsourcing system derives truth from the answer provided by workers called *truth inference*. The goal of this chapter is to tackles the poor performance of crowdsourced data against data poisoning attacks. Given the increasing demand in crowdsourcing, reaching high-quality labeling is becoming essential. In an adversarial environment,

a certain percentage of attackers may behave maliciously and strategically in order to flip the true label of tasks.

This chapter investigates a solution that brings robustness in terms of the performance of truth inference methods in the crowdsourcing system. First, we present a solution based on identifying sensitive tasks that has a higher chance of manipulation by adversaries. Then, it is shown that incorporating this information can preserve the robustness of the system in the presence of data poisoning attacks. We summarize our contributions below.

- We present a data augmentation method (EdgeInfer) focused on boundary tasks that can be used to enhance the robustness of existing truth inference methods against potential data poisoning attacks. The intuition behind this is that boundary tasks are more likely to be targeted by malicious workers to achieve a successful attack due to the weak agreement among contributing workers. This method can be used as a preprocessing step to enhance existing truth inference algorithms.

- As shown in the experimental survey [82] the state-of-the-art methods based on neural networks and PGM perform better and generally more robust. Therefore, we propose Edge-NN and Edge-PGM that are based on neural networks and PGM models and utilizing prior information to enhance these methods. Edge-NN offers an enhanced neural network-based inference method by replacing raw data distribution based prior with a stronger prior inferred from a probabilistic graphical model (PGM). By incorporating the prior, the method takes advantage of two sources of knowledge from two distinctive and complementary models, which promises a boosted performance in terms of accuracy and robustness. In Edge-PGM, we propose an enhanced truth inference method based on PGM by taking advantage of boundary tasks and curating a better prior for it. This PGM inference method incorporates the difficulty level of tasks into their model. However, the estimated difficulty level of tasks is not quite certain. Therefore, utilizing a more substantial prior of the difficulty level

of tasks can enhance the truth inference method.

- We conduct experiments using three real datasets under different data poisoning attacks in crowdsourcing. The results verify that the proposed approach outperforms state-of-the-art truth inference methods under a variety of attack scenarios.

1.2.3 Robust Federated Learning under Data Poisoning Attack (Chapter 5)

This chapter extends the usage of the truth inference method for federated learning systems with a dynamic environment with randomly choosing the workers/clients from a pool and more vulnerable to adversary environment. Our goal is to mitigate the effect of adversary clients on the global model. To achieve this goal, we propose a novel aggregation method for federated learning that deals with malicious clients with the following contributions.

We propose to use a trustworthy approach, originally proposed in the crowdsourcing domain, against such attacks in a federated learning setting to measure the trustworthiness of provided updates. The intuition is that in federated learning, benign clients' provided updates would be similar to each other in each round. Therefore the global server can implicitly learn to rely more on clients' updates similar to each other. Measuring the clients' trustworthiness can be estimated by adopting a truth inference mechanism. We derive our aggregation algorithm by incorporating the reliability of each clients' provided parameters. Based on estimated reliability, the one is chosen if their reliabilities are not far from the other clients. Finally, the local parameters are aggregated based on clients' estimated reliability in the proportion of the number of local data trained. We briefly summarize our contributions as:

- We develop an aggregation method that describes the dependencies of locally shared parameters and the clients' reliability. We present an application of

adopting the truth inference method in federated learning to estimate the client's reliabilities' true state.

- We further enhance the aggregation method by considering the statistical weights from previous rounds that explicitly describe the clients' temporal correlations.

- We compare our proposed method to several baselines by conducting experiments on three image datasets. Our proposed aggregation mitigates the impact of attacks in the IID setting and outperforms other baselines.

Chapter 2

Related Works

In this chapter, we provide a brief review of crowdsourcing. Then, we review truth inference methods along with adversarial attacks on crowdsourcing. Afterward, we describe adversarial attacks on federated learning. Subsequently, we survey the existing defense and robustness methods in federated learning.

2.1 Crowdsourcing

Crowdsourcing has emerged as a practical paradigm to efficiently address labeling large datasets and performing various learning tasks by utilizing hundreds of workers (i.e., the crowd). Access to crowd resources has been made easier due to public crowdsourcing platforms, such as Amazon Mechanical Turk(MTurk), CrowdFlower, and Upwork [10, 51]. Even though crowdsourcing can be efficient and relatively inexpensive, inference of true labels from the noisy responses provided by multiple unknown expertise workers can be challenging, especially in the typical unsupervised scenario, where no ground truth data is available. Therefore controlling the quality of the crowdsourcing framework is getting more attention. One important component in the crowdsourcing framework that improves the quality of the system is truth inference components.

2.1.1 Truth Inference Methods

Crowdsourcing aggregates the wisdom of the crowd (i.e., workers) to infer the truth label of tasks in the system, which is called truth inference. Truth inference is a key component of crowdsourcing, and inferring the truth necessitates a method to handle workers' data quality by assessing the behavior of workers and measuring the reliability of the workers and the aggregation method. There are four main categories of truth inference techniques: 1) direct computation, 2) optimization, 3) probabilistic graphical model (PGM), and 4) neural networks. Direct computation aggregates the workers' answers by majority voting while treating all workers equally or heuristically assigning a weight to the workers [39]. Optimization based methods [39,52,53,108,111] treat the estimated labels and worker reliability values as unknown variables and use an optimization approach to find them. Probabilistic graphical models (PGM) explicitly model workers' reliability and possibly workers' dependency to estimate the labels [19, 20, 43, 44, 85, 92]. A recent experimental study compared various truth inference methods [107].

The simplest truth inference method is majority voting, which works well if all workers provide answers to all of the tasks. However, it fails in complicated cases. Comparably, the probabilistic graphical model (PGM) has a better performance and can be solved by diverse techniques, such as EM algorithms, convex optimization, and variational inference. In general, optimization and PGM based methods follow an iterative EM-based approach with two steps: 1) inferring the label of tasks given the estimated reliability of workers, and 2) computing workers' reliability given the current inferred labels of tasks. More recently, unsupervised neural network based approaches [29, 98] has been proposed by incorporating answers of each task in a neural network with its output as the inferred label of the task. Other approaches based on tensor augmentation and completion with limited performance have also been suggested [110]. One of the famous PGM based models is the Dawid & Skene's model (D&S) [19] that models worker reliability using a confusion matrix. The confusion matrix π^{w_i} for worker w_i is an $|L| * |L|$ matrix where element $\pi^{w_i}_{p,q}$ denotes the

probability of worker w_i reporting label q given the truth label p in $\mathbf{L}$. Also, the D&S algorithm focused on applying an EM (Expectation-Maximization) algorithm which consists of two steps: 1) an expectation step (E-step), and 2) a maximization step (M-step). In the E-step, inferring the label of tasks given the estimated reliability of workers, and in the M-step, workers' reliability is estimated based on the inferred truth labels. EM algorithm iteratively applies to find the maximum likelihood estimation (MLE) for the inferred truth labels and the confusion matrices $\pi^{\mathbf{W}}$. Truth inference methods in crowdsourcing can be classified based on worker model and techniques [107].

Some truth inference methods do not have an explicit worker model. In contrast, others employ a worker model that can be represented as: 1) a single *worker reliability* or *penalty* parameter related to how reliable the answers by each worker are [20, 92], 2) a *confusion matrix* that captures a worker's probability of providing a certain label given the true label [19, 44, 85], 3) worker *bias/variance* [74] for numeric tasks, or 4) *confidence* [53] related to the number of tasks answered by each worker.

Many of the truth inference methods utilize the confusion matrix in their modeling. The confusion matrix π^{w_i} for worker w_i is an $|L| * |L|$ matrix where element $\pi^{w_i}_{p,q}$ denotes the probability of worker w_i reporting label q given the truth label p in $\mathbf{L}$. Assuming a binary label set $\mathbf{L} = \{0, 1\}$, we simplify the notation using two variables, α and β, where $\alpha = \text{pr}(c^{w_i}_{t_j} = 1 \mid z^*_{t_j} = 1)$ and $\beta = \text{pr}(c^{w_i}_{t_j} = 0 \mid z^*_{t_j} = 0)$, indicating the probability of worker w_i correctly reporting task t_j given its true label 1 or 0 respectively.

2.1.2 Data Poisoning Attacks in Crowdsourcing

Data poisoning attacks [27, 65, 66] have been recently studied against representative truth inference methods, namely D&S [19] and PM [53] assuming attackers have full knowledge of other workers' answers and the inference method being used. They formulate an optimization problem and assume the adversary does not know the ground truth of the tasks. Hence the optimization goal is to maximize the number of flipped

labels after the attack when compared to inferred labels before the attack. The attack also attempts to maximize the attackers' collective confusion matrix parameters (reliability) inferred by the system. Intuitively, this will help them obfuscate their malicious nature and be more successful in misleading the system.

Comparison with data poisoning attacks in machine learning: Data poisoning attacks have been increasingly studied in recent years for various machine learning (ML) algorithms [7, 30, 35, 40, 48, 75, 80, 88].There are a few distinct differences between data poisoning attacks in crowdsourcing and machine learning (ML):

First, most existing data poisoning attacks in ML deal with supervised models; hence the goal is to degrade the model's performance (trained on the poisoned data) on a separate validation dataset. The truth inference problem typically does not assume the ground truth of tasks are available for training and thus is formed as an unsupervised problem. Second, the data poisoning attacks in ML typically assume a certain number or percentage of records are poisoned by an attacker, and all the features associated with the poisoned record (e.g. an image) can be altered to carry out the attack. For crowdsourcing, it is often the case that a certain number or percentage of workers may be malicious (due to Sybil attacks). Here a malicious worker can only change the answers for tasks assigned to him/her. Finally, ML problems typically have a rich set of features for each record, while in crowdsourcing for each task only a set of ratings from workers is available. Hence we believe crowdsourcing systems are more susceptible to data poisoning attacks, due to its open and unsupervised nature, and lack of rich features for the truth inference problem.

Shilling attack [12, 15, 33, 67, 71, 104] is another type of data poisoning attacks in recommender systems where intruders attempt to simulate the behavior of a subset of users, leading to their dissatisfaction in a recommender system, and disruption in other users' activities [33, 71]. Various types of shilling attacks such as using the random or average ratings as the adversary response [12, 67] have been studied. As a defense mechanism, supervised, unsupervised and statistical approaches have been studied such as KNN-based, k-means clustering technique, hidden Markov model

and sparse variation of the Hv-score metric [33, 71, 89, 102, 109]. The main difference between recommender systems and truth discovery is that the true labels of tasks in recommender systems are subjective as each user may have a personal opinion, and therefore a tailored ground truth, while there is universal ground truth for truth discovery. Besides, the answers are usually numeric in recommender systems, reflecting each user's personal preferences, which falls outside the current work scope.

Other related attacks include spammer [21, 36, 37, 73, 92] and sybil [13, 50, 79, 84, 87, 99–101, 103] attacks. In a spammer attack, workers (bots) randomly submit answers to tasks [21, 36, 37]. In sybil attacks, infiltrators create fake identities to affect the performance of the system [101, 103]. For example, the attack and defense methods in social media and IoT typically utilize metadata such as connectivity of graph and relationship between each node, IP address, and MAC address. Sybil and spammer attacks mainly focus on the system infiltration as part of the attack and can be considered as means to achieve a data poisoning attack. We consider the data poisoning attacks in this paper assume adversaries have successfully created or compromised multiple workers and injected strategic answers.

2.1.3 Matrix Completion

Matrix completion has received a lot of attention these past years since it brings a unique property and could be applicable in various domains. Its application varies from wireless communications, traffic sensing to integrated radar and communications [55]. Matrix completion is a promising technique that can recover an intact matrix with low-rank property from incomplete data. One example is the rating matrix in the recommendation systems representing users' tastes on products []. Since users expressing similar rating on multiple products tend to have the same interest for the new product, columns associated with users sharing the same interest are highly likely to be the same, resulting in the low-rank structure. One benefit of the low-rank matrix is that the essential information, expressed in terms of the degree of freedom, in a matrix is much smaller than the total number of entries. Therefore,

even though the number of observed entries is small, we still have an excellent chance to recover the whole matrix.

Various matrix completion methodologies have been developed from different perspectives [41,55,91], one of these methodology is matrix factorization [91]. The basic idea behind the matrix factorization is to utilize two low-rank matrices to represent the objective matrix, assuming that the original matrix's rank is known. A fundamental challenge in the study of matrix completion is that, in some applications, the revealed entries will be inaccurate or corrupted. Some studies have utilized the matrix completion method to robust their model. Yang et al. [94] propose a defense mechanism that leverages matrix estimation in deep neural networks. In their proposed method, a data matrix from noisy and incomplete observations is recovered. Some studies apply tensor completion concepts to crowdsourcing problems [54,110]. They propose to augment the data tensor with an extra ground truth layer and explore various tensor completion techniques to infer the true labels in the ground truth layer.

2.2 Federated Learning

When the size of data is enormous or data is distributed on a series of devices, or data privacy is considered a sensitive matter, utilizing federated learning is an excellent solution. Federated learning was first introduced by Google [47]. In a federated learning setting, multiple devices, end-user devices such as mobile phones, or organization infrastructure such as hospital servers, contribute to learning a machine learning model. The machine learning model can be a deep neural network and a simpler model, with fewer parameters such as a logistic regression model.

In federated learning, the original training data never leaves the corresponding local device that collected it. Each device keeps a version of the same model; the model's updates are shared with a central server, which averages the new models from all participating devices. Once a new version of the model has been trained,

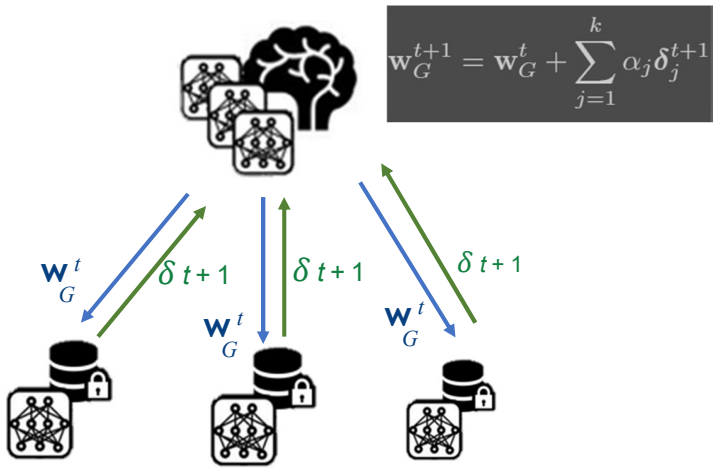


Figure 2.1: Framework of a Federated Learning

it is pushed back down to all devices. This process repeats continuously until the model converges.

Figure 2.1 displays a graphical representation of this process. First, the global server model is pushed down to clients, which subsequently trains the model on local data. To make federated learning practical, the optimization problem in the context of federated learning needs to be solved. Naively, SGD can be applied to the federated optimization problem. The single batching gradient calculation can be achieved by a communication round in federated network.

2.2.1 Aggregation Methods in Federated Learning.

A considerable amount of research has been conducted to study federated learning models in terms of better communication bandwidth and malicious clients in the system [62, 76, 77, 93]. Federated learning trains a global machine learning model while the clients can keep their data private. In a federated learning system, a central node called a server controls the learning process. It aggregates the information from the clients that train the model locally using their local datasets. The clients send the model updates to the server that aggregates all the information to update the

shared model. A naive aggregation rule is to average the local model parameters as the global model parameters.

Federated learning avoids centrally storing the data. Instead, it allows users to train a shared model collaboratively by using its local data. Therefore, it guarantees data privacy and decreases the communication cost. In general, the learning task is achieved by a federation of participants (clients) supervised by a central coordinator (server). Since the clients never upload their local data to the server, the clients compute the shared model's updates, and only this update is communicated.

In federated learning, the adversary can control the whole local training dataset, local hyper-parameter of a model, and their local model. The studies show that using Mean as the aggregation method fails in the presence of malicious clients that can compromise the entire performance and the convergence of the shared model [6, 25]. The potential attack strategies that can be exploited in federated learning categorize into three parts [57]: 1) data poisoning attack, 2) model targeted poisoning attack, and 3) freerider attacks.

Label flipping attacks, byzantine attacks, and noisy data attacks fall into the data poisoning attacks. The model poisoning attack [6, 24] aims to poison the global model directly, and adversary attempts to substitute the global model with their target model. In freeriding attacks [59], the adversary takes advantage of the system without contributing enough to the learning process.

Fung et al. [25] proposed a defense method, called FoolsGold, against data poisoning attack in federated learning. They assume that the non-iid follows data in the federated learning system. They can detect the ordinary workers from adversary one by calculating the similarity of their updated gradients.

2.2.2 Adversarial Attacks on Federated Learning

In federated learning, the adversary can control the whole local training dataset, local hyper-parameter of a model, and local model parameters. The potential attacks that can be exploited in federated learning categorize into three parts: 1) data poisoning

attack, 2) model poisoning attack, and 3) freerider attacks.

In the data poisoning attack [25], the malicious workers create poisoned training samples and inject them into their local training dataset. Then, the local model is trained on the dataset contaminated with such poisoned samples. The purpose of this attack is to manipulate the global model to misclassified the true class. These attacks can be further folded into two categories: 1) label-flipping attacks [25] and 2) noisy features attack [25]. The label-flipping attack occurs where the labels of training examples of one class are flipped to another class while the data features remain unchanged. For example, an attacker can train a local model with cat images misclassified as a dog and then share the poisoned local model for aggregation. A successful attack forces a model to incorrectly predicts cats to be dogs. However, in the noisy features attacks, the adversary adds noise to the features while the class label of each data point intact [25], noisy data, and the backdoor attacks fall in this type of attack.

The model poisoning attack [6,24] aims to poison the global model directly, and adversary attempts to substitute the global model with their target model. To achieve this goal, the problem is formulated as an optimization problem and the loss function is changed so that the adversary achieves their goal.

The federated learning is vulnerable to poisoning attacks. Studies show [6,25] that just one or two adversarial clients are enough to compromise the performance of the global model. Hence, developing a robust method against these attacks is essential. Fung et al. [25] proposed a defense method, called FoolsGold, against data poisoning attack in federated learning in a non-IID setting. Their solution differentiates the benign clients from the adversary one by calculating the similarity of their submitted gradients. Other techniques use recursive Bayes filtering method [69] in the IID setting to mitigate the data poisoning attack.

In some studies, researchers assume that the global server accesses the golden data that represent data distribution from clients. However, this assumption in practice is hard to be true. By assuming access to the golden dataset, the server can detect

adversary and prevent attacks by assessing the effectiveness of provided updates on the global model's performance. If the updates do not improve the global model's performance, the client is flagged as a potential adversary [6]. However, this method needs to have a golden validation dataset which would be difficult to achieve in practice.

2.2.3 Byzantine-Robust Federated Learning

Byzantine worker's goal is to ensure that the global model does not converge or convergence to the sub-optimum model. The byzantine workers add Gaussian noise to the gradient estimators and then send this perturb value to the server. Therefore, the byzantine gradients can have similar variance and magnitude as the correct gradients, making them hard to distinguish.

In an adversarial setting, the naive aggregation rule, averaging the local model parameters, is not robust under adversarial settings. Byzantine-robust algorithms have been increasingly studied in recent years [8, 96]. One of the first methods is called Krum [8] which selects the local model as the global model that is similar to other models by calculating the Euclidean distance. As Krum converges slowly compared to other aggregation rules, the Multi-Krum [72] a variant of the previous algorithm that achieves similar performance at a faster convergence rate. Trimmed mean [96] aggregation rule aggregate each model parameter independently, for each jth parameter, it first sorts jth parameters of all k local models, then removes the largest and smallest β of them, and computes the mean of the remaining parameters as the jth parameter of the global model. Bulyan [32] combines Krum and a variant of the trimmed mean. The median aggregation rule, sorts the jth parameters of all k local models and takes the median as the jth parameter of the global model. Aggregation such as Krum and trimmed mean need to know the upper bound of the number of adversary clients to set parameters appropriately.

Byzantine-Robust methods have been studied in recent years [**?**, **?**, 8, 14, 23, 69, 96]. Most existing methods assume that data is distributed IID among clients, and the

proposed methods are based on statistical robustness such as median-based approach.

A popular line of robust methods against the Byzantine attack is based on the median of the updates [14]. This method sorts the local models' jth parameters and takes the median as the jth parameter for the global model. Trimmed mean [97] is another method that sorts jth parameters of all local models, removes the portion of largest and smallest of them, and computes the mean of the remaining parameters as the jth parameter of the global model. Krum [8] selects the local model with the smallest sum of distance as the global model. Aggregation method such as Krum and trimmed mean need to know the upper bound of the number of compromised workers. There are other methods that extend based on Krum such as Multi-Krum [8] and Bulyan [23].

Chapter 3

Evaluation of Truth Inference Methods under Data Poisoning Attack

This chapter proposes a comprehensive data poisoning attack taxonomy for truth inference in crowdsourcing and systematically evaluates the state-of-the-art truth inference methods under various data poisoning attacks. We use several evaluation metrics to analyze the robustness or susceptibility of truth inference methods against various attacks, which sheds light on the resilience of existing methods and ultimately builds more robust truth inference methods in an open setting.

3.1 Problem Definition

In this section, we define the truth inference problem and introduce some terminology. Table 3.1 lists notations used throughout this chapter.

Given a set of tasks $\mathbf{T}$, set of workers $\mathbf{W}$ and a bipartite graph indicating tasks assigned to each worker, a truth inference returns a set of predicted true label for tasks, denoted as $\hat{\mathbf{Z}}$. Figure 3.1 illustrates an example of a crowdsourcing system with three workers and five tasks and a bipartite task-worker assignment graph. The input of a truth inference method is an answer matrix $\mathbf{C}$ provided by the workers for tasks. Workers provide labels 0 or 1 for each task, while x reflects that the task is

Table 3.1: Notation

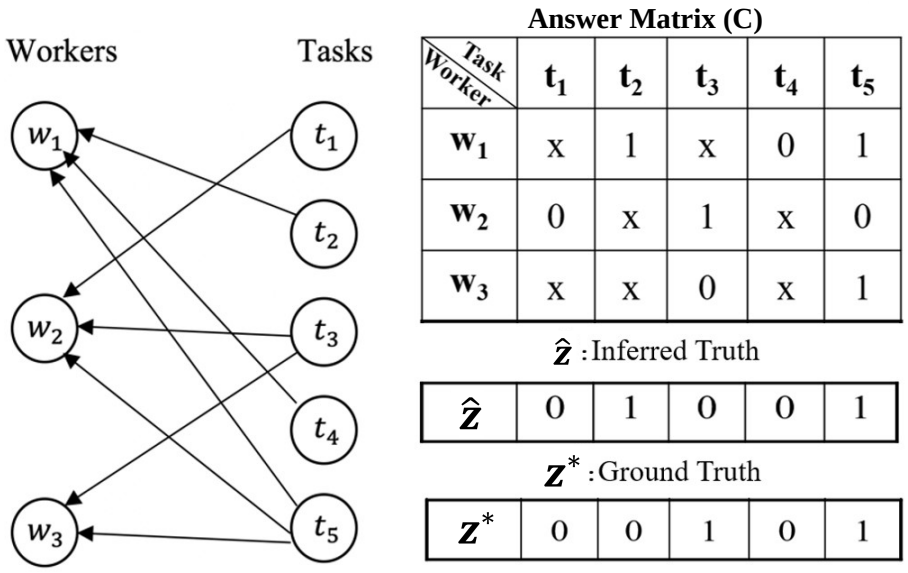
Symbol	Description
n	Number of tasks
W_j	Worker number j
T_i	Task number i
π_j	Confusion matrix of j'th worker
W'_j	Worker number j, who is malicious
π'_j	Confusion matrix of j'th worker, who is malicious
t^*	Ground truth vector
$\hat{t}$	Predicted truth vector
tpw	Average number of tasks assigned to each worker
T_{w_j}	Tasks assigned to worker j

not assigned to that worker. The output is the inferred truth vector $\hat{\mathbf{Z}}$ reflecting the inferred answer for each task. The ground truth vector $\mathbf{Z}^*$ is shown as a reference.

Due to their open nature, crowdsourcing systems are subject to *data poisoning attacks* [40, 48, 80] in which malicious workers may intentionally and strategically report incorrect answers in order to mislead the system to infer the wrong label for all or a targeted set of tasks. These strategical answers are different from unreliable behavior, which is typically non-malicious, unintentional, and non-strategic.

In this section, the goal is to present a comprehensive data poisoning attack taxonomy in crowdsourcing. We analyze the attacks along different dimensions, including attack goal (targeted vs. untargeted), adversarial knowledge (black-box vs. white-box), and attack strategy (heuristic vs. optimization based). Therefore, propose several metrics to evaluate the robustness of different methods against data poisoning attacks.

Traditional Sybil detection methods in online social networks [2,86] typically rely on additional features or metadata such as connectivity graph and IP addresses and may not be effective in crowdsourcing applications due to the limited availability of the



Worker \ Task	t_1	t_2	t_3	t_4	t_5
w_1	x	1	x	0	1
w_2	0	x	1	x	0
w_3	x	x	0	x	1

$\hat{z}$	0	1	0	0	1

z^*	0	0	1	0	1

Figure 3.1: Example of a crowdsourcing system

metadata. This chapter focuses on the truth inference algorithms that only rely on worker answers and their robustness against such attacks, which are orthogonal and complementary to Sybil detection methods when additional metadata are available.

3.2 Truth Inference Methods

Our goal is to be comprehensive and represent each category of techniques and worker model applicable to shed light on their role in the robustness of an inference method under data poisoning attacks. When possible (e.g., for optimization and PGM-based techniques), we leverage the findings from the previous study [107] by selecting the best-performing methods.

For direct computation, we include MV (majority voting) as a baseline and its enhanced version, MV-Soft [39]. In optimization based approaches, we include MV-hard [39] which employs a semi-matching optimization formulation for the worker-task bipartite graph and PM [53] which formulates the optimization problem based

Table 3.2: Selected Truth Inference Methods

Method	Technique	Worker Model
MV	Direct Computation	No Model
MV-Soft [39]	Direct Computation	Worker Penalty
MV-Hard [39]	Optimization	Worker Penalty
PM [53]	Optimization	Worker Probability
D&S [19]	Probabilistic Graphical Model	Confusion Matrix
BCC [44]	Probabilistic Graphical Model	Confusion Matrix
KOS [43]	Probabilistic Graphical Model	Worker Probability
LAA-S [98]	Neural Network	No Model

on inferred labels and worker reliability. For PGM based methods, we include the best performing D&S [19] and BCC [44] using the confusion matrix as worker reliability model and KOS [43] with a single worker reliability parameter. Among neural network based methods, we include LAA-S [98] since it is the only one applicable to a non-complete bipartite graph. Below we briefly review these selected methods.

3.2.1 Direct Computation

- Majority voting (MV): MV simply sets the label for a task to the most voted label without considering any worker reliability. We include this method as a baseline.

- MV-Soft: The majority voting with soft penalty (MV-soft) [39] implicitly models a worker's reliability via a penalty for unreliable workers. The intuition is that workers whose answers agree with the majority answer (or with more workers) are more reliable and thus assigned a lower penalty. The iterative algorithm calculates each worker's penalty in each iteration and removes up to

10 workers with the highest penalty for majority voting.

To compute the penalty, the algorithm considers tasks that do not have the same unanimous answer from all workers, i.e. conflicted tasks. It builds a bipartite graph with two set of vertices $\mathbf{W}$ and $\mathbf{T}_{conf}$, where $\mathbf{W}$ is the set of workers and $\mathbf{T}_{conf} \subseteq \mathbf{T}$ is the set of conflicted tasks. Each conflicted task is represented by two nodes, t_j^+ and t_j^-. An edge $\{w_i, t_j^+\}$ or $\{w_i, t_j^-\}$ is added to the graph if worker w_i provides label 1 or 0 to task t_j, respectively. The penalty for worker w_i is inversely proportional to the number of other workers who have the same answer as w_i, which is measured by the degree of each node in the conflicted task set $\mathbf{T}_{conf}$.

$$Pen_{w_i} = \frac{\sum_{t_j \in \mathbf{T}_{conf}^{w_i}} \frac{1}{deg(t_j^+)}.1(c_{t_j}^{w_i} = 1) + \frac{1}{deg(t_j^-)}.1(c_{t_j}^{w_i} = 0)}{|\mathbf{T}_{conf}^{w_i}|} \tag{3.1}$$

where $\mathbf{T}_{conf}^{w_i}$ is the conflicted tasks to which worker w_i has contributed, $deg(t_j^+)$ and $deg(t_j^-)$ indicate the degree of task t_j for label 1 and 0 respectively, $c_{t_j}^{w_i}$ indicates the answer provided by w_i for task t_j and 1 indicates the identifier function whose value is 1 if the condition is true and 0 otherwise.

3.2.2 Optimization Based Methods

Optimization based methods [39, 52, 53, 108, 111] use an optimization approach to obtain the optimal labels and/or the worker reliability values. Existing works differ in the formulation of their optimization problems and objectives. For our evaluation, we include PM [53] which is a representative method and has shown promising results among similar methods [52, 53, 108, 111] in [107]. We also include Majority Voting with hard penalty (MV-hard) [39] that is not included in [107] and uses a bipartite graph formulation.

- MV-Hard: While the MV-Soft method heuristically assigns a penalty to all workers contributing to a task, MV-Hard [39] utilizes the optimal semi-matching

algorithm for the tasks-workers bipartite graph to assign a penalty to workers. A semi matching is a matching subgraph of the original bipartite graph, such that exactly one worker among all contributors for each label of each task is chosen. The optimal matching is the one that minimizes the sum of accumulated degree (a typical formulation of the semi-matching problem) of all workers:

$$\min_{Match} \sum_{w \in \mathbf{W}} \sum_{i=1}^{deg_{Match}(w)} i \tag{3.2}$$

where $deg_{Match}(w)$ denotes the degree of worker w in matching $Match$.

Once the optimal semi-matching is obtained, the degree of each worker is considered as its penalty. Finally the label of each task is determined as the label connected to the worker with the lower degree or lower penalty.

- PM: The PM method [53] solves an optimization problem explicitly modeling the reliability of each worker w_i using a single probability value r^{w_i}. The intuition is that the answers of workers with higher reliability would be closer to the true label. The iterative algorithm involves two steps of updating the truth labels and workers' reliability.The optimization problem is defined as follows:

$$\min_{\mathbf{r}, \hat{\mathbf{Z}}} \sum_{w_i \in \mathbf{W}} r^{w_i} . \sum_{t_j \in \mathbf{T}_{w_i}} d(\hat{z}_{t_j}, c_{t_j}^{w_i}) \tag{3.3}$$

where $d(\hat{z}_{t_j}, c_{t_j}^{w_i})$ indicates the distance between the inferred truth of task t_j and the answer provided by worker w_i for task t_j and r^{w_i} indicates the reliability value of worker w_i. The algorithm then solves the optimization problem by an iterative method with two steps that update the truth labels and reliability of workers iteratively.

3.2.3 Probabilistic Graphical Models

Probabilistic graphical model (PGM) based methods represent the true label, the workers' answers, and workers' reliability as probabilistic variables and model their

dependence via a PGM [9, 17, 19, 20, 28, 43, 44, 61, 74, 85, 90, 92, 105, 106]. Figure 3.2 illustrates a general PGM model which is the basis of many existing works [107]. $c_{t_j}^{w_i}$ is the observed variable denoting the answer provided by worker w_i for task t_j. It depends on two hidden variables: 1) π^{w_i} showing the reliability of worker w_i, and 2) $z_{t_j}^*$ representing the ground truth of task t_j. The two boxes of M tasks and N workers indicate that there are N and M repeated variables for each task and worker respectively. Additional variables may be included such as **a** and **b** to denote the prior distribution for the worker reliability and ground truth, respectively.

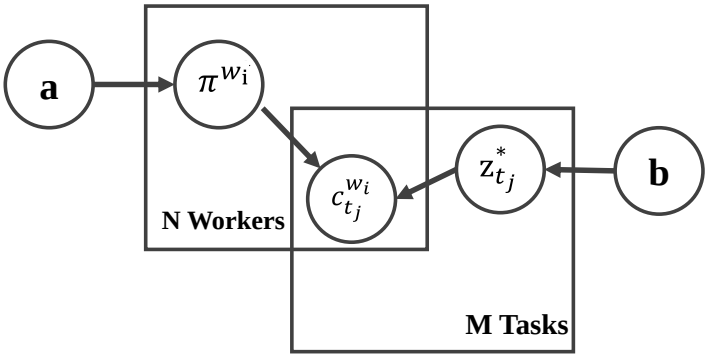


Figure 3.2: General Probabilistic Graphical Model (PGM) for Truth Inference in Crowdsourcing

Existing works differ in multiple ways, such as different worker reliability models [19, 20, 43, 44, 85], difficulty of tasks [92], domain or type of tasks [9, 61, 90], semantics information of tasks [106], models for spatiotemporal tasks [28], workers' expertise [17, 61] and bias and variance of workers for numerical tasks [74]. For our evaluation focusing on binary decision tasks, we include 1) D&S [19] and BCC [44] which use a confusion matrix worker model and are best performing in non-adversarial settings among the applicable methods [107], and 2) KOS [43] which uses a single probability to model worker reliability as a comparison.

- D&S Method: The seminal D&S (Dawid and Skene) [19] method employs a graphical model similar to Figure 3.2 without **a** and **b** priors and models worker

reliability using a confusion matrix.

Given the graphical model, the D&S [19] algorithm uses EM (Expectation-Maximization) to iteratively find the maximum likelihood estimation (MLE) for the inferred truth labels $\hat{\mathbf{Z}}$ and the confusion matrices $\pi^{\mathbf{W}}$ iteratively. $\mathbf{W}^{t_j}$ is the workers responding to task t_j. The objective function of this method is:

$$\max_{\hat{\mathbf{Z}},\pi^{\mathbf{W}}} \prod_{j=1}^{N} \sum_{l \in \mathbf{L}} pr(\hat{z}_{t_j} = l) \prod_{w_i \in \mathbf{W}^{t_j}} \pi_{l,c_{t_j}^{w_i}}^{w_i} \tag{3.4}$$

- BCC: The Bayesian Classifier Combination (BCC) algorithm [44] uses a graphical model similar to Figure 3.2 with $\mathbf{a}$ and $\mathbf{b}$ priors and finds the maximum a posteriori (MAP) estimation for the unknown parameters by optimizing the joint posterior probability. For easy tasks, a single shared confusion matrix is used for all workers, reducing the computations. Whereas for hard tasks, each worker will have its own separate confusion matrix. We can formally formulate the inference problem as:

$$\max_{\mathbf{a},\mathbf{b},\hat{\mathbf{Z}},\pi^{\mathbf{W}}} \prod_{j=1}^{N} pr(\hat{\mathbf{Z}} \mid \mathbf{a}) \prod_{i=1}^{M} pr(\pi^{w_i} \mid \mathbf{b}) \prod_{i=1}^{M} pr(c_{t_j}^{w_i} \mid \pi^{w_i}, \hat{\mathbf{Z}}) \tag{3.5}$$

  There are extensions of the BCC method that take into account the difficulty of tasks and workers collaboration, but we do not consider them in this study.

- KOS: The KOS (Karger, Oh, Shah) method [43] models worker reliability using a single parameter instead of a confusion matrix. For estimating worker reliability, it attempts to maximize the joint probability distribution.

$$\max_{\hat{\mathbf{Z}},\pi^{\mathbf{W}}} \quad \prod_{j=1}^{N} \sum_{l \in \mathbf{L}} pr(\hat{z}_{t_j} = l) \prod_{w_i \in \mathbf{W}^{t_j}} \pi_{l,c_{t_j}^{w_i}}^{w_i} \tag{3.6}$$

  Since direct estimation of the distribution is intractable, an iterative belief propagation is used to estimate a distribution for the worker reliability. The label of tasks is determined based on the weighted product of estimated worker's reliability and their answers.

3.2.4 Neural Networks and Other Methods

Neural networks have also been proposed for truth inference in crowdsourcing [29,98]. Gaunt et al. [29] utilizes a neural network for aggregating labels trained on inferred labels from majority voting as the ground truth. One significant disadvantage is that most methods assume a complete graph where all tasks are assigned to all workers. In our study, we include the LAA-S method [98] which is also suitable for a sparse graph.

LAA-S: The basic idea is to represent each task as a vector consisting of answers provided by all workers and then train a neural network to learn its truth label as a latent feature or representation. Figure 3.3 shows LAA-S network architecture, which contains two shallow neural networks: 1) an encoder or classifier (q_θ) which encodes the task vector (v) into the latent feature (z) indicating the truth label, and 2) a decoder or reconstructor (p_ϕ) which reconstructs a task vector based on the latent feature. Note that this is an unsupervised approach, the intuition is to learn the latent truth label that can best represent the original task vector.

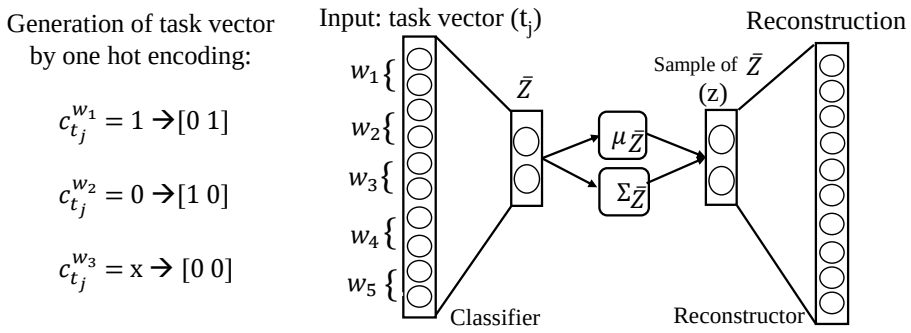


Figure 3.3: Architecture of LAA-S Method

The network is trained on all task vectors by minimizing the reconstruction error between the original and reconstructed task vectors. To create each task vector, one-hot encoding is applied to each worker's answer, as illustrated in Figure 3.3. In addition, the training also enforces the latent truth label distribution to resemble the original answer distribution. The objective function is shown below.

$$\min_{q_\theta, p_\theta} \mathbb{E}_{q_\theta(z|v)} log p_\phi(v|z) - D_{KL}(q_\theta(z|v)||\ prior) \tag{3.7}$$

where the first term is the reconstruction error and the second term ensures the distribution of inferred labels follows a specific prior using the negative KL divergence D_{KL}, and the prior is calculated based on the fraction of labels in the workers' answers for each task.

3.3 Proposed Approach

We design heuristic and optimization based attacks that can be used on various truth inference methods as part of our evaluation methodology. The heuristic-based attacks assume black-box or no adversarial knowledge and model the worker behavior using a confusion matrix [19] and an additional disguise parameter to hide their malicious behavior. The optimization based attacks assume white-box or full adversarial knowledge including the truth inference methods being used and other workers' answers, and are adapted from existing optimization based attacks [65] while making them more generic, so they apply to broader types of truth inference methods.

3.3.1 Attack Methods

Heuristics Based Attacks (HeurAtt). We design heuristics based attacks that are applicable for a black-box setting following non-collusive strategy. The simplest heuristic for an attacker would be to always report the wrong answer for each of their assigned tasks. However, this may be easily recognized by most of the truth inference systems (besides majority voting) which model the workers' reliability. The attackers could disguise themselves as honest workers by providing true answers for some tasks so that they won't be detected by the system. The worker's behavior is modeled by defining a confusion matrix π^w that captures a worker's probability of providing a certain label given the true label. Given the label set $L = \{0, 1\}$, α and β indicate the probability of workers provide a correct label given the true label

of 1 and 0, respectively. Each malicious worker w' is associated with a malicious confusion matrix $\pi^{w'}$ with α' and β' and a normal confusion matrix π^{w} with α and β, and a disguise parameter γ. w' behaves as a normal worker modeled by π^{w} with probability γ and as a malicious worker modeled by $\pi^{w'}$ with probability $1-\gamma$. For example, a malicious worker with a moderate level of disguise may be modeled with $\pi^{w} = \begin{bmatrix} \beta=1 & 0 \\ 0 & \alpha=1 \end{bmatrix}$, $\pi^{w'} = \begin{bmatrix} \beta'=0 & 1 \\ 1 & \alpha'=0 \end{bmatrix}$, and $\gamma = 0.2$.

For targeted attacks, the best strategy for the malicious workers is to flip the labels for the targeted tasks while acting truthfully for other tasks, escaping detection and building their reliability. Hence the disguise parameter (γ) is set to be 0.

Optimization Based Attacks (OptAtt). We adopt the optimization based attack in a white box setting where an adversary has the full knowledge of truth inference algorithm, provided answers to the system, and task assignments and optimally injects manipulated answers to maximize the damage to the system. The attack goal is to maximize the number of flipped labels before and after the attack along with maximizing the attackers' collective confusion matrix parameters inferred by the system. Intuitively, this will help them to obfuscate their malicious nature and cause more disruption in the system.

Let $\hat{z^a_{t_j}}$ and $\hat{z^b_{t_j}}$ denote the inferred answer for task t_j by the D&S after and before attack respectively. $\hat{\alpha}_{w'}$ and $\hat{\beta}_{w'}$ show the inferred confusion matrix parameters of the malicious worker w'. The optimization is posed as Equation (4.1) where λ controls the trade-off between the objectives of maximizing the collective reliability of malicious workers and the number of flipped labels.

$$\max_{\mathbf{C}'} \sum_{j=1}^{M} 1(\hat{z^a_{t_j}} \neq \hat{z^b_{t_j}}) + \lambda \sum_{w' \in \mathbf{W}'} (\hat{\alpha}_{w'} + \hat{\beta}_{w'}) \tag{3.8}$$

For untargeted attack, we extend the attack from [65] to all inference methods. For confusion matrix based ones, we use the same formulation as equation (4.1). For methods with a single reliability parameter or no worker model, we set $\lambda = 0$.

For targeted attack, the aim is to maximize the number of targeted tasks $\mathbf{T}_{tar} \subseteq$

$\mathbf{T}$ whose label is flipped. The optimization is $\max_{\mathbf{C}'} \& \sum_{t_j \in \mathbf{T}_{tar}} 1(\hat{z^a_{t_j}} \neq \hat{z^b_{t_j}}) + \lambda \sum_{w' \in \mathbf{W}'} (\hat{\alpha}_{w'} + \hat{\beta}_{w'})$. Similar to untargeted attacks, the second term is considered only for confusion matrix based methods. We varied λ in the interval [0,1] and observed that a λ value in [0.9, 1] leads to the most successful attack.

3.3.2 Selected Truth Inference Methods

We systematically evaluate the state-of-the-art truth inference methods under both heuristics and optimization based data poisoning attacks. The truth inference methods are selected carefully to represent the different types of methods including majority voting based [39], optimization based [53], probabilistic graphical model based [19,43,44], and neural network based [98]. They also portray different worker's behavior models including probability based [53], confusion matrix based [19, 43, 44], and implicit models [39, 53]. Our study includes not only the best performing methods from the experimental study [107], but also additional direct computation [39] and optimization based methods [39, 53] and more recent neural network based methods [98]. A summary of the selected methods with their key features is given in Table 3.2.

3.3.3 Metrics

We use the following metrics to assess the robustness of inference methods.

Accuracy Accuracy is the fraction of correctly inferred tasks, i.e $\sum_{j=1}^{N} 1(\hat{z_{t_j}} = z^*_{t_j})/N$, where $\hat{z_{t_j}}$ and $z^*_{t_j}$ are inferred and ground truth of the jth task. A lower accuracy means a more successful attack. We note that [65] defined the attack success metric as the percentage of inferred labels flipped due to attack, comparing labels before and after the attack. We believe this metric does not truly capture the attackers' success where some inferred labels may be wrong without attack and were flipped to correct due to the attack, i.e., adversaries help the system to correctly infer the

label of an otherwise wrongly labeled task. Instead, we use the flipped labels w.r.t. the ground truth, i.e. accuracy, as the metric for attack success. We also report F1 score to account for the skewness of classes in the unbalanced dataset.

Accuracy_Targeted *Accuracy_Targeted* is the fraction of the targeted tasks $\mathbf{T}_{tar}$ whose truth are inferred correctly, i.e
$1 - \sum_{t_j \in \mathbf{T}_{tar}} 1(\hat{z_{t_j}} \neq z^*_{t_j})/|\mathbf{T}_{tar}|$.

Area Under Curve (AUC) Since the inference methods' accuracy changes over parameters, e.g. percentage of malicious workers, we use AUC to compare the global performance of methods on an interval of parameter values, if feasible.

Recognizability To assess the adversary detection ability of inference methods with explicit worker models, we define *Recognizability* as the similarity between the simulated (ground truth) worker reliability and the inferred reliability. A higher *Recognizability* means the method is better at detecting malicious workers. The worker behavior is modeled by a normal confusion matrix with α and β, a malicious one with α' and β', and a disguise parameter γ. We aggregate these into a single value $r_{w'}$ showing the expected reliability of a worker and define *Recognizability* as $1-\frac{1}{|\mathbf{W}'|}\sum_{w' \in \mathbf{W}'} |r_{w'} - \hat{r}_{w'}|$, where $r_{w'}$ and $\hat{r}_{w'}$ are the simulated and inferred reliability of malicious worker w' respectively.

$$
\begin{aligned}
r_{w'} = \frac{1}{\mid \mathbf{T}_{w'} \mid} \sum_{t_j \in \mathbf{T}_{w'}} & (\alpha_{w'} \times \gamma + \alpha'_{w'} \times (1-\gamma)) \times 1(z^*_{t_j} = 1) \\
& + (\beta_{w'} \times \gamma + \beta'_{w'} \times (1-\gamma)) \times 1(z^*_{t_j} = 0)
\end{aligned}
\tag{3.9}
$$

$$
\hat{\mathrm{r}}_{w'} = \frac{1}{|\mathbf{T}_{w'}|} \sum_{t_j \in \mathbf{T}_{w'}} \hat{\alpha_{w'}} \times 1(z^*_{t_j} = 1) + \hat{\beta_{w'}} \times 1(z^*_{t_j} = 0) \tag{3.10}
$$

3.4 Evaluation Results

In this section, we report the robustness of various truth inference methods under heuristic-based (HeurAtt) and optimization-based (OptAtt) attacks.

Datasets

For our evaluation, we used two benchmark datasets for decision making tasks [5,38] as well as synthetic datasets constructed based on different parameters. A summary of the datasets and their key properties are shown in Table 3.3.

Table 3.3: Statistics and Properties of Datasets

Dataset	**Product**	**PosSent**	**Synthetic**
N (# of tasks)	8,315	1,000	[200, 40,000]
M (# of workers)	176	85	[100, 500]
V (# of answers)	24,945	20,000	[10,000, 200,000]
Redundancy (# of answers per task)	3	20	[5, 30]
Engagement (# of answers per worker)	141	235	[100, 400]
Skewness	0.88	0.52	[0.5 0.9]

Product Dataset In this dataset, each task is comparison of two products. An example is "are *iPad Two 16GB WiFi White* and *iPad 2nd generation 16GB WiFi White* the same?" [38]. Workers either assign label *T*, meaning the two products are the same, or *F* otherwise. The Product dataset is unbalanced for the positive and negative tasks with 88% of tasks having a negative label and 12% positive. The dataset is sparse since the average number of answers per task is 3. The average workers' credibility is around 0.79 according to [107].

PosSent Dataset In this dataset, workers assess the general sentiment of a tweet about the reputation of a company. An example tweet is "The recent products of Apple is amazing" [107]. The response is either positive, meaning that the tweet will increase the reputation of the company or negative [5]. PosSent dataset is more

balanced with 52.8% of tasks being negative and 47.2% positive. It is also less sparse than Product dataset in that the average number of answers per task is 20. Similar to Product dataset, the average workers' credibility falls in the range of 0.79 [107].

Synthetic Dataset To analyze datasets with different properties, we generate synthetic datasets and change parameters such as redundancy, number of tasks, and number of workers as shown in Table 3.3. The worker-task assignment graph is generated from Power-law distribution. The ground truth for each task is independently drawn from a Bernoulli distribution with prior 0.5, resulting in a balanced dataset. We also experimented with alternative distributions and observed no significant difference in results and hence 0.5 was chosen as the default value. The workers provide answers according to a confusion matrix with α and β drawn from a Beta distribution.

3.4.1 Heuristic-Based Attacks (HeurAtt): Untargeted

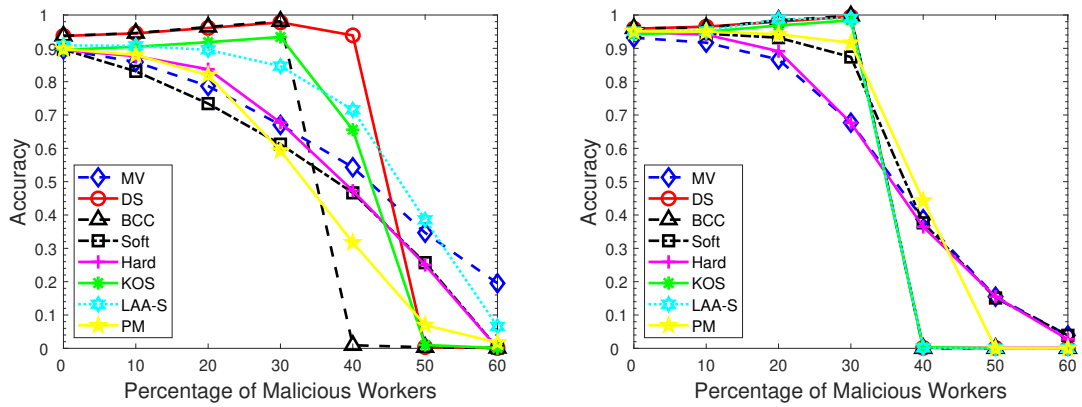
In untargeted attacks, the goal is decreasing the overall accuracy of the system.

Impact of Percentage of Malicious Workers Here the number of normal workers is fixed and the percentage of added malicious workers varies from 0 to 60%. Adversary behavior setting is $\gamma = 0$, $\alpha' = 0$ and $\beta' = 0$.

Table 3.4: AUC of Methods' Accuracy w.r.t. % of Malicious Workers: Untargeted HeurAtt

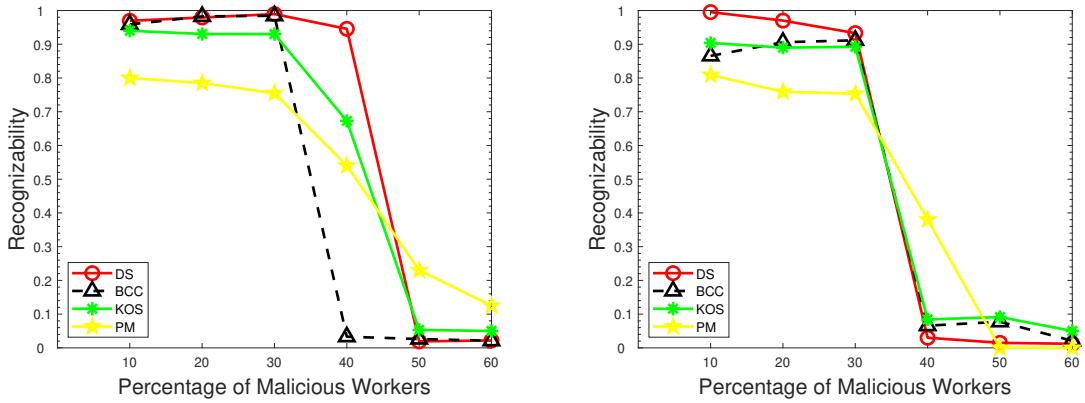
	MV	**DS**	**BCC**	**Soft**	**Hard**	**KOS**	**LAAS**	**PM**
Product	34.78	**42.9**	33.69	32.2	34.3	38.5	**40.09**	30.92
PosSent	32.88	34.01	34.2	34.99	34.215	33.6	34.062	34.12

Accuracy Figure 3.4 shows the accuracy of the methods w.r.t. the percentage of malicious workers. We omitted the result for synthetic dataset showing similar trends. Increasing the number of malicious workers drops the accuracy of all methods. The direct computation (MV, MV-Soft, MV-Hard) and neural network method's



(a) Product (b) PosSent

Figure 3.4: Untargeted HeurAtt:
Accuracy vs. % of Malicious Workers



(a) Product (b) PosSent

Figure 3.5: Untargeted HeurAtt:
Recognizability vs. % of Malicious Workers

(LAA-S) drop is almost linear early on, PGM methods (D&S, BCC, and KOS) and probabilistic method (PM) are more resistant especially with few adversaries and drop to 0 once the percentage goes beyond (40% to 50%). Overall, D&S and LAA-S

are the most resilient for this attack. Comparing the datasets, Product dataset is more susceptible to the attack due to its low redundancy. Table 3.4 shows the AUC of methods for Product dataset that confirms the same patterns for the relative global performance of methods. Since the AUC of PosSent dataset for all methods is around 34, there is no clear winner among them.

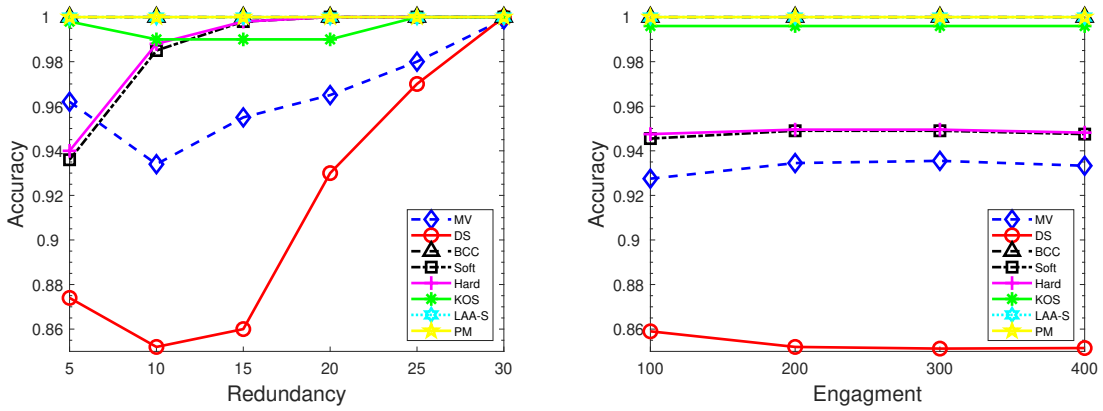
Recognizability To show adversary detectability in inference methods, we report recognizability of methods with explicit worker modeling, i.e. D&S, BCC, PM, and KOS. We exclude MV and LAA-S as they do not explicitly model reliability. MV-Hard and MV-Soft are excluded too since recognizability is over the average adversaries' reliability and they only remove the least credible worker.

Figure 3.5 shows the methods' recognizability w.r.t. varying percentages of malicious workers. We omitted the result for synthetic dataset which was similar. D&S and KOS perform better than BCC in adversary detection, while PM performs the worst. This explains the robustness of the accuracy of D&S and KOS we observed earlier. Comparing Figure 3.4 and 3.5, the accuracy and recognizability of D&S and KOS decrease as the percentage of malicious workers increases, i.e. worker modeling with good detection is the key to a robust inference algorithm under attack.

Impact of Redundancy and Engagement Redundancy is the mean number of workers assigned per task, while worker engagement is the mean number of tasks per worker. Figure 3.6 shows the accuracy w.r.t. varying redundancy and engagement values. As expected, with increased redundancy, it is harder for adversaries to reduce the accuracy (the percentage of attackers is set to 20%). D&S and MV-based models were more sensitive to redundancy in the sparse dataset (Product). Worker engagement had no significant impact, since it does not directly impact their reliability.

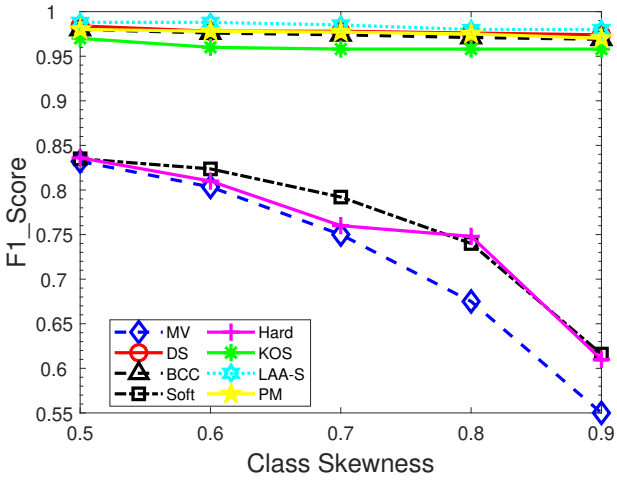
Impact of Class Skewness We show the effect of skewness on F1-score in synthetic data. Figure 3.7a shows the F1-score w.r.t. varying ratios of the majority class. MV-based methods are vulnerable to imbalance while others are robust.

Impact of Disguise (γ) We show the trend of accuracy and recognizability w.r.t disguise. When in disguise, adversaries' behavior is governed by $\alpha = 1$ and $\beta = 1$



(a) Redundancy (b) Engagement

Figure 3.6: Untargeted HeurAtt: Accuracy vs Redundancy and Engagement (Synthetic)

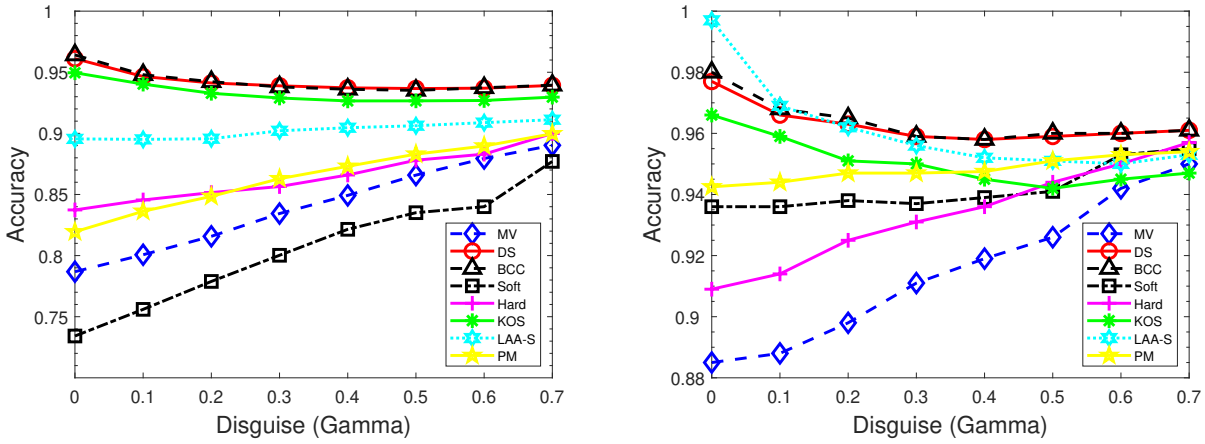


(a) Skewness

Figure 3.7: Untargeted HeurAtt: F1 Score vs Class Skewness

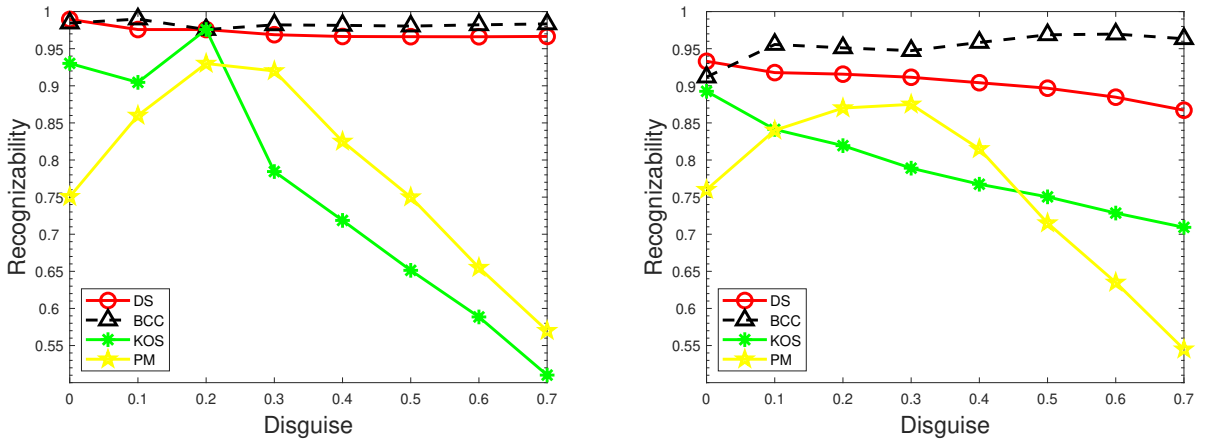
compared to $\alpha' = 0$ and $\beta' = 0$ in pure malicious mode.

Accuracy Figure 3.8 shows the accuracy of methods w.r.t. varying disguise levels. We use a different scale for each dataset's y-axis to highlight their trend. Since



(a) Product (b) PosSent

Figure 3.8: Untargeted HeurAtt: Accuracy vs Disguise



(a) Product (b) PosSent

Figure 3.9: Untargeted HeurAtt: Recognizability vs. Disguise (γ)

increasing disguise after 0.7 resulted in monotonously increasing accuracy for all models, we terminate at 0.7. For methods (e.g. MV) with no inherent attacker recognition, disguising only boosts the accuracy. For more robust methods, as we increase γ slightly, the algorithms fail to identify adversaries leading to the success

of the attack. However, as disguise further increases, the accuracy goes back up due to the correct answers by the disguised malicious workers. Hence there's an optimal level of disguise for attackers.

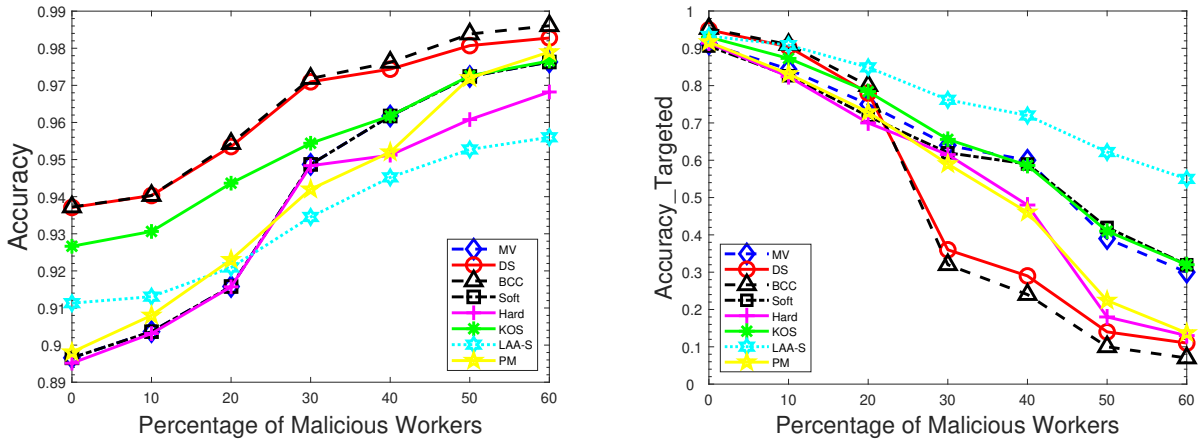
Recognizability. Figure 3.9 shows the recognizability of methods w.r.t. varying disguise levels on real datasets. Confusion matrix based models, BCC and D&S, are more robust to disguise and model the workers accurately regardless of adversarys' disguise. KOS uses a single reliability value and thus is more sensitive to disguise. Generally, the recognizability of adversaries drops as disguise increases, since their behavior more closely resembles normal workers.

3.4.2 Heuristic-Based Attacks (HeurAtt): Targeted

We report the evaluation results for targeted attacks. We focus on parameters relevant to targeted attacks, the percentage of malicious workers and the proportion of targeted tasks.

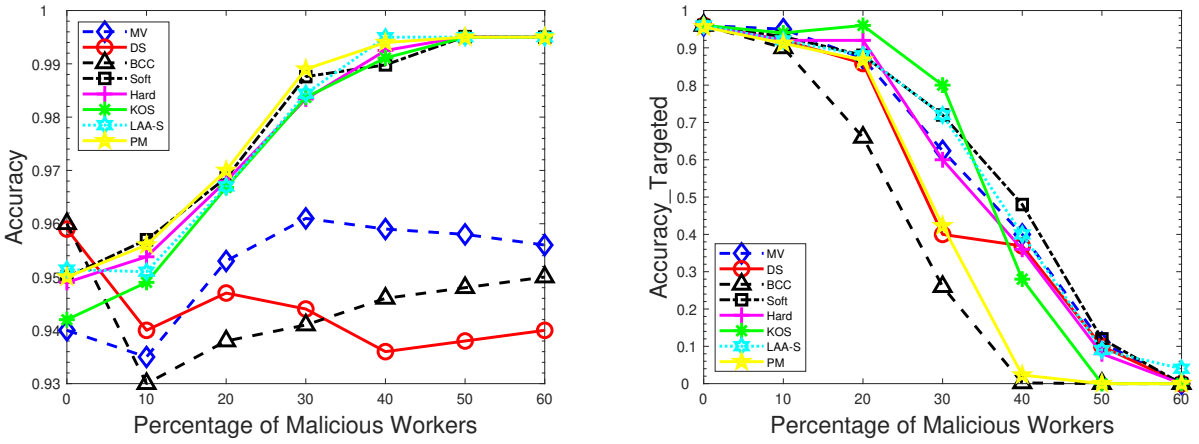
Impact of Percentage of Malicious Workers Figure 3.10 and 3.11 show the accuracy of the methods w.r.t. varying percentage of malicious workers on the real datasets. We fixed the fraction of targeted tasks for Product and PosSent dataset to be 0.2 and 0.1, respectively, based on two factors: 1) targeted attack is impactful, 2) there is an observable difference among methods' performance. The general trend is that increasing the number of attackers, the overall accuracy of the system increases thanks to the truthful contributions of attackers to non-targeted tasks. However, the accuracy of targeted tasks is decreased by attackers.

Remarkably, D&S and BCC which are more robust against untargeted attacks are more susceptible to targeted attacks. While they maintain high overall accuracy, their accuracy for the targeted tasks suffers the most due to their failure to differentiate targeted and untargeted tasks when modeling workers' behavior (i.e. being misled by malicious workers based on their true answers to the untargeted tasks). On the other hand, LAA-S is significantly more robust against targeted attacks, even though the overall accuracy is not as high as other methods, explained by the



(a) Accuracy (b) Accuracy_Targeted

Figure 3.10: Targeted HeurAtt: Accuracy & Accuracy_Targeted vs. % of Malicious Workers (Product dataset)



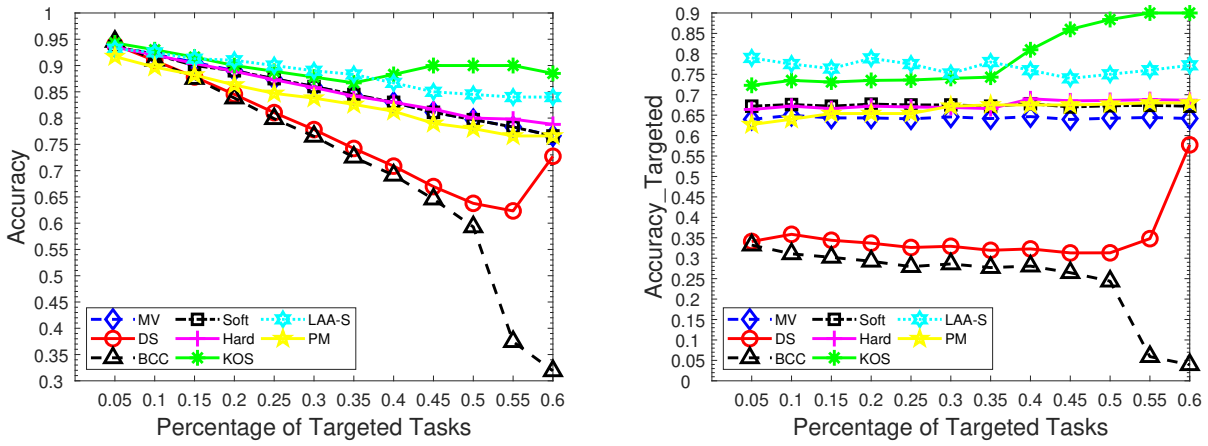
(a) Accuracy (b) Accuracy_Targeted

Figure 3.11: Targeted HeurAtt: Accuracy & Accuracy_Targeted vs. % of Malicious Workers (PosSent dataset)

absence of explicit worker modeling. While MV-Soft performs worse than others in untargeted attacks (Figure 3.4), it is the most resilient alongside LAA-S for targeted

attacks. MV-Soft's resilience is due to accurate detection and penalization of malicious workers when they are the majority contributing to a task with conflicting answers, that happen more frequently while focusing on limited tasks rather than all tasks.

Impact of Proportion of Targeted Tasks Figure 3.12 and 3.13 show the accuracy of the methods w.r.t. varying percentage of targeted tasks on real datasets. As the ratio of targeted tasks increases, the overall accuracy and targeted accuracy decrease. However, when the ratio gets sufficiently large, accuracy increase for D&S and BCC. Since malicious workers have to dilute their efforts among a larger set of targeted tasks, they are more discoverable and less effective.

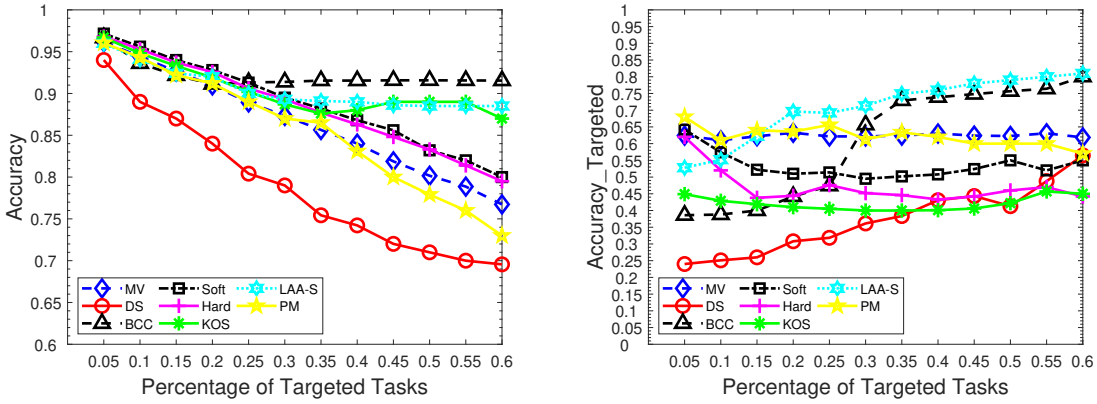


(a) Accuracy

(b) Accuracy_Targeted

Figure 3.12: Targeted HeurAtt: Accuracy vs. Ratio of Targeted Tasks (Product dataset)

Comparing the datasets, answer redundancy inversely affect targeted attack's success, similar to untargeted attacks. Given the Product dataset's lower redundancy, this attack is successful even with a high ratio of targeted tasks.



(a) Accuracy

(b) Accuracy_Targeted

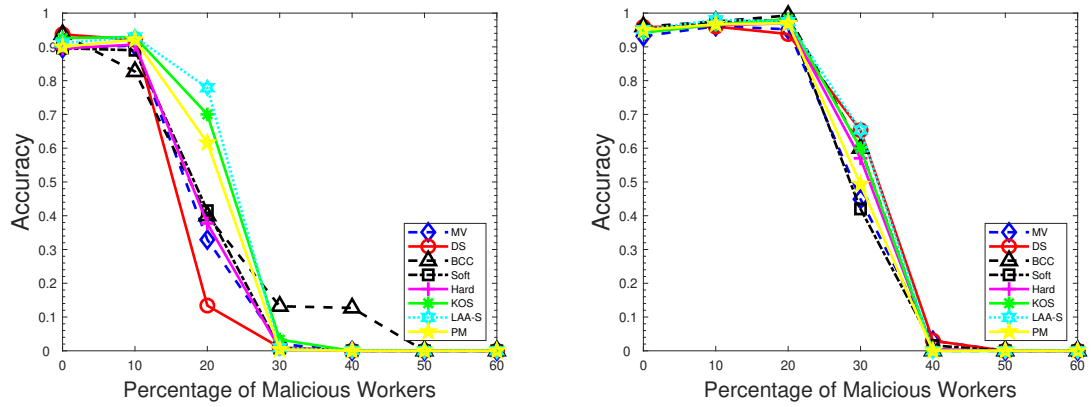
Figure 3.13: Targeted HeurAtt:Accuracy vs. Ratio of Targeted Tasks (PosSent dataset)

Optimization Based Attacks (OptAtt): Untargeted

Impact of Percentage of Malicious Workers (White-Box Attack) We evaluate the OptAtt in white-box setting, i.e. given the full knowledge of the inference method used and all other workers' answers. Figure 3.14 shows the accuracy of the inference methods w.r.t. varying percentage of malicous workers.

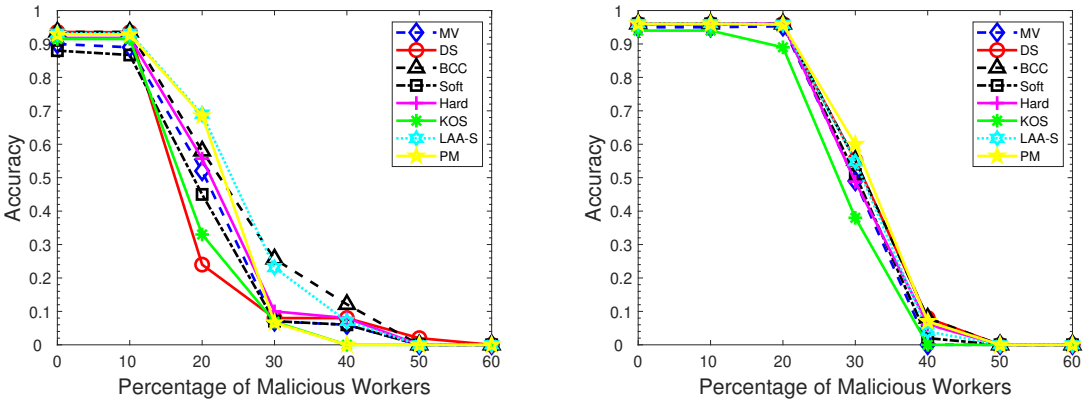
All methods' accuracy drops fairly quickly as the percentage of malicious workers increases. Comparing HeurAtt and OptAtt (Figure 3.4a vs Figure 3.14a for Product dataset, Figure 3.4b vs Figure 3.14b for PosSent dataset), the accuracy under OptAtt drops to zero at a much lower percentage of malicious workers for all methods. The attackers are indeed more successful when using the optimized scheme through stronger adversarial knowledge. Comparing methods, all perform similarly in resiliency and are susceptible to the attack, since it is optimized for that particular inference method. However, LAA-S has a slight edge over others.

Transferability of Inference Methods (Gray-Box Attack) We evaluate the effectivenes of OptAtt when attackers only know other's answers but not the inference method used. We analyze the transferability of attack when the inference method



(a) Product (b) PosSent

Figure 3.14: Untargeted OptAtt: Accuracy vs. % of Malicious Workers



(a) Product (b) PosSent

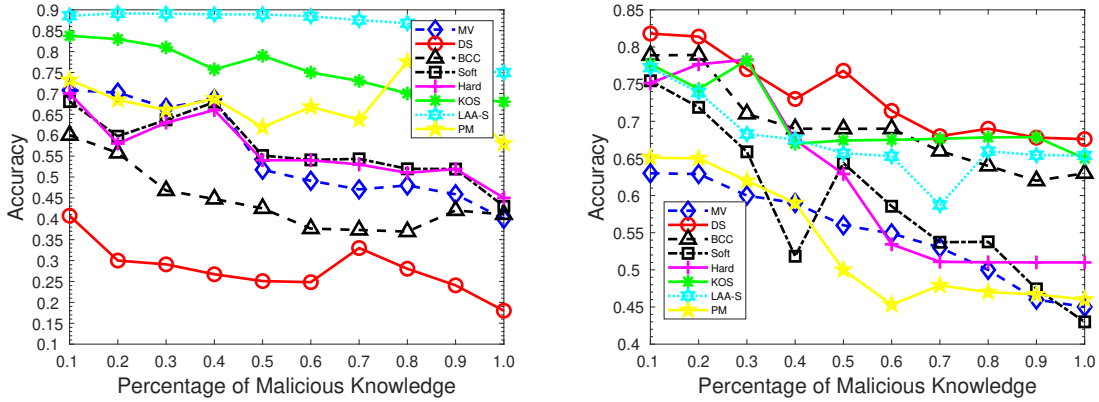
Figure 3.15: Untargeted OptAtt Transferability (Gray-Box): Accuracy vs % of Malicious Workers

assumed by the attack is not the same method used in reality.

Figure 3.15 shows the accuracy of the methods w.r.t. varying percentage of adversaries assuming the common D&S is used as the inference method. Regardless of the inference method used in reality, the attack is as successful as the white-box attack,

i.e. a general attack based on D&S is transferable to others.

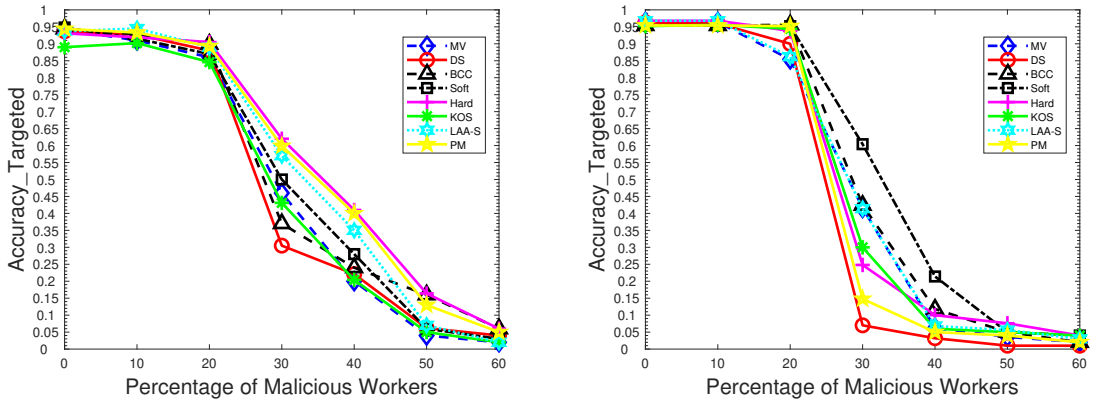
Impact of Adversarial Knowledge (Gray-Box Attack) We analyze OptAtt in gray-box settings when attackers only know part of others' answers plus the inference method used. Attackers infer labels before and after attack using partial answers. The malicious workers are set to 20% and 30% for Product and PosSent datasets respectively. Figure 3.16 shows the accuracy of the methods w.r.t. varying percentage of malicious knowledge of others' answers.



(a) Product (b) PosSent

Figure 3.16: Untargeted OptAtt (Gray-Box):
Accuracy vs Partial Knowledge

The accuracy drops and attack is more successful with more adversarial knowledge. The attack is still quite successful even when a very low percentage of other's answers are known. One explanation is that the reliability of workers for both datasets is quite uniformly distributed around 0.79. Thus for all methods, even with a small fraction of available normal workers' answers (i.e. 0.2), the adversaries estimate the truth quite accurately. OptAtt in gray-box setting is quite realistic and effective disproving that full adversarial knowledge is needed for success.



(a) Product (b) PosSent

Figure 3.17: Targeted OptAtt: Accuracy_Targeted vs. % of Malicious Workers on Product and PosSent dataset

3.4.3 Optimization Based Attacks (OptAtt): Targeted

Impact of Percentage of Malicious Workers We set the ratio of targeted tasks to be 0.01 and 0.005 for Product and PosSent datasets respectively. Since OptAtt is more successful, a lower ratio of targeted tasks is chosen here compared to HuerAtt to portray the same regions of interest for methods' performance. Figure 3.17 shows accuracy_targeted of the inference methods w.r.t. varying percentage of malicious workers on the real datasets. Overall accuracy is not shown due to space limitations. *Accuracy_Targeted* decreases as percentage of malicious workers increases. Comparing targeted attack in HeurAtt and OptAtt (Figures 3.10 & 3.11), HeurAtt is more effective in reducing accuracy at a small percentage of malicious workers. However, with a greater percentage of malicious workers, targeted OptAtt attack is more successful. One probable reason is since a subset of tasks is targeted, the chance of adversary's detection is lower compared to the untargeted setting. So, OptAtt that trades some accuracy drop in exchange for less detection will lose its edge in attack power over HeurAtt which only focuses on accuracy.

Discussion

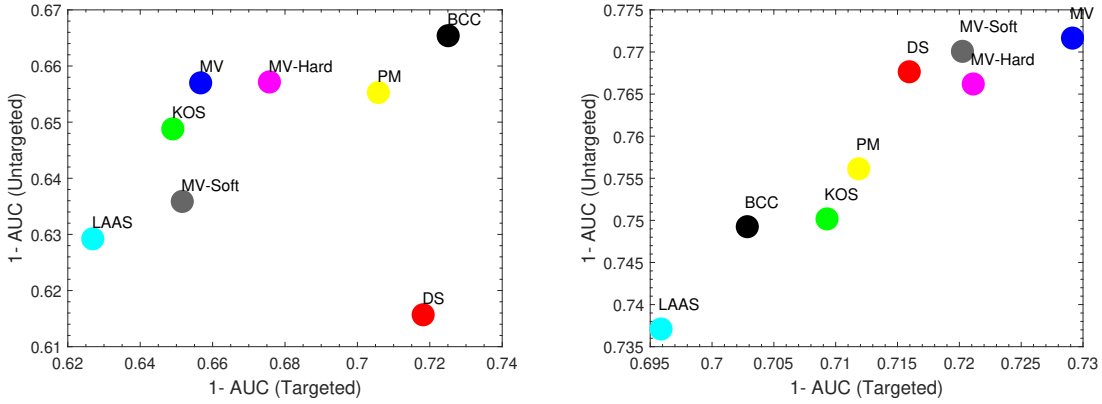
We summarize our key findings on the performance of leading inference methods using diverse techniques under various data poisoning attacks.

Table 3.5: Top 2 Robust Methods under Different Attacks

GoalStrategy	**HeurAtt**	**OptAtt**
Untargeted	D&S (LAA-S)	LAA-S (KOS)
Targeted	LAA-S (MV-Soft)	LAA-S (BCC)

Comparison of Methods Figure 3.18 shows the overall attack susceptibility of different inference methods along two dimensions (untargeted attacks and targeted attacks) under HeurAtt and OptAtt attacks. Susceptibility is defined as 1 - AUC. A more robust method should have a lower susceptibility across both dimensions. The AUC is the accuracy over an interval of fraction of attackers. Since in reality malicious workers are not the majority, we choose the interval [0, 0.5]. The most robust methods should be those dominating others (less vulnerable) in both dimensions, the pareto optimal methods or skyline. Table 3.5 shows the top 2 performing methods for each category of attacks. We also discuss the main findings below.

- Among direct computation methods, MV-Soft is more robust than MV, i.e. dominates MV, for all attacks, thanks to its modeling of worker reliability.
- Among OptAtt, MV-Hard is more robust than PM only under targeted HeurAtt. This can be attributed to its optimal matching algorithm that penalizes or removes malicious workers when they become the majority among the contributing workers for conflicting tasks, which happens when they all give wrong answers to a target set of tasks. On the other hand, PM is more robust than MV-Hard under other attacks.
- For untargeted attacks, among PGM based methods, D&S dominate BCC under HeurAtt. However, under OptAtt, BCC dominates D&S. Note that in



(a) HeurAtt (b) OptAtt

Figure 3.18: Susceptibility of Different Inference Methods

HeurAtt the responses are designed for each individual worker, while under OptAtt, the collective effect of malicious workers is considered. Therefore correlation modeling in BCC can counter the possible collusion of malicious workers, reflected in BCC's more resilient behavior under OptAtt. Overall, D&S is most resilient if only untargeted HeurAtt is anticipated. However, D&S is vulnerable to all other attacks. Between KOS and BCC, KOS has a slight edge considering all four attacks.

- The neural network based method LAA-S dominates all others in all attacks except having a slightly higher susceptibility than D&S in untargeted HeurAtt in the low redundancy dataset (Product). We suspect this is due to the sparsity of the task representation vector in contrast to the large number of network parameters combined with the lower quality of malicious answers in HeurAtt. The superior performance in all other cases can be attributed to its network of parameters, which can be considered as an implicit and sophisticated (non-linear) model of worker reliability.

- Comparing different techniques, besides the best performing neural networks,

PGM based methods are generally more robust than optimization based methods and direct computation methods. This is also consistent with the findings in [107] under normal settings with varying worker reliability.

- Comparing different worker models, the confusion matrix based methods generally outperform those with a single reliability model and MV method with no worker model. Neural network based method LAA-S, even though with no explicit worker model, achieves the best performance thanks to its network of parameters, which can be considered as an implicit and more sophisticated (non-linear) model of the reliability associated with each worker.

Comparison of Attacks From the attack point of view, OptAtt is more effective compared to HeurAtt, especially for untargeted attacks. This is not surprising given the adversarial knowledge and OptAtt strategy. While OptAtts can only be carried out in white-box and gray-box settings given full or partial adversarial knowledge, as we have shown, they can be very effective under gray-box settings even without the knowledge of the inference method or with a small percentage of normal workers' answers. This shows that optimization based attacks can be a real threat to crowdsourcing applications.

For targeted attacks, in a lower percentage of adversaries, HeurAtt can be more successful compared to OptAtt. OptAtt trades off some accuracy drop in exchange for less adversary detection, hence losing its edge in attack power over HeurAtt, solely optimizing for accuracy drop. There is an optimal level of disguise and percentage of targeted tasks for attackers under HeurAtt, however, these may not be easily identifiable as they vary substantially across inference methods and settings.

Comparison of Datasets and Other Factors The comparison between the datasets reveals that a crowdsourcing system with a higher redundancy of answers is generally more robust. It remains an interesting question for a crowdsourcing system provider to find the best trade-off between redundancy and the overall platform cost to ensure the resiliency of the system. However, worker engagement does not have a major impact on the robustness of the system.

Chapter 4

Enhanced Truth Inference Method under Data Poisoning Attack

Despite a wide range of successful applications, crowdsourcing systems have a weakness. They are vulnerable to adversarial countermeasures by attackers aware of their use through either reading publications or self-experimentation. Attackers may become aware of the truth inference algorithm details, e.g., parameters used, and modify their behavior to evade detection. More powerful attackers can also actively tamper with the system by polluting the provided answers, reducing or eliminating its efficacy.

In this chapter, we investigate the robustness of the truth inference algorithm against adversarial attacks in the context of enriching the prior knowledge and equip the algorithm with augmenting data into a specific portion of tasks.

4.1 Problem Definition

Crowdsourcing is a paradigm that provides a cost-effective solution for obtaining services or data from a large group of users, or *crowd*. Amazon's Mechanical Turk (MTurk) and Waze are well-known examples of crowdsourcing systems aggregating human wisdom to estimate the true answer for their corresponding tasks. Many businesses use MTurk to complete simple tasks, for example, tagging images or completing a survey [46]. Another example is Waze [87], a crowd-driven navigation

application. Waze offers users more ways to share information on accidents, police cars actively, and even contribute content like editing roads, landmarks, and local fuel prices. For example, users can report the traffic status at various locations, which is then aggregated to update the traffic condition shown on the map. Today, Google integrates selected crowdsourced data from Waze into its own Maps application.

Although crowdsourcing is a cost-effective solution, attackers could easily take advantage of it and exploit a large number of workers to elevate or reduce support for products or opinions artificially. For example, the rating system of restaurants in the Yelp application could be manipulated by creating fake reviews. Studies have shown [**?**, 3] that the revenue of restaurants in Yelp application is increased up to 9% when the rating score of that restaurant is increased just by one score. Another example of attacks that happened in Foursquare, Yelp and YikYak that attackers can cheaply emulate numerous virtual devices with forged locations to overwhelm these systems via misbehavior. Misbehavior can range from falsely obtaining coupons on FourSquare or Yelp to imposing censorship on YikYak.

Since the answers are collected from non-expert workers in the crowdsourcing system, the collected answers often contain inherent noise. One important component of crowdsourcing systems is *truth inference*, which infers the true labels from the answers provided by workers. Majority voting (MV) is a straightforward method to aggregate answers which naively assumes that all workers have the same reliability. Besides MV, advenced methods such as probabilistic graphical model (PGM) based, and neural network based [19, 20, 39, 43, 74, 98] methods have been proposed to improve the performance of truth inference by considering various parameters such as the reliabilities of workers or the difficulties of tasks.

Most truth inference methods were designed without consideration for malicious intents. However, crowdsourcing applications may be subject to *data poisoning attacks* [40, 80, 88] where malicious users may intentionally and strategically report incorrect information to mislead the system to infer the wrong truth for all or a targeted set of tasks. In the Waze example, the competitors may intend to tarnish

Waze's reputation by providing wrong answers to decrease the system's overall accuracy. This can be often achieved via Sybil attacks [16, 22, 95, 101] where an attacker creates a large number of Sybil workers to report wrong answers strategically.

This chapter defines robustness on the truth inference method in the crowdsourcing system and provides a high-level overview of attack settings in a crowdsourcing system.

4.1.1 Problem Definition

Given a set of tasks $\mathbf{T}$ and a pool of workers $\mathbf{W}$, each task $t \in \mathbf{T}$ is assigned to a subset of workers $w \in \mathbf{W}$. Each worker w_i provides an answer to each of their assigned tasks. The goal of truth inference is to determine the true answer $\hat{\mathbf{Z}}$ based on all the answers provided by the workers for each task. The tasks in a crowdsourcing system can be classified into 1) decision-making tasks where workers select a binary answer such as yes or no, 2) multi label tasks where workers select one label among multiple candidate labels, and 3) numeric tasks where workers provide answers with numeric values. The truth inference methods may consider different factors such as type of tasks, level of difficulty of tasks, and task assignment methods [107]. In this paper, we focus on the decision-making tasks, i.e., the binary truth inference problem, and do not consider other variations.

Definition 4.1. *(Truth Inference) Given a set of tasks $\boldsymbol{T}$, set of workers $\boldsymbol{W}$ and a bipartite graph indicating tasks assigned to each worker, a truth inference method returns a set of predicted true label for tasks, denoted as $\hat{\boldsymbol{Z}}$.*

A certain percentage of attackers may behave maliciously and strategically attempt to flip the true label of tasks in an adversarial environment. The goal of robust truth inference is to effectively infer the truth through correct estimation of the label even in the presence of malicious workers.

4.1.2 Attack Setup

Based on the adversary's level of knowledge, attacks can be classified into black-box, gray-box, and white-box attacks. In black-box attacks, the adversaries only know about their assigned tasks. In white-box attacks, the adversary has full knowledge about the inference method being used, the task assignment and answered provided by other workers. In gray-box attacks, the adversary may have partial knowledge of the above. The two attack methodologies based on different levels of adversarial knowledge are described as follows.

Heuristics Based Attacks

We adapt the heuristics based attacks in a black-box setting when the malicious workers do not know each other, as a non-collusive strategy. The most straightforward heuristic for an attacker would always be to report the wrong answer for each of their assigned tasks. However, this may be easily recognized by most of the truth inference systems (besides majority voting) which model the workers' reliability. The attackers could disguise themselves as honest workers by providing true answers for some tasks so that they won't be detected by the system. To model this behavior, we use the following heuristics approach [82].

The worker's behavior is modeled by defining a confusion matrix π^w that captures a worker's probability of providing a certain label given the true label. Given the label set $L = \{0, 1\}$, α and β indicate the probability of workers provide a correct label given the true label of 1 and 0, respectively. Each malicious worker w' is associated with a malicious confusion matrix $\pi^{w'}$ with α' and β' and a normal confusion matrix π^w with α and β, and a disguise parameter γ. w' behaves as a normal worker modeled by π^w with probability γ and as a malicious worker modeled by $\pi^{w'}$ with probability $1 - \gamma$. This parameter helps attackers to obfuscate their behavior and deceive the system into not detecting them as malicious workers. For example, a malicious worker with a moderate disguise may have $\pi^w = \begin{bmatrix} \beta = 0.85 & 0.15 \\ 0.05 & \alpha = 0.95 \end{bmatrix}$ and $\pi^{w'} = \begin{bmatrix} \beta' = 0.05 & 0.95 \\ 0.9 & \alpha' = 0.1 \end{bmatrix}$, and $\gamma = 0.2$.

Optimization Based Attacks

We adopt the optimization based attack [65, 82] in a white box setting where an adversary has full knowledge of the truth inference algorithm being used, answers provided by other workers, and task assignments, therefore, can optimally inject manipulated answers to maximize the damage to the system. The attack goal is to maximize the number of flipped labels before and after the attack along with maximizing the attackers' collective confusion matrix parameters (reliability) inferred by the system. Intuitively, this will help them to obfuscate their malicious nature and cause more disruption in the system. Let $\hat{z^a_{t_j}}$ and $\hat{z^b_{t_j}}$ denote the inferred answer by the DS method after and before attack for task t_j, respectively, and $\hat{\alpha}_{w'}$ and $\hat{\beta}_{w'}$ represent the inferred confusion matrix parameters of the malicious worker w'. The optimization problem can be formulated as follows:

$$\max_{\mathbf{C}'} \sum_{j=1}^{M} 1(\hat{z^a_{t_j}} \neq \hat{z^b_{t_j}}) + \lambda \sum_{w' \in \mathbf{W}'} (\hat{\alpha}_{w'} + \hat{\beta}_{w'}) \tag{4.1}$$

where λ controls the trade-off between the objectives of maximizing the inferred collective reliability of malicious workers and maximizing the number of flipped labels.

4.2 Defense Methodology

This section investigates a solution that brings robustness in terms of the performance of truth inference methods in the crowdsourcing system. Our solution is based on identifying sensitive tasks with a higher chance of manipulation by adversaries named boundary tasks. Then, utilizing the boundary tasks and incorporating more substantial prior to preserve the robustness of the system in the presence of data poisoning attacks.

This section proposes two solutions: 1) Edge-NN and 2) Edge-PGM that benefits from data augmentation technique followed by the enhanced inference method. Figure 4.1 depicts the overall framework of our solution.

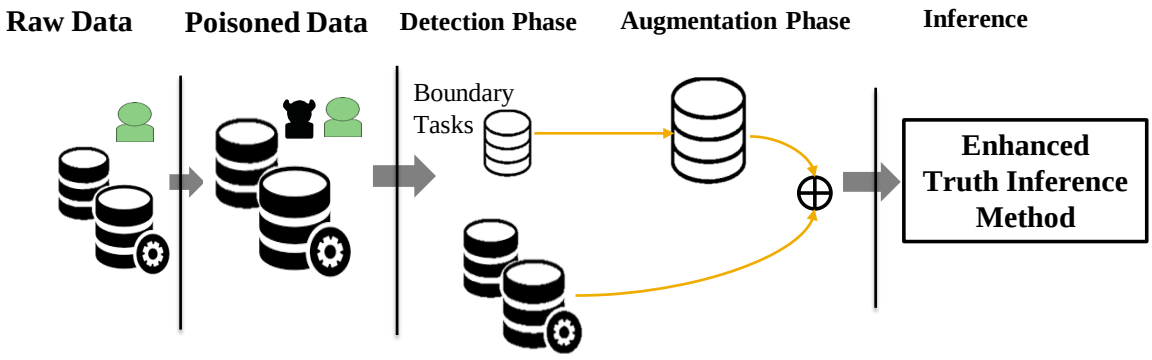


Figure 4.1: Overall Framework of EdgeInfer Solution

4.2.1 Boundary Task based Data Augmentation

Intuitively, malicious workers can gain power in a crowdsourcing system and manipulate the result of predicted labels for some but not all tasks by preventing normal workers from reaching consensus, which generally reflects the task's true label. These boundary tasks for which workers fail to reach a firm consensus are particularly vulnerable to manipulations and may lead to wrong inferences. Hence our proposed approach focuses on these boundary tasks. We present two phases here, 1) detection phase in which vulnerable tasks (i.e., boundary tasks) are identified; and 2) augmentation phase in which matrix completion technique is utilized to nullify the misleading engineered answers of malicious workers.

Detection Phase The vulnerable boundary tasks for which workers do not reach a strong consensus are detected.

Definition 4.2. *(Boundary task) Given a set of* $\boldsymbol{M}$ *tasks and a set of* $\boldsymbol{N}$ *workers as* $t \in \boldsymbol{T} = \{t_1, ..., t_m\}$ *and* $w \in \boldsymbol{W} = \{w_1, ...w_n\}$*, a label set* $\boldsymbol{L} = \{0, 1\}$ *and answer matrix* $\boldsymbol{C}_{N,M}$*, the subset of tasks are called boundary tasks if the certainty in workers' majority label is less than or equal to a threshold* δ*.*

$$\mathbf{BT} = \{t : \ t \in T, \ \max\ (p_t^0, \ p_t^1) \ <= \delta\} \tag{4.2}$$

where p_t^0 and p_t^1 are the probability of the truth label of task t to be 0 and 1, respectively.

Figure 4.2 shows a crowdsourcing system consisting of 10 workers and four tasks where six workers answer each task. For predicting the true label of task t_2 and task t_4, workers strongly agree on label 1 and label 0 for task t_1 and task t_4, respectively. The certainty in workers' majority label is 83% and 100% for t_2 and t_4, which are quite high, so the malicious workers might not have enough power to flip the true label of these tasks. Therefore applying data augmentation for these tasks would be unnecessary and may even introduce noise. However, workers tie on the labels for task t_1 and have a weak consensus on the labels for task t_3. Therefore, malicious workers would potentially have a much greater chance of flipping the true label of these tasks, we call these tasks with less certainty for the majority label as boundary tasks. Since the inferred label of boundary tasks is more likely to be inaccurate, we run the matrix completion on boundary tasks and concatenate the completed matrix to the non-boundary tasks. Then, the truth inference method is run on this augmented answer matrix.

Tasks	C (answers and assigned tasks)						$P(\hat{z}=0 \mid C)$	$P(\hat{z}=1 \mid C)$
t_1	W_1	W_2	W_3	W_4	W_5	W_6	50%	50%
	1	0	1	0	1	0		
t_2	W_1	W_2	W_3	W_4	W_5	W_6	17%	83%
	0	1	1	1	1	1		
t_3	W_1	W_2	W_3	W_4	W_5	W_6	34%	66%
	1	0	0	1	1	1		
t_4	W_1	W_2	W_3	W_4	W_5	W_6	100%	0%
	0	0	0	0	0	0		

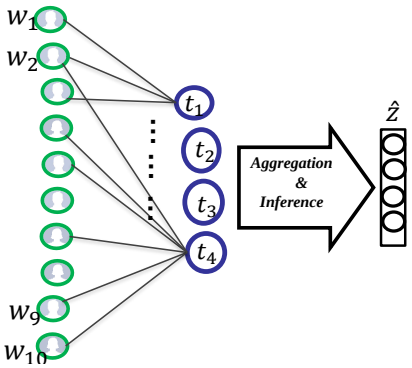


Figure 4.2: Example of a Crowdsourcing System

4.2.2 Augmentation Phase

In a comprehensive comparison of truth inference methods, Zheng et al. [107] point out that MV outperforms other inference methods in case of complete answer set, i.e. all workers provide answers to all tasks. Based on this conclusion, having a complete answer can be advantageous in improving a crowdsourcing system's accuracy. Furthermore, study [82] shows that redundancy, i.e., the average number of workers assigned per task, is an essential factor in resilience against attacks in crowdsourcing.

However, real-world data are typically sparse, meaning the number of answers per task is relatively low, even though some workers can answer a large number of tasks. Motivated by this, we propose a data augmentation method to address the sparsity challenge and enhance existing truth inference methods' robustness. The augmentation phase is only applied to boundary tasks as a pre-processing step before the inference to neutralize adversaries' potential contaminated data.

Matrix Factorization is a common technique in data compression and feature learning [49,78]. Using this technique for matrix completion shows the underlying interactions between workers, tasks, their corresponding level of worker reliability and task difficulty. This technique factorizes a matrix to find two matrices such that their product would generate the original matrix.

Definition 4.3. *(Matrix Factorization): Given a set of* $\boldsymbol{N}$ *workers and a set of* $\boldsymbol{M}$ *tasks. Let* $\boldsymbol{C}$ *of size* $|N| \times |M|$ *be the matrix that contains all the answers that the workers provided to the tasks. Find two matrices* $\boldsymbol{P}_{|N|,|K|}$ *and* $\boldsymbol{Q}_{|M|,|K|}$ *such that* $P.Q^T$ *approximate* $\boldsymbol{C}$*.*

Each row of $\mathbf{P}$ and $\mathbf{Q}$ draws the association of workers and tasks with features. To obtain $\mathbf{P}$ and $\mathbf{Q}$, gradient descent is used to minimize the difference between their product and the answer set C, iteratively. To avoid overfitting, a parameter η is used to control the magnitude of the workers-feature and task-feature vectors such that $\mathbf{P}$ and $\mathbf{Q}$ produce an accurate approximation of $\mathbf{C}$ without having the elements of

these matrices to be unnecessarily large.

$$\min \quad (\mathbf{C}_{n,m} - \sum_{k=1}^{K} p_{n,k}\ q_{k,m})^2 + \frac{\eta}{2} \sum_{k=1}^{K} (\|P\|^2 + \|Q\|^2) \tag{4.3}$$

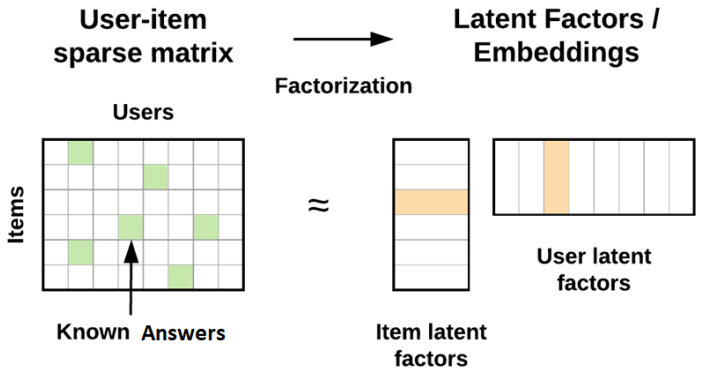


Figure 4.3: Example of Matrix Factorization

Figure 4.3 shows that the element is expressed as the inner-product of two vectors, the corresponding row vector of Item (I) and column vector of User (U). Our learning task is determining the value of matrices I and U by enforcing $I_i U_j$ to be similar to $C_{i,j}$ for all observed entries. Although the answer matrix C is sparse, I and U would be dense matrices, provided that we have at least one observation for all Items (rows) and all Users (columns). As I and U are dense matrices, we can estimate unseen entries by taking the inner-product of two corresponding vectors from I and U.

Our augmentation approach is generalizable and agnostic of the inference method, and it is applied as a processing step before inference to increase the redundancy of answers. There are studies based on tensor completion [**?**, 110] that use matrix completion as an inference method under a non-adversarial setting and apply it on the entire provided-answer matrix and their empirical results are generally not as impressive as the other state-of-the-art methods, therefor, we exclude them from our analysis.

4.2.3 Enhanced Inference Method

Inference involves aggregating provided answers to estimate the true label of each task. The details of inference methods in Edge-NN and Edge-PGM approaches are described here.

Edge-NN Inference: The recent survey paper [107] compares the traditional inference methods and concludes that DS is one of the overall winners. Also, this study shows [82] that LAA-S as the latest neural network based method outperforms previous methods. Here we present a neural network based approach enhanced with a more substantial prior to achieving more robustness. The existing state-of-the-art neural network based approach LAA-S [98] adopts a variational autoencoder (VAE) network [45] to leverage the learned latent truth label distribution that best represents the original task vector for inference. However, LAA-S can also be subjective to poisoning attacks as it is employing the distribution of the original answers as a prior (essentially majority voting). Our main idea is to enhance it with a stronger prior that considers workers' reliability. We propose a hybrid model that utilizes the neural network model while incorporating the DS result as its prior. Since DS models the reliability of workers as a confusion matrix it outperforms majority voting in terms of approximating the true labels of the tasks. Therefore, utilizing the distribution labels of DS as the prior might result in some unsophisticated attackers being filtered out from the system.

Figure 4.4 shows our proposed enhanced inference method composed of two parts: 1) prior estimation that approximates the distribution of labels using the inferred truth of the tasks from DS inference method and 2) inference through the enhanced LAA-S algorithm, while leveraging the prior obtained in the previous step.

DS method [19] is a PGM based method that models the reliability of each workers with a confusion matrix. It utilize the EM algorithm to calculate the maximum likelihood and estimate the item's true label and worker's reliability (i.e. their confusion matrix).

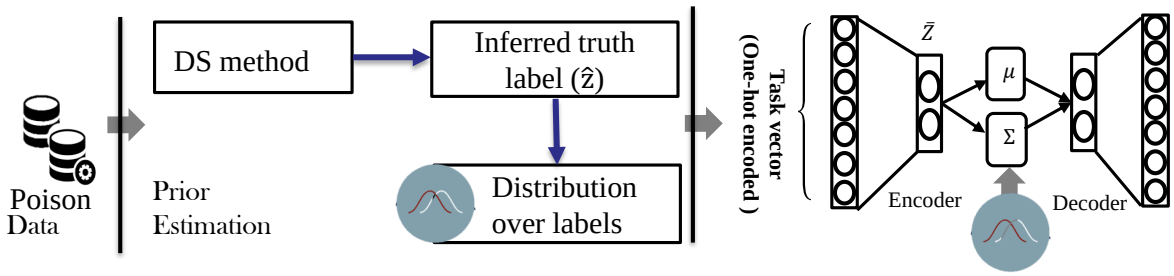


Figure 4.4: Components of Inference Method in Edge-NN

LAA-S [98] inference algorithm adopts a VAE network [45] to leverage the learned latent truth label distribution that best represent the original task vector. This model contains two shallow neural networks: 1) an encoder or classifier (q_θ) transforming the task vector (v) into the latent feature (z) indicating the truth label, and 2) a decoder or reconstructor (p_ϕ) which recovers a task vector based on the latent features.

Inputs and output of the VAE represent each task as a vector consisting of the one-hot encoding of the provided answers by each worker to the task. The network is trained on all one-hot encoded task vectors ($\mathbf{v}$) by minimizing the reconstruction error between the original and recovered task vectors. Additionally, the training also considers how closely the learned estimated ground truth resemble those inferred by DS, supplied to the model as the prior. The objective function is shown below.

$$\min_{q_\theta, p_\theta} \mathbb{E}_{q_\theta(z|v)} log p_\phi(v|z) - D_{KL}(q_\theta(z|v)||\ prior) \tag{4.4}$$

where the first term is the reconstruction error and the second term enforces the distribution of inferred labels to follow a specific prior through the negative KL divergence D_{KL} term.

Edge-PGM Inference: GLAD inference method [92] is a PGM based model that considers workers' reliability and the difficulty level of task to estimate the true answer of tasks. One drawback of GLAD method mentioned in Zheng et al. study [107] is that in some cases the estimation of the difficulty level parameter related to the

tasks is inaccurate. Hence, the GLAD fails to outperform the other inference methods.

We adopt the GLAD truth inference method which incorporates the level of difficulty of tasks into their inference method by replacing the vanilla prior of task difficulty level with a better prior based on boundary tasks [92]. Our key insight is that the more difficult tasks are the one that workers fail to reach a strong consensus on and therefore are particularly vulnerable to manipulations and may lead to wrong inference. Figure 4.5 depicts the Components of Inference Method in Edge-PGM.

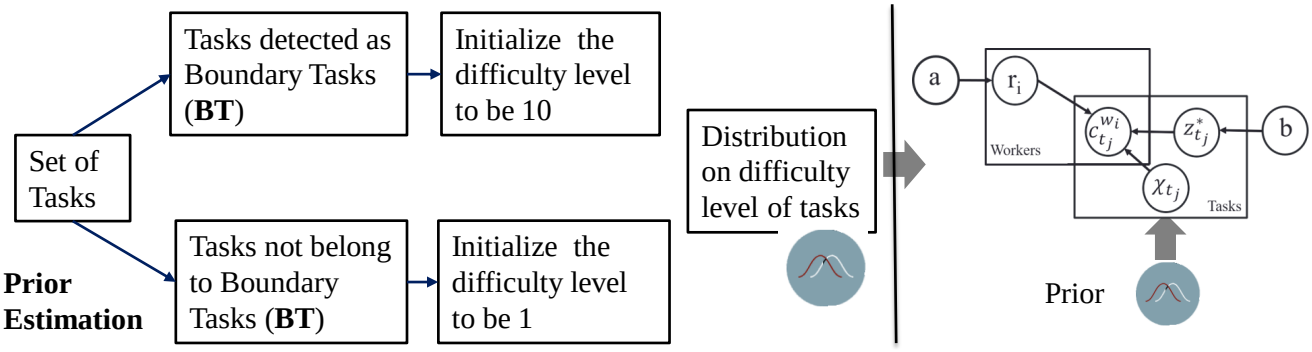


Figure 4.5: Components of Inference Method in Edge-PGM

GLAD models each client i reliability as a single value $r_i \in [0, +\infty)$, a higher value implies a higher reliability, and each task's j difficulty level as a single value $1/\chi_{t_j} \in [0, +\infty)$. A high value of $1/\chi_{t_j}$ implies task t_j is more sensitive or difficult. The true label of tasks ($\hat{Z}$) are estimated based on workers' reliability (r) and tasks' difficulty (χ) parameters, as follow:

$$p(z_{t_j} | C, r, \chi) = \prod_{i \in \mathbf{W}^{t_j}} \frac{1}{1 + e^{-r_{w_i} \chi_{t_j}}} \tag{4.5}$$

The Edge-PGM inference method initializes the difficulty level (χ) of detected boundary tasks as 10 and the other tasks are all treated equally and their difficulty level is set to 1, as follow:

$\{1/\chi_{t_j} = 10 \, if \, t_j \in \mathbf{BT} \, else \, 1/\chi_{t_j} = 1 \; for \, t_j \in \mathbf{T}\}$

Then the expectation–maximization (EM) algorithm is iteratively applied to estimate parameters (r,χ).

4.3 Experiments and Results

4.3.1 Experiment Setup

In this section, we introduce the datasets and conduct experiments on them to evaluate the performance of the proposed robust mechanism.

Datasets: We tested our method on three public benchmark datasets for decision making tasks. Table 4.1 shows the summary of the datasets and their key properties.

Table 4.1: Statistics and Properties of Datasets

Dataset	**Product**	**PosSent**	**Temp**
N (# of tasks)	8,315	1,000	462
M (# of workers)	176	85	76
V (# of answers)	24,945	20,000	4620
Redundancy (# of answers per task)	3	20	10
Engagement (# of answers per worker)	141	235	60
Avg workers' credibility	0.79	0.798	0.73
Truth Labels Ratio (negative,positive)	(88%, 12%)	(52.8%, 47.2%)	(50%, 50%)

- Product Dataset. This dataset includes 8315 tasks where each task is a question about a comparison of two products. An example is "are iPad Two 16GB WiFi White and iPad 2nd generation 16GB WiFi White the same?" [38].
- PosSent Dataset. This dataset contains information about the general sentiment of a tweet about the reputation of a company. Workers assess each tweet

and provide positive label, meaning that the tweet will increase the reputation of the company, or negative to indicate the opposite [5].

- Temp Dataset. In this dataset, each task is to identify whether or not one event happened before another in a given context. [**?**]. An example news text is "John fell. Sam pushed him.", and the task is to decide if the events that the colored words describe happened before or after each other. "pushed" happened before "fell".

Poisoning Dataset: For poisoning answers, assuming adversaries W', with the fraction of malicious workers being $\frac{|W'|}{|W|+|W'|}$, we applied heuristics based (Black Box) and optimization based (White Box) attacks given the attack strategies described in Section 4.1.2. Heuristic based attacks are designed based on black-box knowledge and the disguise parameter (γ) is set to 0.0. Optimization based attacks are designed based on white-box knowledge and use DS as a inference methods in which worker's reliability is modeled based on confusion matrix, we set the λ parameter to be 1.

Parameters: In Edge-NN method, for data augmentation phase, the learning rate for Temp, PosSent and Product dataset are set as [0.01, 0.01, 0.001], respectively. Also the number of latent dimensions for each of the Temp, PosSent and Product dataset are set to [13, 13, 20], respectively. The regularization parameter(η) is set to 0.005. In Edge-PGM method, the range for δ parameter is limited to [50%-70%] since other values are just complementary to this interval. The δ in Edge-PGM is chosen as 0.55 for all the experiments.

Metrics and Comparison: We evaluate inference performance using accuracy and F-score, computed based on the predicted labels and ground truth. Since the Product dataset is heavily unbalanced, F-score is chosen as the metric. For the other two datasets, we report accuracy. Accuracy is defined as the fraction of tasks whose truth are inferred correctly.

4.3.2 Experiment Results

We conduct several experiments with three real datasets to assess the effectivness of our solutions.

Effect of Certainty Threshold (δ) on Accuracy/F-score: Figure 4.6 shows the effect of the parameter δ (certainty threshold) on accuracy/f-score of the proposed method, Edge-NN, for different percentage of malicious workers. We run this experiment on three datasets, and contaminated these datasets based on heuristic and optimization attacks described in Section 4.1.2.

The δ= 1 corresponds to augmenting all tasks and δ= 0.45 corresponds to no augmentation. When the % of malicious workers (%mal) is low, having no augmentation performs the best and the accuracy drops when augmentation increases. This trend is as expected since the data has high quality and augmentation does not help. However, as the %mal increases, accuracy increases as augmentation increases and then drops back when augmenting all tasks. This verifies the benefit of augmenting the boundary tasks only when there is significant noise in the data.

As show in Figure 4.6, we observe that the accuracy of the system is sensitive to the value of noise added for augmentation on boundary tasks. Note that the number of candidate tasks is directly proportional to the certainty threshold (δ). Furthermore, it is shown that the optimal δ for an effective defense is dependent on %mal. Intuitively, at a higher %mal, there will probably be more contaminated tasks, and so by choosing a higher δ, we will apply augmentation on more tasks. We replicated this experiment on the PGM based methods and observed similar trends, which stresses that the success of boundary task augmentation is not bound to a specific inference method. This trend remains the same in both heuristic and optimization attack.

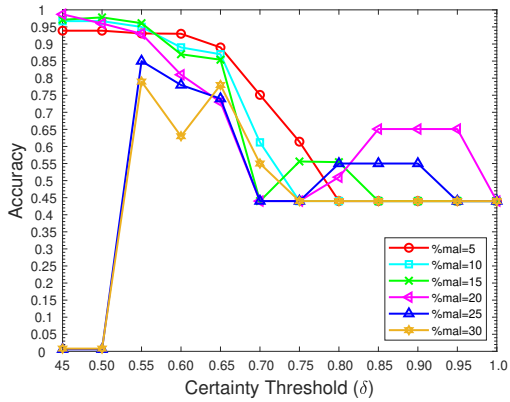
For the remaining comparison, we choose to set δ as 0.55, 0.55, 0.65 for Temp, PosSent and Product dataset for heuristic and optimization attack, respectively. In general, the range of [0.5, 0.65], corresponding to the boundary tasks, gives a good

overall performance over varying percentage of malicious workers. This also confirms our intuition that it is beneficial to only perform data augmentation on the boundary tasks. We note that in practice, the percentage of malicious workers will not be too high due to the cost of creating sybil workers. Given that Product dataset is comparatively more sparse it would be harder to reach a strong consensus with fewer labels for each task. Therefore, to correctly infer the truth of those boundary tasks, more of them should be included in matrix completion, hence a higher $\delta = 0.65$ is selected.

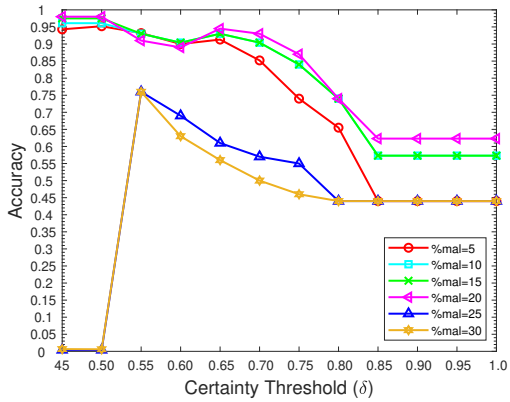
Ablation Study: We assess the impact of augmentation as a preprocessing step and utilizing an enhanced prior on the three truth inference methods, DS, GLAD, and LAA-S (i.e. majority voting as prior). The methods that just consider the edge augmentation are called in the form of *Edge-$method-name$*. The methods with enhanced prior are named in the form of *$method-name$+*. Also, the methods that combine both of these techniques are named as *Edge-$method-name$+*. For example, Edge-DS is a DS inference method with edge augmentation, LAA-S+ is a LAA-S method with enhanced prior (i.e. DS as prior) and Edge-LAA-S+ is a LAA-S method with enhanced prior along with edge augmentation.

First, we assess the impact of augmentation, as shown in Figure 4.7, in all three datasets, Edge-GLAD, Edge-DS and Edge-LAA-S+ outperform GLAD, DS and LAA-S, especially in the most likely range for %mal, from 20 to 30. This confirms the benefit of the data augmentation technique in enhancing the performance of the existing truth inference methods.

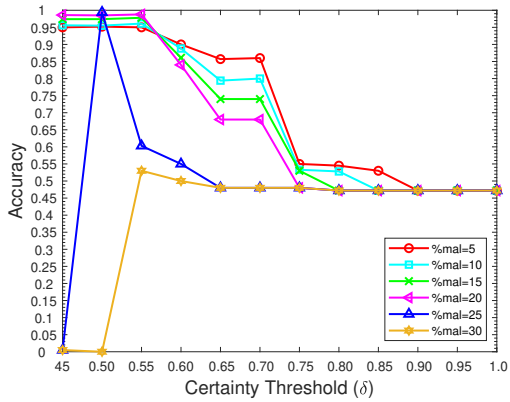
Moreover, we assess the impact of using DS as prior on the LAA-S method instead of majority voting. As it is shown in Figure 4.7, using the DS as a prior (LAA-S+) helps to slightly improve the performance of the model, however, if the number of malicious workers is higher than 20%, the DS model completely failed, therefore using DS could not outperform the original LAA-S model. Edge-LAA-S+ (i.e. Edge-NN) outperforms other methods and verifies the benefit of both augmentation as a preprocessing phase and using DS as prior.



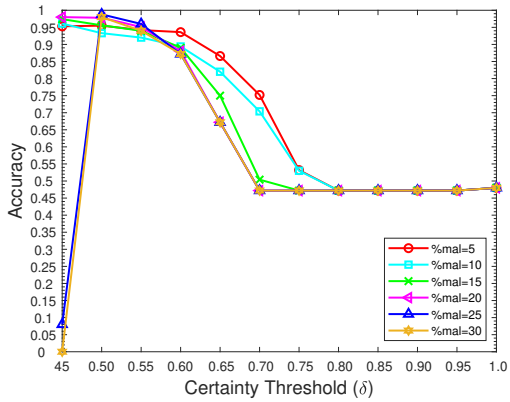
Temp Dataset: Heuristic



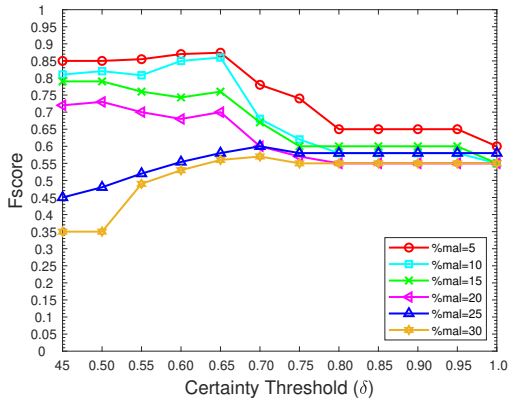
Temp Dataset: Optimization



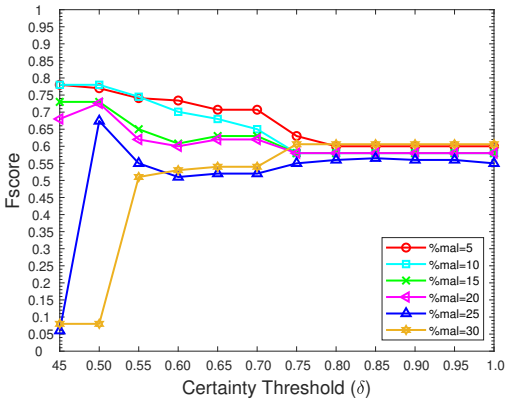
PosSent Dataset: Heuristic



PosSent Dataset: Optimization

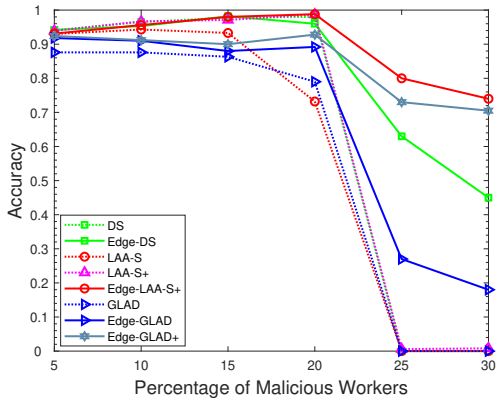


Product Dataset: Heuristic

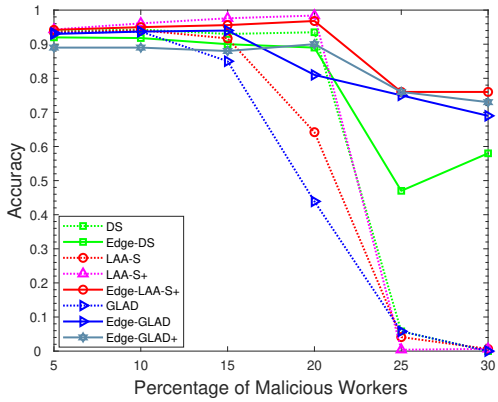


Product Dataset: Optimization

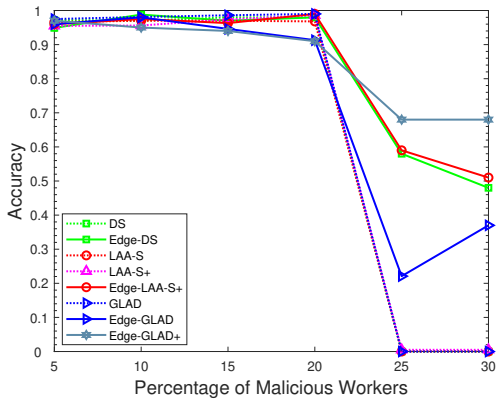
Figure 4.6: The Effect of Certainty Threshold (δ) on Accuracy across different %mal



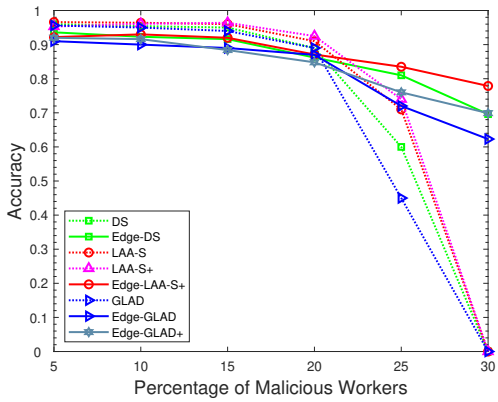
Temp Dataset: Heuristic



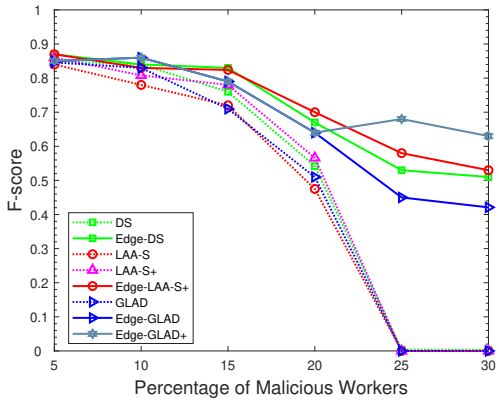
Temp Dataset: Optimization



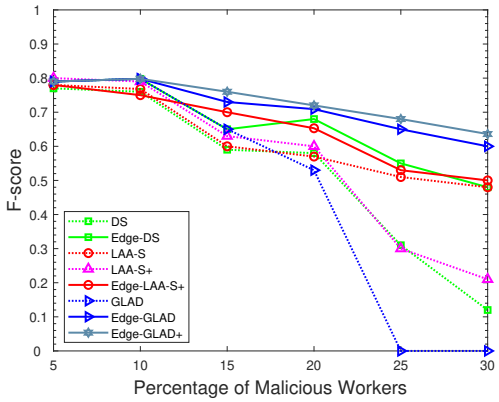
PosSent Dataset: Heuristic



PosSent Dataset: Optimization



Product Dataset: Heuristic



Product Dataset: Optimization

Figure 4.7: Ablation Study: Robustness vs $\%mal$

Furthermore, we assess the impact of using different distribution for boundary tasks versus other tasks as prior on the GLAD method instead of using uniform distribution for all tasks. As it is shown in Figure 4.7, differentiating the difficulty level of tasks and using it as a prior (Edge-GLAD+) helps to improve the performance of the model.

Comparison of Edge-NN and Edge-PGM: Figure 4.7 shows that Edge-NN (i.e. Edge-LAA-S+) and Edge-PGM (i.e. Edge-GLAD+) are robust against both type of attacks (i.e. black-box and white-box) and reduce the attack's success rate.

Edge-NN performs better than Edge-PGM when the %mal is less than 20%, since the performance of Edge-NN depends on the prior and DS method at that specific interval performs well. By increasing the number of malicious workers in the system the DS method is unable to perform well and it effects the performance of Edge-NN method. However, Edge-PGM outperform Edge-NN when the %mal is greater than 20%.

Chapter 5

Robust Federated Learning under Data Poisoning Attack

Artificial intelligence (AI) is becoming an inevitable part of our life by spreading itself through all applications in various domains. Training a meaningful predictive machine learning model by access to vast volumes of data is challenging since it can be costly to collect data from clients or organizations. Also, sharing data with any third party could compromise the privacy of clients or organizations. Recently, federated learning has emerged as a promising new collaborative learning framework to build a shared model while keeping the clients' data private [1, 58, 62].

The federated learning framework is quite practical and also flexible enough to be applied in various domains, such as conversational AI and healthcare [58, 62, 68]. Training a model for these domains requires a diverse dataset. Access to data from several organizations and centralized them in a third-party service provider is a burden and can be impractical considering the personal information privacy regulations. Yet, we still wish to use data across various organizations because a model trained on data from one organization might show poor generalization performance. Utilizing federated learning becomes promising to collaboratively train a machine learning model with good generalization performance without the requirement to share raw private local datasets [1].

In this chapter, We consider the adversarial attacks problem in the federated learning setting in which a global server is responsible for aggregating model parameters

shared by clients. Our goal is to efficiently assign reliability score to clients and incorporate the parameters shred by reliable clients.

We use a trustworthy approach, originally proposed in the crowd-sourcing domain, against such attacks in a federated learning setting to measure the trustworthiness of provided updates. The intuition is that in federated learning the provided updates by benign clients would be similar to each other in each round. Therefore the global server can implicitly learn to rely more on clients' updates that are similar to each other. Measuring the clients' trustworthiness can be estimated by adopting a truth inference mechanism. We derive our aggregation algorithm by incorporating the reliability of each clients' provided parameters. Based on estimated reliability, the one is chosen if their reliabilities are not far from the other clients. Finally, the local parameters are aggregated based on the estimated reliability of clients in the proportion of the number of local data trained.

5.1 Problem Formulation

In a federated learning framework, there is a central node (global server) and several participating clients. The global server controls the learning process and aggregates the parameters submitted by clients during multiple communication rounds. The clients train the same model locally using their local datasets. Then, they share their updated local model parameters, not their raw data, with the central node (i.e., server), which aggregates all their contributions and broadcasts back the updated global model parameters.

The common federated learning aggregation algorithm is called FedAvg [62] that takes the weighted average of updated locally model parameters. This aggregation method is vulnerable to adversarial attacks or unintentional errors in a system. Due to strategic adversarial behavior or infrastructure failure in the system, some of the clients can become faulty during training and send malicious or arbitrary values to the global server. Thus, the overall convergence performance can be affected severely.

This kind of fault in which client nodes act arbitrarily is called Byzantine faults [**?**]; therefore, robust federated learning is necessary.

Recently, several methods have been proposed to mitigate attacks in federated learning or distributed learning [8, 14, 25, 26, 97]. The statistical method such as median or trimmed mean aggregation algorithms [97] perform well under Byzantine attack however these methods are failing under other types of attacks such as label-flipping and Gaussian noise attacks [8, 18, 31, 42].

5.1.1 Federated Learning

The federated learning framework is important when the participating organizations desire to keep their data private. Instead of sharing data, they share the model parameters to take advantage of a high volume of data with different distribution and improve the model's generalization. Federated learning consists of K clients and a global server, G. The same model is shared among the global server and all clients. Each client c_i accesses to their own local dataset $\mathbf{D}_i$, where $|D_i| = l_i$. The total number of samples across all the clients is $\sum_{i=1}^{K} l_i = l$. Each client c_i keeps their data private, i.e. $\mathbf{D}_i = \{x_1^i, ...x_{l_i}^i\}$, from the global server. The goal of federated learning is to learn a global model parameters vector with n parameters $w_G \in \mathbb{R}^n$ in which this parameter vector minimizes the loss among all samples $D = \bigcup_{i=1}^{K} D_i$ in the aim that the global model generalizes well over the test data, D_{test}.

At each time step t, a random subset from the clients is chosen for synchronous aggregation, i.e. the global server computes the aggregated model, then sends the latest update of the model to all selected clients. Each client $c_i \in K$ uses their local data $\mathbf{D}_i$ to train the model locally and minimize the loss over its own local data. After receiving the latest global model, the clients starts the new round from the global weight vector w_G^t and run model for E epochs with a mini-batch size B. At the end of each round, each client obtains a local weight vector $w_{c_i}^{t+1}$ and computes its local update $\delta_{c_i}^{t+1} = w_{c_i}^{t+1} - w_G^t$, then sends the corresponding local updates to the global server, which updates the model according to a defined aggregation rule. The

simplest aggregation rule is a weighted average and formulated as follow:

$$w_G^{t+1} = w_G^t + \sum_{i=1}^{K} \alpha_i \cdot \delta_i^{t+1} \tag{5.1}$$

where $\alpha_i = \frac{l_i}{l}$ and $\sum_{i=1}^{K} \alpha_i = 1$.

5.1.2 Adversarial Model

For the adversarial model, we follow two assumptions: (i) The number of adversaries are less than 50% of whole clients (restricted effect of malicious updates on the global model); (ii) the data is distributed among the clients in an identical independent distribution (IID) fashion. The adversary's goal is to ensure the performance of the system degrade or to cause the global model to converge to a bad minimum.

In the adversarial setting, a subset of the clients act maliciously to the overall accuracy (untargeted attack) or the misclassification of some of the classes or particular data samples (targeted attack). If the malicious client happens to be sampled by the server in each round, it will submit an adversarial update based on its attack strategy. For example, in the case of a label-flipping attack, it changes the true label to the targeted label under attack, updates its local model based on the poison data, and shares the local updates with the server.

We propose a novel defense algorithm called FARel, intending to mitigate the adversarial updates' impact on the global model. At an intuitive level, the server estimates each client's reliability on their submitted parameters and updates the global model parameter based on the estimated reliability. Our main contribution is to introduce the truth inference methods originated in crowdsourcing in a novel way to obtain clients' reliability and design the reliability-based robust aggregation strategy.

Observation					
	t_1	t_2	t_3	t_4	t_5
W_1	1.72	1.62	1.74	1.72	1.72
W_2	1.7	1.61	1.72	1.7	1.71
W_3	1.90	1.85	1.65	1.85	1.85

Tasks	MV	PM	Ground Truth
t_1	1.77	1.72	1.72
t_2	1.69	1.62	1.62
t_3	1.70	1.74	1.75
t_4	1.76	1.72	1.71
t_5	1.76	1.72	1.73

Figure 5.1: Example of crowdsourcing system

5.2 Proposed Robust Model Aggregation

We present our proposed robust aggregation method. We first introduce the truth inference algorithm and utilize this method in federated learning in order to estimate the reliability of provided updates by clients in each round. We further improve upon it by introducing the necessity of applying the outlier detection on our truth inference-based aggregation.

5.2.1 Truth Inference Method

Due to the openness of crowdsourcing, the crowd may provide low-quality or even noisy answers. Thus, it is crucial to control crowdsourcing's quality by assigning each task to multiple workers and aggregating the answers given by different workers to infer each task's correct response. A fundamental component is called Truth Inference, defined as follows: given a set of tasks and a pool of workers, each task is assigned to a subset of workers. Each worker provides an answer to each of their assigned tasks. The goal of truth inference is to determine the true answer based on all the workers' answers for each task.

To better understand of crowdsourcing, Figure 5.1 shows an example, given three workers $\mathbf{W}=\{w_1, w_2, w_3\}$ and five tasks $\mathbf{T}=\{t_1, t_2, .., t_5\}$. As it is shown, e.g., worker w_1 answers t_4 with '1.7', i.e., w_1 thinks that the answer to task t_4 is '1.72'. A naive

solution to infer the true answer per task is Majority Voting (MV), which regards the choice answered by majority workers as the truth. Based on Figure 5.1, the truth derived by MV for task $t_1$is '1.77'.The MV incorrectly infers the true label for that task. Another method, like the PM method, resolves conflicts from different sources for each entry. It provides more accurate results comparing with averaging. Compared with the ground truths, it is clear that worker w_1 and w_2 provides more accurate information (more reliable) while w_3 are not very reliable.

We can map the federated learning into the truth inference by considering the model's weight parameters as tasks. Also, the workers are the clients that provide local parameters to the global server.

Another importan element in a crowdsourcing system is type of the tasks. The type of tasks can be classified into three categories: 1) decision-making where workers select a binary answer such as yes or no, 2) multi-label where workers select one label among multiple candidate labels, and 3) numeric tasks where workers provide answers with numeric values. In federated learning, the type of task falls into the numeric task. Subsequently, we discuss how to obtain each client's reliability based on their submitted parameters to the global server.

Algorithm 2 show the truth inference framework for numeric tasks, the reliability of each worker $i \in [k]$ is denoted as r_{c_i}. It initializes clients' reliability with the same reliability as $r_{c_i} = 1$. Also, it initialize the estimated truth for each weight parameter as the median of all values provided by the clients. Then it adopts an iterative approach with two steps, 1) inferring the truth, and 2) estimating client reliability.

5.2.2 Robust Aggregation Method: FARel

In this section, details of our proposed aggregation method are provided. To begin of each round, we compute the reliability level of each clients by applying the truth inference method.

Let $\delta^t_{c_i} = \{\delta^t_{c_i}[1], \delta^t_{c_i}[2], ..., \delta^t_{c_i}[n]\}$ be the local updates that is shared by client c_i

Algorithm 2 Obtain Clients Reliability

Input: Provided parameters by local clients $\delta_k = \bigcup_{i=1}^{K} \delta_{c_i}, w_G^t$

Output: output: $R = \bigcup_{i=1}^{K} r_{c_i}$

Initialize clients' reliability ($r_{c_i} = 1$ for i $\in K$),
Initialize inferred truth of each update parameter ($\hat{\Delta}_G$) as the median of local updates of δ_k
while *True*: **do**
    Step 1: Inferring the Truth
    for each weight parameter $j \in N$ **do**
        Inferring the $\hat{\Delta}_G$ based on δ_k and **R**
    end for
    Step 2: Estimating client reliability
    for each client **do**
        estimate **R** based on δ_k and $\hat{\Delta}_G$
    end for
    if converge **then**
        break
    end if
end while

at round t. Let $\mathcal{K} = \{c_1, c_2, ...c_k\}$ be the set of clients. Hence, at round t, the updated parameters δ_k^t are collected from K clients. Given the updated parameters δ_k^t provided by K clients, the goal of utilizing the truth inference is to infer the reliability of each clients $R = \{r_{c_1}, ...r_{c_k}\}$ and incorporate this reliability score into the aggregation method.

The idea is that benign clients provide trustworthy local updates, so the aggregated updates should be close to benign clients' updates. Thus, we should minimize the weighted deviation from the true aggregated parameters where the weight reflects the reliability degree of clients. Based on this principle, we utilize the PM method, which is a truth inference method applicable in numerical tasks [53]. First, we need to search for the values for two sets of unknown variables Δ and R, which correspond to the collection of truths and clients'reliability respectively, by minimizing the objective function f(Δ,R). The loss function measures the distance between the aggregated parameters (estimated truth) and the parameters provided by client (observation). When the observation deviates from the estimated truth, the loss function return a high value. To constrain the clients' reliabilities into a certain range, the regularization function is defined and it reflects the distributions of clients' reliabilities.

Intuitively, if a client is more reliable, the high penalty will be received if this client's observation is quite different from the estimated truth. In contrast, the observation made by an unreliable client with a low reliability is allowed to be different from the truth. In order to minimize the objective function, the estimated truth rely more on the clients with high reliability. The estimatd truth and clients' reliabilites learn together by optimizing the objective function through a joint procedure. We formulate this problem as an optimization problem that involves two sets of variables. The values of one set is updated iteratively to minimize the objective function while the values of another set is fixed. This optimization problem is defined as follow:

$$\min_{R,\hat{\Delta}} \sum_{i=1}^{K} r_{c_i} \cdot \ dist\ (\hat{\Delta}_G, \delta_{c_i}^t), \tag{5.2}$$

Where r_{c_i}, $\delta^t_{c_i}$ and $\hat{\Delta}_G$ represent client c_i's reliability, provided update by client c_i at time t, and aggregated updates at time t on the global server, respectively. Also $dist\ (\hat{\Delta}_G, \delta^t_{c_i})$ is a distance function from the aggregated updates of all clients to the clients' provided update.

The goal is to minimize the overall weighted distance to the aggregation parameters in the global server in a way that reliable clients have high weights, as follow, where $dist\ (\hat{\Delta}_G, \delta_{c_i})$, is the distance between the inferred truth and the local update parameter vector provided by client c_i and r_{c_i} is client c_i's reliability [53]. In our problem, the type of parameters provided by clients are continuous, therefore Euclidean distance is used as a distance function, $\sqrt{\sum_{j=1}^{N}\left(\hat{\Delta}^j_G - \delta^j_{c_i}\right)^2}$, where N is the number of local parameters and $\delta^j_{c_i}$ indicates the j-th local parameter shared by client c_i. The client c_i's reliability is modeled using a single value r_{c_i}. Intuitively, workers with answers deviating from the inferred truth tend to be more malicious. The algorithm iteratively conduct the following two steps, 1) Updating the client's reliability and 2) updating the estimated truth for parameters.

Updating the client's reliability In this step, we fix the values for the truths and compute the clients reliability that jointly minimize the objective function subject to the regularization constraints.

Initially, each clients is assigned with the same reliability, $\forall_{i\in\ \mathcal{K}}\ r_{c_i}$=1. The reliability score of each clients after each iteration is updated as:

$$r_{c_i} = -\log\left(\frac{\sum_{j=1}^{N} dist(\hat{\Delta}^j_G\ ,\ \delta^j_{c_i})}{\sum_{k'=1}^{K} dist(\hat{\Delta}^j_G\ ,\ \delta'^j_k)}\right) \tag{5.3}$$

Equation 5.3 indicates that a clients reliability inversely proportional to the difference between its observations and the truths at the log scale. The negative log function maps a number in the range of 0 and 1 to a range of 0 and , so it helps to enlarge the difference in the clients reliability.

Truth computation At this step, the reliability of each clients is fixed, and we update the truth for each entry to minimize the difference between the truth and the client' observations where clients are weighted by their reliability degrees.

At the aggregation process (FARel), the global server incorporate the provided parameters of each clients based on their reliability. Hence, the global parameters are updated as follow:

$$w_G^{t+1} = w_G^t + \sum_{i \in K} r_{c_i}^t \, \alpha_i \, \delta_{c_i}^{t+1} \tag{5.4}$$

5.2.3 Reduce Effect of Malicious Clients: FARel_adapt

The above robust aggregation method, FARel, does not include explicit mechanisms to detect malicious clients. Since the reliability of each client plays a role in the aggregation method, it is important to minimize the effect of the malicious parameters shared by the non-reliable client. Thus, the truth inference method allows us to detect non-reliable clients that provide malicious updates. Algorithm 3 summarizes FARel_adapt robust aggregation method.

Algorithm 3 Robust Aggregation (FARel_adapt)

Input: selected clients K^t, $\mathbf{R}^t(reliabilityofallclients), w_G^t$,
Output: w_G^{t+1}

Cand (set of clients' candidate) initialized to $\emptyset$
$\mathbf{R}^t \leftarrow getClientsReliablity()$
$\mu, \sigma \leftarrow median(\mathbf{R}^t), std(\mathbf{R}^t)$
for each client c_i **do**
    if reliability of client c_i between $\mu - \sigma$ and $\mu + \sigma$ **then**
        Add c_i to **Cand**
    end if
end for
$\mathrm{w}_G^{t+1} \leftarrow w_G^t \; + \; \sum_{i \in [\mathbf{Cand}]} r_{c_i}^t \alpha_i \delta_{c_i}^{t+1}$

Intuitively, we expect to observe the following behavior: the server assigns a higher score as a reliability to the honest clients and a lower reliability score to the mali-

cious one. This behavior is detected when no attack happens in the system or the specific type of attacks such as Byzantine or apply noise to the features is occurred. On the contrary, if a label-flipping attack happens in the system this behavior is shifting, i.e., the truth inference system assigns higher reliability to the malicious clients. Therefore, it is required to disregard the reliability of clients that deviates significantly from the others. To detect anomalous reliability that does not fit with others, outlier detection methods could be effective.

As it is shown in Algorithm 3, after obtaining the reliability of each clients, the median ($\bar{\mu}$) and standard deviation (σ) of the reliabilities are computed for all the clients participated in the round t. The clients whose reliability fit in the range of $[\bar{\mu} - \sigma, \bar{\mu} + \sigma]$ are selected as a candidate, and the global parameters are updated as follow: $\mathrm{w}_G^{t+1} = w_G^t \ + \ \sum_{i \in [\mathbf{Cand}]} r_{c_i}^t \, \alpha_i \, \delta_{c_i}^{t+1}$

5.2.4 Further Improving the Defense Capability: FARel_hist

As we have noted, the above version of our method ignores the temporal relationship between weight parameters. Ignoring this temporal relationship might miss remarkable insights that the parameters shared by clients each round. Therefore, we incorporate the statistical information of the previous rounds during the reliability estimation, which we call (FARel_hist) algorithm. Incorporating the statistical information into the system is depended on the way of choosing the client that can fall into static and dynamic phase:

Static Phase: the selected clients at each round are fixed, therefore, the temporal relationship between weight parameters are added as the statistics of previous rounds. These statistics are added as the new tasks to the vector of weights. These statistics are the number of large weights, number of small weights, median of weights and average of weights.

Dynamic phase: clients will dynamically join or leave the federated training. In this case, we can add median and average of weights from previous round as the weights provided by the new clients.

5.3 Evaluation

5.3.1 Experiment Settings

Dataset.

We consider the following three public datasets.

- MNIST dataset: This dataset contains 70,000 real-world hand written images with digits from 0 to 9 with 784 features. We split this dataset in which training has 60,000 and test data has 10,000 samples.
- Fashion-MNIST (fMNIST) dataset: This dataset consists of 28×28 gray scale images of clothing and footwear items with 10 type of classes. The number of features for this dataset is 784. We split this dataset in which training has 60,000 and test data has 10,000 samples.
- CIFAR-10 dataset: This dataset contains 60,000 natural color image of 32x32 pixels in ten object classes with 3,072 features. We split this dataset in which training has 50,000 and test data has 10,000 samples.
- MNIST and fMNIST dataset: For these dataseta, we use a 3-layer convolutional neural network with dropout (0.5) as the model architecture. The learning rate and momentum set as 0.1 and 0.9, respectively.
- CIFAR-10 dataset: We use VGG-11 as our model. The droput, learning rate and momentum set as 0.5, 0.001, 0.9, repectively.

Adversarial Attacks.

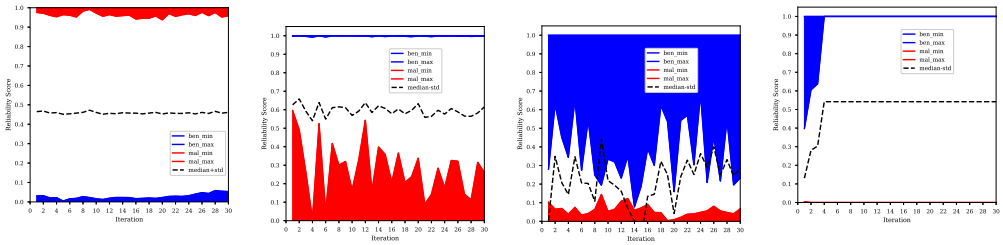
We apply three data poisoning attacks on real-world datasets. We assume an attacker has full knowledge about the training datasets on the local client devices. We consider the training data splits equally across all clients. Among these clients, we assume that 30% of the clients are adversary. We consider three scenarios on how the adversary

client poisoned their data or noisy the updated parameters to compromised the global model.

- Label-Flipping Attacks: Adversaries flip the labels of all local training data on one specific class (e.g., class #1) and train their models accordingly.
- Noisy Data: In MNIST and FMNIST, the inputs are normalized to the interval [0,1]. In this scenario, for the selected noisy clients we added uniform noise to all the pixels, so that $x \leftarrow x + U(1.4, 1.4)$. Then we cropped the resulting values again to the interval [0,1].
- Byzantine Attack: adversary perturb the model updates and send the noisy parameters to the global server. $\delta_i^t \leftarrow \delta_i^t + \epsilon$, where ϵ is a random perturbation drawn from a Gaussian distribution with $\mu = 0$ and $\sigma = 20$.

5.3.2 Experiment Results

Effect of Attacks on Reliability Score of Clients Fig.5.2 shows the reliability range of malicious and benign clients under label-flipping and Byzantine attacks in static mode for FARel and FARel_hist, correspondingly. It shows that the reliability score when confronted with flipping and label Byzantine attack are in contrast with each others in FARel approach. The model assign higher reliability to benign workers in Byzantine attack, however, the opposite behavior is observed in presence of flipping attack.By incorporating the statistical information of previous rounds, the model assign higher reliability to the benign clients in present of flipping attack with lots of fluctuation. We observe that the trend for noisy attack follows the Byzantine trend. As it is shown in Fig.5.2, the maximum and minimum reliability of all benign clients are between $[\bar{\mu} - \sigma, \bar{\mu} + \sigma]$ for both of the flipping label and Byzantine attack. However, the reliability of benign clients in flipping attack are [0.0, 0.005] which indicates that these clients are not provide a good quality weight parameters. In our experiments without considering the statistical information from previous rounds, we observe that aggregation method in presence of label-flipping attack preforms better



(a) Flipping Label: FARel (b) Byzantine: FARel (c) Flipping: FARel_hist (d) Byzantine: FARel_hist

Figure 5.2: Range of Clients' Reliability on FMNIST dataset (10 clients, 30% malicious clients)

if the parameters sent by the clients with higher reliabilities are ignored. On the contrary, the performance of the model under the Byzantine attack is improved by removing the clients with the lower reliability.

Robustness The results of our experimental evaluation are shown in Table 5.1. We consider two cases with the static and dynamic pool of clients for previously described scenarios in which there are 30% malicious clients.

In this experiment we compare our robust aggregation methods with the state-of-the-art baselines. The variant of our aggregation methods are called FARel, FARel_adapt, FARel_hist. The FARel method is blindly aggregate the calculated reliability of clients. However, FARel_adapt ignores the clients provided data if their reliabilities are far from the others. Finally, FARel_hist considers statistical information of previous weight parameters to estimate clients' reliabilities.

- **Static Mode.**

  In this experiment, clients that participate at each round are fixed. The total number of clients are considered to be 10 in which 30% of them (i.e. 3 clients) are malicious ones. As it is shown in Table 5.1, FARel_adapt and FARel_hist are robust in most scenarios. As expected, Avg's performance is significantly affected under the presence of bad clients, especially in byzantine and flipping

Table 5.1: Aggregation Method Comparison in Static & Dynamic Mode (30% malicious clients)

Static Mode

Dataset	Attack	FedAvg	Median	Trim_mean	Krum	FARel	FARel_adapt	FARel_hist
CIFAR_10	Clean	70.25	**70.75**	**70.78**	57.75	68.05	69.74	69.75
	Byzantine	10.0	55.01	10.29	57.24	44.64	**59.66**	52.67
	Flip Label	51.37	41.34	46.74	10.0	10.0	**52.34**	51.10
	Noisy	67.51	**68.31**	68.22	57.67	67.22	67.64	67.80
FMNIST	Clean	**91.15**	90.95	91.05	87.79	91.05	91.05	91.07
	Byzantine	10.0	89.20	10.0	87.66	81.25	**90.62**	84.59
	Flip Label	79.05	77.58	73.23	10.0	14.55	80.38	**83.52**
	Noisy	99.25	99.20	99.32	94.78	94.09	97.74	**98.0**
MNIST	Clean	99.29	**99.31**	**99.34**	98.51	99.01	**99.3**	**99.32**
	Byzantine	11.35	98.18	11.35	97.43	91.35	98.21	**98.34**
	Flip Label	94.58	**97.80**	94.47	11.35	11.40	95.56	96.34
	Noisy	92.08	93.01	88.26	83.16	80.04	96.74	**96.82**

Dynamic Mode

Dataset	Attack	FedAvg	Median	Trim_mean	Krum	FARel	FARel_adapt	FARel_hist
CIFAR_10	Clean	69.22	**69.58**	68.22	56.69	67.87	**69.22**	67.25
	Byzantine	12.53	44.93	10.00	**61.49**	55.0	58.78	**60.56**
	Flip Label	10.0	35.00	10.07	10.32	11.56	**57.73**	55.53
	Noisy	63.27	63.35	61.18	61.36	61.67	63.43	**63.78**
FMNIST	Clean	91.68	92.00	88.26	89.79	91.79	91.98	91.87
	Byzantine	10.0	88.90	25.0	**90.36**	81.35	**89.85**	83.00
	Flip Label	10.0	68.23	10.25	11.04	11.35	70.93	**78.24**
	Noisy	89.08	88.12	86.13	81.12	89.24	90.01	**90.24**
MNIST	Clean	99.32	99.35	99.28	99.01	99.32	99.34	99.33
	Byzantine	11.35	97.05	10.01	96.37	96.27	97.07	94.38
	Flip Label	10.28	94.63	10.54	11.35	12.16	94.99	95.23
	Noisy	80.12	96.67	95.34	94.23	87.37	96.10	96.07

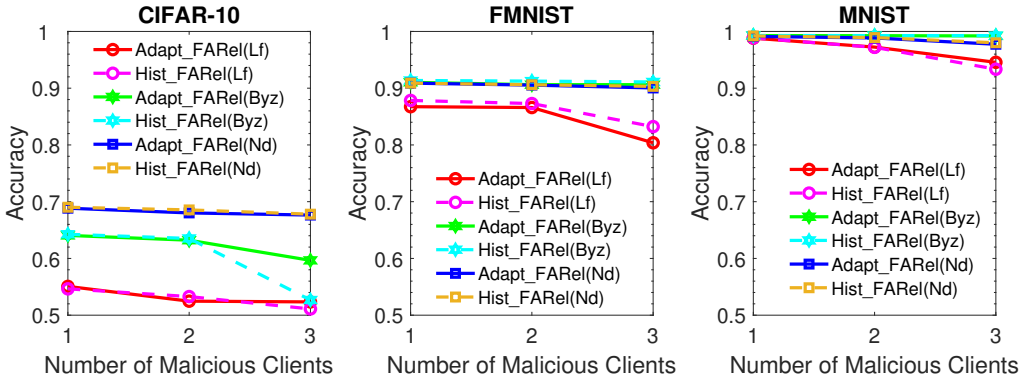


Figure 5.3: Effect of number of Malicious Clients

scenarios. It is also interesting to observe that both Krum and Median are very sensitive to label flipping attacks.

- **Dynamic Mode.** In this experiment, at each round, 10 clients are randomly selected from a pool of 100 clients consists of 30 malicious clients and 70 normal clients. The total number of clients participated in each round are considered to be 10. The result of this experiment is shown in table 5.1, we observe that FARel_adapt performs better we observe that in both dataset Krum performs better under Byzantine attack.

Impact of number of Malicious Clients We study the impact of the number of malicious clients on the proposed aggregation method. As it is shown in Fig.5.3, by increasing the number of malicious clients, the performance of global model slightly drops. It can be observed that FARel_hist improves upon FARel_adapt for FMNIST and MNIST datasets that have a higher accuracy on their clean data (i.e. no attack). However, in the CIFAR_10 dataset that has a poor performance on clean data, FARel_hist could not improve the performance.

Chapter 6

Conclusions and Future Work

Crowd-computing has numerous applications in various domains, including health, application reviews, traffic monitoring, and annotating detests. In this dissertation, we proposed methods to build robust aggregation model to handle strategically and intentionally manipulated data in such applications. Our solutions increase the robustness of these applications.

6.1 Summary

In Chapter3, we investigated the effectiveness of existing truth inference state-of-the-art methods to make a more knowledgeable decision for choosing the best method based on the domain. To this end, we conducted an extensive experimental evaluation of several state-of-the-art methods. Our selected methods included direct-based, PGM-based, and optimization approaches to estimate the true labels for binary or decision-making tasks. We carefully chose our test data sets from both real and synthetic data to conduct a fair and comprehensive evaluation. We also designed extensive experiments to show both the strengths and weaknesses of each method.

Chapter 4 proposed two solutions Edge-NN and Edge-PGM to improve the robustness of existing truth inference methods against data poisoning attacks in crowdsourcing systems. The proposed solutions provide a novel algorithm that applies matrix completion on a subset of tasks in which workers cannot reach a sufficiently

strong consensus. Also, it is combined with two enhancements to existing state-of-the-art inference methods by utilizing prior information. For the evaluation of our work, we applied a heuristic based attack and optimization based attack and our results confirm the effectiveness of our defense solution.

Chapter 5 investigated trustworthy solutions for federated learning based on the notions of truth inference method. First, we proposed two novel solutions to aggregate parameters without degrade the performance of the global model. Our solutions utilized truth inference method and outlier detection mechanisms to integrate provide clients parameter based on their reliability. We evaluated our methods extensively using real datasets. Our experiments on three common real-world datasets show that FARel_adapt and FARel_hist are robust to malicious clients with label flipping, noisy data and Byzantine attacks, while prior aggregation methods are not robust enough. This study focuses on IID distribution data among clients, future research could focus towards non-IID distribution.

6.2 Future Work

Possible future research directions for the proposed methods in this dissertation are presented as follows.

6.2.1 Extending robustness in crowdsourcing under data poisioning attack

The goal of our evaluation study [82] is to understand the resilience of existing leading-edge truth inference methods and ultimately build more robust systems. Towards this end, several directions for future work can be explored.

- Attack resistant truth inference: while existing methods provide certain level of resistance to data poisoning attacks, it remains an open question whether we can build more robust systems, e.g. by designing hybrid methods that combine

the strength of existing techniques (e.g. LAA-S against optimization based attack and targeted heuristic based attack and D&S against untargeted heuristic based attack) or designing entirely new techniques. A more fundamental question is whether we can have quantifiable guarantees for the robustness of the system against data poisoning attacks.

- Semi-supervised learning: All the truth inference approaches evaluated in this paper assume no access to the ground truth. This unsupervised nature makes the methods inherently susceptible to attacks. A semi-supervised approach [4, 83] may help provide resilience to attacks within which ground truth and workers' answers for historical tasks can be used for training the system.

- Dynamic behavior of workers: Our study is focused on a fixed set of workers and tasks. In practice, the workers could join and leave the system as they wish. They could also alter their behaviors dynamically in a strategic way to maximize the attack effect. Understanding the impact of such behaviors and building robust systems in a dynamic setting is also an important direction.

Deriving the true answer of tasks in crowdsourcing systems based on user-provided data is susceptible to data poisoning attacks, whereby malicious users may intentionally or strategically report incorrect information to mislead the system into inferring the wrong truth for a set of tasks. In our proposed methods [81] in chapter 4, we focused on binary tasks and proposed a solutions base on 1) detecting and augmenting the answers for the boundary tasks in which users could not reach a strong consensus and hence are subjective to potential manipulation, and 2) enhancing inference method with a stronger prior. As future work, our approach can be extended to take into account other types of attacks such as targeted attacks and further improvement on robustness, especially against strong white-box attacks.

6.2.2 Extending robustness in federated learning setting

Real-world federated learning environments are dealing with non-IID data environments. So far, most existing works mainly focus on IID distribution. The studies that focus on non-iid consider the global server's validation dataset to signal the model's performance. As one of the core parameters of federated learning is data distribution among clients. In our research, we considered data distributed identically among clients; we did not use any golden dataset in the global server. Therefore, we built a robust aggregation model to infer the reliability of each client. As a next step, we plan to extend our approach to consider non-id distribution by incorporating the golden validation set in the server-side.

Due to the multi-iteration/dynamic environment of federated learning compared to a simple crowdsourcing framework, taking advantage of reinforcement learning methods or the Kalman filter-based algorithm would help estimate the client's reliability with more confidence. Therefore, the next step towards the robustness of the federated learning system is expanding our work and integrating the aggregation algorithm with a reinforcement learning algorithm. This integration would learn how likely each client's parameter is used in training the general model.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] Muhammad Al-Qurishi, Mabrook Al-Rakhami, Atif Alamri, Majed Alrubaian, Sk Md Mizanur Rahman, and M Shamim Hossain. Sybil defense techniques in online social networks: a survey. *IEEE Access*, 5:1200–1219, 2017.

[3] Michael Anderson and Jeremy Magruder. Learning from the crowd: Regression discontinuity estimates of the effects of an online review database. *The Economic Journal*, 122(563):957–989, 2012.

[4] Kyohei Atarashi, Satoshi Oyama, and Masahito Kurihara. Semi-supervised learning from crowds using deep generative models. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[5] Multiple Authors. Twitter Sentiment.

[6] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643, 2019.

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

[8] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, pages 119–129, 2017.

[9] Thomas Bonald and Richard Combes. A minimax optimal algorithm for crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 4352–4360, 2017.

[10] Daren C Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence*, 14(1):75–90, 2008.

[11] Alice M Brawley and Cynthia LS Pury. Work experiences on mturk: Job satisfaction, turnover, and information sharing. *Computers in Human Behavior*, 54:531–546, 2016.

[12] Kenneth Bryan, Michael O'Mahony, and Pádraig Cunningham. Unsupervised retrieval of attack profiles in collaborative recommender systems. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 155–162. ACM, 2008.

[13] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. Uncovering large groups of active malicious accounts in online social networks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 477–488. ACM, 2014.

[14] Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):1–25, 2017.

[15] Paul-Alexandru Chirita, Wolfgang Nejdl, and Cristian Zamfir. Preventing shilling attacks in online recommender systems. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 67–74. ACM, 2005.

[16] Hongkyu Choi, Kyumin Lee, and Steve Webb. Detecting malicious campaigns in crowdsourcing platforms. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 197–202. IEEE Press, 2016.

[17] Peng Dai, Christopher H Lin, Daniel S Weld, et al. Pomdp-based control of workflows for crowdsourcing. *Artificial Intelligence*, 202:52–85, 2013.

[18] Georgios Damaskinos, El Mahdi El Mhamdi, Rachid Guerraoui, Arsany Hany Abdelmessih Guirguis, and Sébastien Louis Alexandre Rouault. Aggregathor: Byzantine machine learning via robust gradient aggregation. In *The Conference on Systems and Machine Learning (SysML), 2019*, number CONF, 2019.

[19] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.

[20] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478. ACM, 2012.

[21] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. Mechanical cheat: Spamming schemes and adversarial techniques on crowdsourcing platforms. In *CrowdSearch*, pages 26–30, 2012.

[22] John R Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[23] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Louis Alexandre Rouault. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, number CONF, 2018.

[24] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. *arXiv preprint arXiv:1911.11815*, 2019.

[25] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*, 2018.

[26] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. The limitations of federated learning in sybil settings. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2020)*, pages 301–316, 2020.

[27] Ujwal Gadiraju, Patrick Siehndel, Besnik Fetahu, and Ricardo Kawase. Breaking bad: Understanding behavior of crowd workers in categorization microtasks. In *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, pages 33–38, 2015.

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

[29] Alex Gaunt, Diana Borsa, and Yoram Bachrach. Training deep neural nets to aggregate crowdsourced responses. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence. AUAI Press*, page 242251, 2016.

[30] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples (2014). *arXiv preprint arXiv:1412.6572*.

[31] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[32] Rachid Guerraoui, Sébastien Rouault, et al. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, pages 3521–3530, 2018.

[33] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review*, 42(4):767–799, 2014.

[34] Chi Hong and Yichi Zhou. Label aggregation via finding consensus between models. *arXiv preprint arXiv:1807.07291*, 2018.

[35] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM, 2011.

[36] Nguyen Quoc Viet Hung, Duong Chi Thang, Matthias Weidlich, and Karl Aberer. Minimizing efforts in validating crowd answers. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 999–1014. ACM, 2015.

[37] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.

[38] M. J. Franklin J. Wang, T. Kraska and J. Feng. Crowderd. Crowdsourcing entity resolution. *PVLDB*, 5(11):1483–1494, 2012.

[39] Srikanth Jagabathula, Lakshminarayanan Subramanian, and Ashwin Venkataraman. Reputation-based worker filtering in crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 2492–2500, 2014.

[40] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and

countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE, 2018.

[41] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013.

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

[43] David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.

[44] Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In *Artificial Intelligence and Statistics*, pages 619–627, 2012.

[45] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[46] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[47] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[48] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[49] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[50] Brian Neil Levine, Clay Shields, and N Boris Margolin. A survey of solutions to the sybil attack. *University of Massachusetts Amherst, Amherst, MA*, 7:224, 2006.

[51] Guoliang Li, Jiannan Wang, Yudian Zheng, Ju Fan, and Michael J Franklin. Crowdsourcing background. In *Crowdsourced Data Management*, pages 11–20. Springer, 2018.

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

[53] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. 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*, pages 1187–1198. ACM, 2014.

[54] Shao-Yuan Li and Yuan Jiang. Multi-label crowdsourcing learning with incomplete annotations. In *Pacific Rim International Conference on Artificial Intelligence*, pages 232–245. Springer, 2018.

[55] Xiao Peng Li, Lei Huang, Hing Cheung So, and Bo Zhao. A survey on matrix completion: Perspective of signal processing. *arXiv preprint arXiv:1901.10885*, 2019.

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

[57] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *arXiv preprint arXiv:1909.11875*, 2019.

[58] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 2020.

[59] Jierui Lin, Min Du, and Jian Liu. Free-riders in federated learning: Attacks and defenses. *arXiv preprint arXiv:1911.12560*, 2019.

[60] Michael Luca. Reviews, reputation, and revenue: The case of yelp. com. *Com (March 15, 2016). Harvard Business School NOM Unit Working Paper*, (12-016), 2016.

[61] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 745–754, New York, NY, USA, 2015. ACM.

[62] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[63] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Aguera y Arcas. Federated learning of deep networks using model averaging. 2016.

[64] H Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data, april 2017. *URL https://ai.*

googleblog. com/2017/04/federated-learning-collaborative. html. Google AI Blog, 2018.

[65] Chenglin Miao, Qi Li, Lu Su, Mengdi Huai, Wenjun Jiang, and Jing Gao. Attack under disguise: An intelligent data poisoning attack mechanism in crowdsourcing. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 13–22. International World Wide Web Conferences Steering Committee, 2018.

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

[67] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Jeff J Sandvig. Attacks and remedies in collaborative recommendation. *IEEE Intelligent Systems*, 22(3):56–63, 2007.

[68] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 2020.

[69] Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125*, 2019.

[70] An Thanh Nguyen, Byron C Wallace, and Matthew Lease. A correlated worker model for grouped, imbalanced and multitask data. In *UAI*, 2016.

[71] Michael O'Mahony, Neil Hurley, Nicholas Kushmerick, and Guénolé Silvestre. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology (TOIT)*, 4(4):344–377, 2004.

[72] Shashank Rajput, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Detox: A redundancy-based framework for faster and more robust gradient aggregation. In *Advances in Neural Information Processing Systems*, pages 10320–10330, 2019.

[73] Vikas C Raykar and Shipeng Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *Journal of Machine Learning Research*, 13(Feb):491–518, 2012.

[74] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(Apr):1297–1322, 2010.

[75] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, pages 6106–6116, 2018.

[76] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[77] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[78] Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336, 2005.

[79] Gianluca Stringhini, Pierre Mourlanne, Gregoire Jacob, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. {EVILCOHORT}: Detecting communities

of malicious accounts on online services. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*, pages 563–578, 2015.

[80] Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 1299–1316, 2018.

[81] F. Tahmasebian, L. Xiong, M. Sotoodeh, and V. Sunderam. Edgeinfer: Robust truth inference under data poisoning attack. In *2020 IEEE International Conference on Smart Data Services (SMDS)*, pages 45–52, Los Alamitos, CA, USA, oct 2020. IEEE Computer Society.

[82] Farnaz Tahmasebian, Li Xiong, Mani Sotoodeh, and Vaidy Sunderam. Crowdsourcing under data poisoning attacks: A comparative study. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 310–332. Springer, 2020.

[83] Wei Tang and Matthew Lease. Semi-supervised consensus labeling for crowdsourcing. In *SIGIR 2011 workshop on crowdsourcing for information retrieval (CIR)*, pages 1–6, 2011.

[84] Amol Vasudeva and Manu Sood. Survey on sybil attack defense mechanisms in wireless ad hoc networks. *Journal of Network and Computer Applications*, 120:78–118, 2018.

[85] Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. Community-based bayesian aggregation models for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*, pages 155–164. ACM, 2014.

[86] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y Zhao. You are how you click: Clickstream analysis for sybil detection.

In *Presented as part of the 22nd {USENIX} Security Symposium ({USENIX} Security 13)*, pages 241–256, 2013.

[87] Gang Wang, Bolun Wang, Tianyi Wang, Ana Nika, Haitao Zheng, and Ben Y Zhao. Defending against sybil devices in crowdsourced mapping services. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, pages 179–191. ACM, 2016.

[88] Gang Wang, Tianyi Wang, Haitao Zheng, and Ben Y Zhao. Man vs. machine: Practical adversarial detection of malicious crowdsourcing workers. In *USENIX Security Symposium*, pages 239–254, 2014.

[89] Qing-Xian Wang, Yan Ren, Neng-Qiang He, Meng Wan, and Guo-Bo Lu. A group attack detecter for collaborative filtering recommendation. In *2014 11th International Computer Conference on Wavelet Actiev Media Technology and Information Processing (ICCWAMTIP)*, pages 454–457. IEEE, 2014.

[90] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010.

[91] Zaiwen Wen, Wotao Yin, and Yin Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4(4):333–361, 2012.

[92] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[93] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[94] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.

[95] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):2, 2014.

[96] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates supplementary material.

[97] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. *arXiv preprint arXiv:1803.01498*, 2018.

[98] Li'ang Yin, Jianhua Han, Weinan Zhang, and Yong Yu. Aggregating crowd wisdoms with label-aware autoencoders. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1325–1331. AAAI Press, 2017.

[99] Haifeng Yu, Michael Kaminsky, Phillip B Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. *ACM SIGCOMM Computer Communication Review*, 36(4):267–278, 2006.

[100] Haifeng Yu, Chenwei Shi, Michael Kaminsky, Phillip B Gibbons, and Feng Xiao. Dsybil: Optimal sybil-resistance for recommendation systems. In *2009 30th IEEE Symposium on Security and Privacy*, pages 283–298. IEEE, 2009.

[101] Dong Yuan, Guoliang Li, Qi Li, and Yudian Zheng. Sybil defense in crowdsourcing platforms. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1529–1538. ACM, 2017.

[102] Fuzhi Zhang, Zening Zhang, Peng Zhang, and Shilei Wang. Ud-hmm: An unsupervised method for shilling attack detection based on hidden markov

model and hierarchical clustering. *Knowledge-Based Systems*, 148:146–166, 2018.

[103] Kuan Zhang, Xiaohui Liang, Rongxing Lu, and Xuemin Shen. Sybil attacks and their defenses in the internet of things. *IEEE Internet of Things Journal*, 1(5):372–383, 2014.

[104] Yongfeng Zhang, Yunzhi Tan, Min Zhang, Yiqun Liu, Tat-Seng Chua, and Shaoping Ma. Catch the black sheep: unified framework for shilling attack detection based on fraudulent action propagation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

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

[106] Yudian Zheng, Guoliang Li, and Reynold Cheng. Docs: a domain-aware crowdsourcing system using knowledge bases. *Proceedings of the VLDB Endowment*, 10(4):361–372, 2016.

[107] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552, 2017.

[108] Dengyong Zhou, Sumit Basu, Yi Mao, and John C Platt. Learning from the wisdom of crowds by minimax entropy. In *Advances in neural information processing systems*, pages 2195–2203, 2012.

[109] Wei Zhou, Junhao Wen, Yun Sing Koh, Qingyu Xiong, Min Gao, Gillian Dobbie, and Shafiq Alam. Shilling attacks detection in recommender systems based on target item analysis. *PloS one*, 10(7):e0130968, 2015.

[110] Yao Zhou and Jingrui He. Crowdsourcing via tensor augmentation and completion. In *IJCAI*, pages 2435–2441, 2016.

[111] Yao Zhou, Lei Ying, and Jingrui He. Multic2: an optimization framework for learning from task and worker dual heterogeneity. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 579–587. SIAM, 2017.