Solution Probing Attack Against Blockchain-based Quality-aware Crowdsourcing Platforms

Heng Wang, Yunlong Mao, Sheng Zhong
State Key Laboratory for Novel Software Technology, Nanjing University
Nanjing, China

***Abstract*—Conventional crowdsourcing platforms primarily rely on a central server as the broker for information exchange. Although many efforts have been made, centralized platforms are still vulnerable to underlying security issues such as the trusted central server and single-point failure. Fortunately, blockchain has emerged as an alternative infrastructure for building crowdsourcing platforms. Many excellent designs of decentralized crowdsourcing platforms (DCSPs) built atop blockchain have been proposed recently. Benefiting from blockchain, DCSPs can provide fascinating features, like tampering resistance and anonymity. However, new security issues keep cropping up. While existing studies have proved that DCSPs are vulnerable to various attacks, our study has identified a new attack named solution probing attack against DCSPs. The adversary of solution probing attack can take advantage of the anonymity of DCSPs to probe valid solutions using a generative model. Due to the transparency of blockchain transactions, our probing attack is effective even if the solutions are encrypted. We evaluate the probing attack on large-scale crowdsourcing tasks. Experimental results show that the adversary is capable of rewarding fraud without any meaningful contribution to the crowdsourcing task. To protect DCSPs from the probing attack, we propose a defense solution based on a mix-and-match strategy. Defense evaluation results show that our solution can defeat the probing attack effectively.**

***Index Terms*—crowdsourcing security, decentralized platform, solution quality evaluation, probing attack**

I. Introduction

Crowdsourcing has contributed tremendously to traditional industries. Applications of crowdsourcing like question answering [1], fraud detection [2], and ride sharing [3] have changed the way people live. One major application of crowdsourcing is data collection and labeling for large amounts of data in areas such as machine learning and big data analysis. However, solutions from a single worker is not reliable due to individual bias or lack of expertise. To tackle this issue, a common practice for requesters is to collect aggregated solutions from multiple workers. Then, high-quality information is extracted from those aggregated solutions with truth discovery algorithms [4]–[7].

Conventional crowdsourcing platforms are designed in a centralized fashion. A broker with global information can allocate crowdsourcing jobs to proper workers efficiently [8]. Given a specific set of constraints such as computing resources, job requirements, and so on, centralized crowdsourcing platforms can provide the optimal (or approximately optimal) task allocations. However, a centralized crowdsourcing platform requires a trusted third party to act as a broker, which in practice, is hardly available and vulnerable to single-point failure, DDoS attack, and Sybil attack [9]. Consequently, redesigning crowdsourcing platforms in a decentralized manner becomes a natural demand. Fortunately, the emergence of blockchain offers a promising decentralized solution for the design of crowdsourcing platform. Some recent studies have suggested that excellent decentralized crowdsourcing platforms [10]–[14] are often built atop well-developed blockchains, such as Ethereum.

Decentralized crowdsourcing platforms (DCSPs) on the one hand, solve the problems of centralized platforms. But on the other hand, bring new challenges. Since blockchain uses distributed peers as a public ledger, task solutions submitted by workers will be publicly accessible, resulting in confidential data leakage. Several studies such as [14], have proposed that we can encrypt the solutions before storing in the ledger. However, encrypted solutions are not suitable for real-time quality evaluation and fair rewarding proof. To tackle this problem, recent studies have recommended a solution combining zero-knowledge proof with solution encryption [11]. Unfortunately, other security issues still exist in DCSPs. Even though both workers and requesters use pseudo-anonymity for trading, the linkage between workers and requesters will be permanently recorded in the ledger. Coin mixing techniques [15] for blockchain can mitigate this linkage issue. However, recent studies [16] have revealed that the relationship between anonymous users can still be inferred by learning transaction graph knowledge.

Moreover, we have identified a new security vulnerability in DCSPs. When a DCSP is implemented atop blockchain, a publicly shared ledger is essential for each task. Although task solutions can be recorded in cipher text, reward transactions between the requester and workers cannot be concealed in the ledger. Based on observations on existing DCSPs designs, we identify a new threat named solution probing attack. In particular, an adversary can take advantage of the anonymity of blockchain to repeatedly submit generated solutions with the goal of defrauding the requester. We note that solutions are not forged randomly. The amount of a reward is highly relevant to the quality of a solution. Hence, the adversary can collect reward transactions from the ledger and take them as training data to build a generative model, capturing the hidden relation between rewards and solution quality. In this way, the adversary can forge valid solutions after necessary probing without doing the actual work.

We evaluate the feasibility of the solution probing attack with three state-of-the-art truth discovery algorithms, Gaussian truth model (GTM) [6], conflict resolution on heterogeneous data (CRH) [5] and PACE [4], on both synthetic and real-world datasets widely used in crowdsourcing studies. For example, one real-world dataset called Weather contains weather information of a city from different sources in the timestamp, which can be viewed as different workers collecting weather information for a city in a task. Our experiments present that the solution probing attack can generate high-quality solutions for all three truth discovery algorithms, even if the adversary is not qualified for the task, e.g., located in New York city but forging the real-time weather in Ushuaia.

To mitigate the solution probing attack, we propose a defense based on a mix-and-match strategy to prevent the adversary from obtaining reward information. The intuition of the defense is that the reward information can be preserved by adding appropriated random masks, which require workers to participate in multiple companion tasks to offset added masks. Our experimental results reveal that once the defense is involved, the rewards of generated solutions by the adversary are decreased significantly.

Overall, our contribution are three-fold:

- We have identified a new type of attack against DCSPs named solution probing attack, in which the adversary obtains rewards with generated solutions.
- We implement this attack by using a generative neural network model and confirm its effectiveness by simulating experiments on both synthetic and real-world datasets in three state-of-the-art truth discovery algorithms. This attack indicates that rewards in quality-aware crowdsourcing platforms may be the exploited vulnerability.
- To defeat the solution probing attack, we propose a defense based on a mix-and-match strategy to preserve the reward information of solutions. Experimental results show that the proposed defense effectively decrease rewards obtained by the adversary from generated solutions.

In the remaining of this paper, we first reviewed related works in Section II, then discussed system and threat model, formulated the problem in Section III. The proposed attack and its optimizations are introduced in Section IV and Section V conducts experiments on the proposed attack. Section VI proposes the mix-and-match defense and its experiment results. Finally, Section VII concludes the paper.

II. Related Work

A. Blockchain Crowdsourcing Platform

The emergence of blockchain infrastructure offers a promising solution for DCSP designs. Some recent studies have proposed excellent DCSPs [10], [11], [13], [14] atop well developed blockchain. In two famous DCSPs, CrowdBC [14] and ZebraLancer [11], coordination of workers and requesters, such as task match, worker selection and rewarding are accomplished by self-executing smart contracts in form of transactions. In [13], smart contracts along with ciphertext policy attribute based encryption are leveraged to build a DCSP with fine-grained authorization for data trading. NF-Crowd [10] proposes a protocol that reduces the lower bound of transaction fees of the underlying blockchain to $\mathcal{O}(1)$ regards to the number of participants of a task. Besides, blockchain is tamper resistance, which makes it easier to trace out the source of malicious behaviors [17]. Despite the benefits of DCSPs, the attack proposed in this paper is effective due to the exploitation of stored reward transactions in the underlying blockchain.

B. Quality-Aware Crowdsourcing

Quality-aware crowdsourcing refers to crowdsourcing platforms that reward workers according to solution quality. This strategy is beneficial for the platform, in the way that it increases its reliability; it is also beneficial for requesters, who obtain high-quality result from the task; and it is also beneficial to workers, who obtain more rewards by providing high-quality data [4], [7]. Therefore, it is necessary for the platform to utilize truth discovery algorithms to estimate the truth from the aggregated data and evaluate their corresponding quality. In [4], [5], the truth estimation problem is modeled as an optimization problem, in which the truth is the value that minimizes the distance from all data. The quality of each data is then measured by their distance from the estimated truth because the higher the quality, the closer it is to the truth. Besides, Bayesian probabilistic model is also leveraged for estimating the truth and qualities of workers, in which the expectation maximization algorithm is utilized to update the estimated truth and qualities [6]. We note that the attack proposed in this paper is effective both for the optimization and the probabilistic model of truth and quality estimation algorithms.

C. Crowdsourcing Attack

In crowdsourcing, an attack can be launched by requesters, such as *false-reporting attack*, *clogging attack*, or by workers, such as *free-riding attack*, *data poisoning attack* [11], [14], [18]–[21]. In the false-reporting attack, requesters misreport the quality of the solutions in order to reduce the reward owed, or claim they have not received the solutions [14], [19]. In [19], reputation mechanisms are employed for managing workers and requesters to defend against the false-reporting attack. In DCSPs, smart contracts can be utilized to automatically reward workers from deposits provided by the requester according to pre-defined reward polices [11], [14]. In the clogging attack, requesters publish fake tasks to drain the resources of the workers, especially in mobile crowdsourcing [18]. In the free-riding attack, workers obtain rewards exerting little or no effort in the task, e.g. by submitting random noise. To tackle the free-riding attack, workers are selected and rewarded according to their reputations or quality [19] or they are required to deposit to smart contracts before participating in a task in DCSPs, then retrieve the deposition if they submit high-quality solutions [11], [14]. A *common-prefix-linkable* scheme is proposed in [11] to detect malicious workers who repeatedly submit solutions to the same task, but it requires a trusted third party for authentication. In the data poisoning attack, the adversary intentionally forges data to deviate the estimated truth [20], [21]. To defend against data poisoning attack, median-of-weighted-average and maximum influence estimation are leveraged to mitigate the influence of the forged data [20], [21]. We note that the proposed solution probing attack is different from the free-riding attack since the solution probing attack aims at quality-aware crowdsourcing tasks, which are tricky for free-riding attackers. On the other side, we note that the proposed probing attack tries to affect the estimated truth as little as possible, which is totally different from the data poisoning attacks.

III. Model and Problem Formulation

This section first presents participants of DCSPs and the formulas of truth discovery algorithms. Then threat model of the proposed solution probing attack is introduced, which contains the goal and the capacity of the adversary.

A. System Model

The system model elaborates the characteristics and characters of DCSPs.

1) ***Characteristics of Decentralized Crowdsourcing Platforms:*** A DCSP is built atop blockchain, which is constructed by a list of data blocks with each block chained to its previous block by some cryptographic technique except the genesis block to prevent the structure from being tampered with. In general, blockchain can be deemed as a public ledger in which every participant communicates with others by broadcasting transactions. Besides, blockchain also supports smart contracts, which contains a property that certain instructions can be self-executed once pre-defined conditions are satisfied. The self-execution property of the smart contract can be leveraged to automatically reward workers in DCSPs [11], [14].

One of the features of DCSPs is the public ledger, which invokes that every participant is able to view the content of blocks. Therefore, participants other than the requester are able to the view solutions of a task, which can be exploited by the adversary to launch attacks. In order to tackle the aforementioned issue, workers are demanded to encrypt solutions with the public key of the requester before submission [11], [14]. However, the solution probing attack proposed in this paper is effective with encrypted solutions.

2) ***System Model:*** DCSPs intend to facilitate the bargain of crowd knowledge just as in its centralized counterparts. We consider a scenario where each task contains multiple subtasks. In each subtask, workers are required to submit solutions related to the task, which is a common practice to handle the lack of prior knowledge [21]. There are four roles in a decentralized crowdsourcing paradigm, which are *requester*, *worker*, *crowdsourcing platform* and *miner* respectively.

- *Requester.* A requester is commonly an individual or an organization who hires the crowd to finish a predefined task. The task usually requires some specific conditions or human intelligence to be accomplished, such as environment sensing or data entry annotation. To motivate the crowd to join the task, the requester should provide a proper incentive.
- *Worker.* A worker can be any individual with the necessary requirements. If a worker is interested in a published task, the former can try to earn the reward by finishing the mentioned task according to the specified requirements in the description. By submitting a valid solution to the requester, the worker becomes a legal candidate for reward.
- *Crowdsourcing platform.* A DCSP is implemented by blockchain to act as a manager of requesters, workers, and tasks. Generally, the crowdsourcing platform should be in charge of solution quality checks since the quality of solutions is crucial to the requester.
- *Miner.* A miner is responsible for maintaining the underlying blockchain by generating blocks to store data and validating them in the network. The miner will receive monetary or service reward by maintaining the blockchain backbone.

It is worth mentioning that the above roles are not fixed and disjoint. For example, a miner can also be a worker and converts to a requester when it publishes a crowdsourcing task.

$$\begin{aligned}\max_{\mathcal{M},\Sigma} f(\mathcal{M},\Sigma) = &-\sum_{s\in\mathcal{S}}\left(2(\alpha+1)\log\sigma_s+\frac{\beta}{\sigma_s^2}\right)\\ &-\sum_{e\in\mathcal{E}}\frac{(\mu_e-\mu_0)^2}{2\sigma_0^2}-\sum_{e\in\mathcal{E}}\sum_{c\in\mathcal{C}_e}\left(\log\sigma_{s_c}+\frac{(o_c-\mu_e)^2}{2\sigma_{s_c}^2}\right).\end{aligned} \tag{1}$$

$$\begin{aligned}&\min_{\mathcal{X}^*,\mathcal{W}} f\left(\mathcal{X}^*,\mathcal{W}\right)=\sum_{k=1}^{K} w_k\sum_{i=1}^{N}\sum_{m=1}^{M} d_m\left(v_{im}^*,v_{im}^k\right),\\ &s.t.\quad \sum_{k=1}^{K} exp\left(-w_k\right)=1.\end{aligned} \tag{2}$$

$$q_i=\frac{\frac{1}{\theta_i+\epsilon}}{\sum_i\frac{1}{\theta_i+\epsilon}}. \tag{3}$$

The truth discovery algorithms considered in this paper are GTM, CRH and PACE, which are based on different models and assumptions. The GTM algorithm is based on the probabilistic model with the goal of maximize (1), in which $\mathcal{M}$ is the estimated truth value of each data, Σ is the variance of each data, $\mathcal{S}$ denotes the participating workers, $\mathcal{C}_e$ is the dataset of e-th dimension, o_c is the normalized data, μ_0 and σ_0 is the prior knowledge of the task, α and β are hyperparameters of the model. The variance of each data is negatively correlated with data quality. While in the CRH algorithm, the goal is to minimize (2), in which w_k is the quality of k-th data, v_{im}^k is the k-th data for i-th task of m-th dimension, v_{im}^* is the estimated truth value of the data, $d(\cdot,\cdot)$ denotes the distance function. The constraint of (2) is to ensure the practicality of the solutions. In PACE, the truth is estimated as the centroid χ of all the data, then the deviation of each data from the centroid is computed as $\theta_i=|d(\chi,d_i)|$, in which d_i denotes i-th data, finally the quality is calculated as in (3).

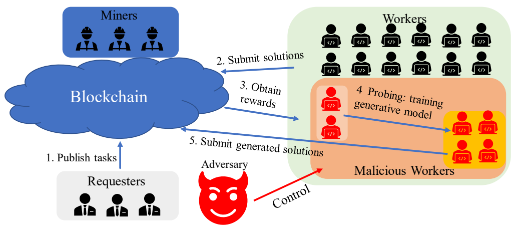

Fig. 1. The threat model of the solution probing attack.

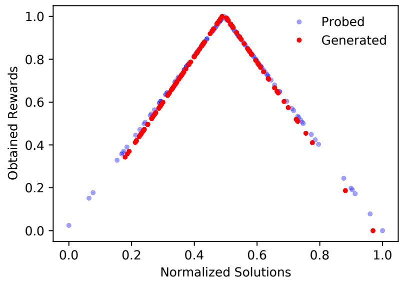

Fig. 2. The probed and generated solutions and their corresponding rewards.

B. Threat Model

This section describes the goal and the capacity of the adversary in the proposed solution probing attack. We give an illustration of the threat model in Fig. 1.

*1) **The Goal of the Adversary**:* The goal of the adversary is to maximize its rewards from DCSPs by submitting generated solutions. This attack is motivated by the development of generative models and is harmful for both the requesters and the platform. For instance, due to the lack of human intelligence, generated solutions do not take the scenario of the task into consideration, which is catastrophic in some cases such as real-time navigation or traffic monitoring. The fact that the adversary obtains high rewards without exerting efforts to the task also disincentivizes workers to provide high-quality solutions, thereby reducing the reliability of the platform.

*2) **The Capacity of the Adversary**:* In the solution probing attack, solutions can be categorized into two types: solutions with efforts (we call them honest solutions as they are submitted by honest workers) and generated solutions. The goal of the adversary is to maximize rewards obtained from the platform with the generated solutions. To achieve this goal, we hypothesized that the adversary has the ability to first probe *partial* honest solutions and their rewards (we call them solution-reward pairs for simplicity). It achieves it by bribing other workers or submitting honest solutions itself by multiple anonymous identifiers. Afterwards a generative model is trained with the probed solution-reward pairs. After completing the training process of the model, the adversary generates solutions from the model and submits those solutions to the platform for the remaining tasks.

C. Solution Probing Attack Formulation

To state formally, for a task T, the proposed attack contains two processes: *probing* and *attack*. In the probing process, the number of honest workers participating in T is $\mathcal{N}$ and the adversary $\mathcal{A}$ is able to probe N solution-reward pairs. In the attack process, by investigating the linkage between solutions and their rewards from the probed solution-reward pairs, $\mathcal{A}$ can heuristically generate and submit new solutions with new identifiers. For better performance of the attack, $\mathcal{A}$ builds a generative model $\mathcal{G}$ to generate solutions. After probing N solution-reward pairs $\mathbb{D}$ from DCSPs, $\mathcal{A}$ trains $\mathcal{G}$ to generate $\mathcal{M}$ solutions similar to valid solutions. For T, if the average rewards of the generated solutions by $\mathcal{A}$ are higher than the honest workers, we say that the probing attack is successful. Otherwise $\mathcal{A}$ is able to obtain more rewards by solving T with efforts. Therefore, the goal of $\mathcal{A}$ is to maximize the average rewards from DCSPs, which can be formulated as follows:

$$\underset{d_g \leftarrow \mathcal{G}(\mathbb{D})}{\text{maximize}} \frac{1}{\mathcal{M}} \sum_{d_{gi} \in d_g} \mathcal{R}(d_{gi}) \tag{4}$$

where d_g denotes solutions generated by $\mathcal{G}$, $\mathcal{R}$ is the reward policy of the crowdsourcing task.

More specifically, the adversary exploits generative adversarial networks (GAN) [22] as the generative model, which involves two deep neural networks, a generative model and a discriminative model competing against each other. In a GAN, the generative model learns to generate solution-reward pairs and the discriminative model learns to differentiate the probed solution-reward pairs from the generated. To prove genericness of the solution probing attack, the adversary attacks DCSPs with different privacy preserving methods and truth discovery algorithms in the same GAN configuration.

IV. Proposed Attack: Solution Probing Attack

In this section, we present the solution probing attack on DCSPs by introducing in turn the overview, a strawman attack algorithm, two optimizations for improving the effectiveness of the proposed attack and in the case where workers are demanded to submit encrypted solutions.

A. Overview

Benefiting from the anonymity of the underlying blockchain, DCSPs offer anonymous submissions for workers. Although anonymity can apparently preserve the identity of a worker, the linkage between solutions and rewards are still exists due to the public ledger feature of the blockchain. Therefore, an adversary $\mathcal{A}$ can generate the required solutions of a task based on the linkage between the probed solutions and rewards to reap rewards. To maximize reaped rewards, $\mathcal{A}$ exploits a trained generative model with the probed solution-reward pairs to generate solutions similar to valid solutions.

To prove the feasibility of the solution probing attack, we demonstrate the attack using a synthetic dataset which requires 256 continuous numeric data as solutions. $\mathcal{A}$ probes half of the solutions and generates another half. For proof of concept, probed solutions are generated randomly following a Gaussian distribution. Valid solutions should be within a truncated interval. The data samples submitted will be evaluated by comparing to the thresholds of the interval. The closer it is to the interval, the higher the quality will be, leading to more rewards. During the probing phase, $\mathcal{A}$ probes 128 solution-reward pairs as subsequent training data. During the attack phase, $\mathcal{A}$ trains a generative adversarial network (GAN) with the probed training data as the generative model to generate solutions. Fig. 2 presents the probed and generated solutions along with rewards, all of which are normalized using min-max normalization. $\mathcal{A}$ generates the remaining 128 solutions, as we can see from the figure, most of the generated solutions obtain rewards not less than 0.4, which means the pattern of the solutions and their rewards are learned by $\mathcal{A}$. Therefore, $\mathcal{A}$ is capable of reaping rewards from the platform with generated solutions.

B. Solution Probing Attack

We have introduced a high-level idea of the solution probing attack and demonstrated that the adversary is capable of obtaining rewards without exerting efforts to perform the task. In this section, we present the detailed procedures of the solution probing attack.

Algorithm 1 Strawman Solution Probing Attack

```
Require: The generative model $\mathcal{G}$; probed solution-reward
    pairs $\mathbb{D} = ((d_1, cash_1), \dots, (d_N, cash_N))$; the number of
    solutions to generate $\mathcal{M}$.
Ensure: $d_g$ is the generated solutions.
1: Configure training parameters of $\mathcal{G}$.
2: while threshold is not satisfied do
3:     Train $\mathcal{G}$ with observed dataset $\mathbb{D}$.
4: end while
5: Leverage $\mathcal{G}$ to generate $\mathcal{M}$ solutions values $d_g$.
6: return $d_g$
```

1) Strawman Attack: Let $\mathcal{N}$ denote the number of honest workers of a task T, $\mathcal{D} = ((d_1, cash_1), \dots, (d_{\mathcal{N}}, cash_{\mathcal{N}}))$ represents solution-reward pairs of T, $\mathbb{D} = ((d_1, cash_1), \dots, (d_N, cash_N))$ denotes the solution-reward pairs probed by the adversary $\mathcal{A}$ from $\mathcal{D}$, $\mathcal{M}$ denotes the number of the generated solutions. The procedure of the strawman attack is presented in Algorithm 1. The idea of the algorithm is straightforward: the adversary feeds the probed solution-reward pairs to a generative model $\mathcal{G}$, which $\mathcal{A}$ exploits to generate solutions. As demonstrated in section IV-A, the solutions generated by $\mathcal{A}$ is capable of reaping rewards. However, $\mathcal{A}$ is not guaranteed to obtain high-quality solutions due to the relatedness of the solutions and rewards is not reflected in the process of training $\mathcal{G}$.

2) Optimizations: Although the strawman solution probing attack is capable of generating solutions that reap rewards from DCSPs, it fails to capture the relatedness of the probed solutions and rewards. To further improve the strength of $\mathcal{G}$, $\mathcal{A}$ utilizes two optimization strategies for model training and solutions generation, which are elaborated below.

Algorithm 2 Solution Probing Attack

```
Require: The generative model $\mathcal{G}$; probed solution-reward
    pairs $\mathbb{D} = ((d_1, cash_1), \dots, (d_N, cash_N))$; the number of
    solutions to generate $\mathcal{M}$; the base Bootstrapping times $B$.
Ensure: $d_g$ is the generated solutions.
1: Set $\mathbb{T} \leftarrow \mathbb{D}$.
2: for $(d_i, cash_i)$ in $\mathbb{D}$ do
3:     Compute the sample times $\mathcal{B}$ according to $B$ and $cash_i$.
4:     for $i = 0$ to $\mathcal{B}$ do
5:         $\mathbb{T} \leftarrow \mathbb{T} \cup (d_i, cash_i)$.
6:     end for
7: end for
8: Configure training parameters of $\mathcal{G}$.
9: while threshold is not satisfied do
10:    Train $\mathcal{G}$ with $\mathbb{T}$.
11: end while
12: $\mathcal{G}$ generates $D_g = ((d'_1, cash'_1), \dots, (d'_M, cash'_M))$ where
    $M$ is greater than $\mathcal{M}$.
13: Sort $D_g$ according to $cash'_i$ in descending order.
14: Select top $\mathcal{M}$ pairs $p_g$ from $D_g$.
15: Select solutions from $p_g$ as $d_g$.
16: return $d_g$
```

Algorithm 2 summarizes the optimized solution probing attack. The purpose of the optimization strategies is to direct $\mathcal{G}$ to generate high quality solutions.

First, in order to learn the pattern of high-quality solutions from the probed solution-reward pairs, Bootstrapping [23] is leveraged to increase the quality of the generated solutions, which utilizes re-sampling to estimate the distribution of some specified random variables. In short, Bootstrapping first independently samples from observed data with replacement and then estimates the distribution based on total samples obtained. Borrowing the re-sampling idea from Bootstrapping, the adversary augments the probed solution-reward pairs according to the reward: the higher reward of a solution, the more this solution is sampled. After Bootstrapping, $\mathcal{G}$ has a higher chance to learn the pattern of high-quality solutions.

Second, noting that $\mathcal{A}$ fails to capture the relatedness of the probed solutions and their rewards in the strawman attack, which contains quality information of the probed solutions. Therefore, instead of training a generative model that directly generates desired solutions, a generative model with two output is trained, which corresponds to the generated solutions and rewards respectively. Although the reward generated from the model is not the real reward from DCSPs, it still provides an indication of the expected rewards to the corresponding generated solutions. Therefore, once the optimized model is trained from the probed solution-reward pairs, the adversary first generates more than $\mathcal{M}$ solution-reward pairs, then sorts those pairs according to the reward in descending order. Finally, the top $\mathcal{M}$ solutions from the sorted pairs are submitted to the DCSP.

Algorithm 3 Solution Probing Attack in Privacy Preserving DCSPs

```
Require: The regression model $\mathcal{R}$; the probed dataset $\mathbb{D} = ((d_1, cash_1), \ldots, (d_N, cash_N))$; the remaining rewards $\mathcal{C} = (cash_1, \ldots, cash_{\mathcal{N}-N})$.
Ensure: $d_g$ is the generated solutions.
1: Configure training parameters of $\mathcal{R}$.
2: while threshold is not satisfied do
3:     Train $\mathcal{R}$ with observed data $\mathbb{D}$.
4: end while
5: Derive solutions $D \leftarrow \mathcal{R}(\mathcal{C})$.
6: Generate probed dataset $\mathbb{D} = \mathbb{D}||(D, C)$.
7: Call Algorithm 2 with $\mathbb{D}$ and assign the result to $d_g$.
8: return $d_g$.
```

3) Privacy Preserving DCSPs: Due to the public ledger feature of blockchain, every participant has directly access to solution-reward pairs, violating the privacy requirements of requesters. To tackle this problem, existing DCSPs demand workers to encrypt solutions using the public key of the requester before submission [11], [14], which renders the solution probing attack harder since there is a reduction in the number of solution-reward pairs the attacker can probe. However, rewards of each worker are still public to participants of DCSPs, which can be exploited by the adversary to enhance the attack effectiveness. Notice that rewards are correlated with solutions, thus the attacker is able to derive solutions from the probed rewards.

To derive solutions from rewards, the attacker first trains a regression model $\mathcal{R}$ with $\mathbb{D}$ to capture the correspondence of rewards and solutions, then derives solutions from the trained regression model with rewards $\mathcal{C} = (cash_1, \ldots, cash_{\mathcal{N}-N})$ probed from DCSPs. After collecting and deriving solution-reward pairs $\mathbb{D}$, the attacker launches the optimized solution probing attack with the obtained solution-reward pairs. Algorithm 3 summaries the attack procedure.

V. Solution Probing Attack Experiments

To demonstrate the effectiveness of the solution probing attack, we evaluate it under both threshold and quality related reward policy in plain and privacy preserving DCSPs with varying probing and attack sizes. For the threshold reward policy, once the quality of a solution satisfies the pre-defined conditions, workers will obtain the same amount of rewards; while for the quality related policy, the higher the quality, the more the reward. Besides, three state-of-the-art truth discovery algorithms, GTM, CRH and PACE, are utilized for calculating quality of the solutions. Workers submit solutions without any modification in plain DCSPs while submit encrypted solutions in privacy preserving DCSPs.

A. Experiment Setup

1) Datasets: To demonstrate the effectiveness of the proposed solution probing attack, we have conducted experiments on both synthetic and real-world *continues numeric values* datasets, which contain different ranges of numeric values and skewness to simulate the variety of crowdsourcing tasks.

The synthetic dataset is first harnessed to demonstrate the effectiveness of the solution probing attack, which contains 256 values $d \sim N(\mu, \sigma^2)$ where μ sampled uniformly from $[10, 30)$ and σ sampled uniformly from $[0, 30)$ correspondingly. The Gaussian distribution is chosen for its wide occurrence in practice, but it is easy for the adversary to find out the pattern once it probes partial data due to the synthetizing strategy. Two benchmark real-world datasets are explored in our experiments. The first dataset is *Weather* [24], which contains weather information of 30 major USA cities from 18 sources. The second dataset is *Emotion* [25], containing numeric assignment from workers to describe their feelings towards some text documents. For the Weather dataset, we choose data from *San Jose* on a given day and for the Emotion dataset, we choose data from *Disgust* and *Valence*, which contain values ranging from $[0, 100]$ and $[-100, 100]$ correspondingly. Those data are explored because they contain the most number of data and are representative among datasets in value range and skewness. The amount of data in each datset is 256, 284, 580, 40 for Synthetic, Weather, Emotion Disgust and Emotion Valence respectively. Each dataset is evenly split into two subtasks as tasks of DCSPs.

2) Models and Quality Rewarding: We implemented the GAN in `Python 3.8.10` with `Pytorch 1.4.0`. The generative model is a four-layer sequential model constructed in linear layers. The discriminative model is also a four-layer sequential model built from linear layers. The regression model the adversary exploits is a three-layer neural network, each of which is constructed in linear layers. The activate function applied for all models is the *rectifier* except for the output of the last layer, which is sigmoid for the generative model and the discriminative model to convert output to interval $[0, 1]$.

During experiments, μ_0 of the GTM is set to be the average of the submitted solutions and σ_0 is 1 both for the task and the workers. The distance function is chosen to be square distance and the initial weights of all solutions equal to $\log |\mathcal{D}|$ in the CRH algorithm, in which $|\mathcal{D}|$ denotes the size of the dataset and log is the natural logarithm. Unless explicitly specified, data are normalized to $[0, 1]$ using min-max normalization before training or during representation.

3) Threshold Reward Policy: In the threshold reward policy, the platform first calculates the standard deviation σ of the collected solutions and estimates the truth using three truth discovery algorithms: GTM, CRH and PACE. It then averages those estimated truth as the final truth τ. A worker is rewarded if and only if its solution d satisfies:

$$|d - \tau| < \sigma \tag{5}$$

4) Evaluation Metrics: To quantify effectiveness of the solution probing attack, we consider two metrics: the average rewards of the honest and generated solutions. Because those two together quantify the effectiveness of the solution probing attack.

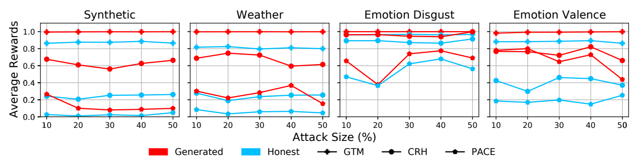

Fig. 3. Solution probing attack in plain DCSPs under quality related policy.

5) Parameter Configuration: During the experiments, we evaluate solution probing attack with two different variables: the number of solution-reward pairs the adversary probed and the number of generated solutions the adversary submitted, which are called probing size N and attack size $\mathcal{M}$ respectively. In plain DCSPs, N equals to $\mathcal{N}$ due to the public ledger feature of blockchain and $\mathcal{M}$ ranges from 10% to 50% with step 10% of $\mathcal{N}$. While in privacy preserving DCSPs, when evaluating N related to the attack effectiveness, it ranges from 10% to 100% with step 10% of $\mathcal{N}$ and $\mathcal{M}$ is 30% of $\mathcal{N}$; when evaluating $\mathcal{M}$ related to attack effectiveness, it ranges from 10% to 50% with step 10% of $\mathcal{N}$ and N is 50% of $\mathcal{N}$. For the GAN, the dropout rate between each layer is 30%. Probing and attack sizes are the ratio of the datasets to offset the impact of different data sizes. The base Bootstrapping parameter B is 100 with each training item augmented qB times where q is the normalized solution quality and epochs for training is 600. The loss function of the GAN is chosen to be *BCELoss*. The adversary generates $M = \mathcal{M}|\mathcal{D}|$ solution-reward pairs where $|\mathcal{D}|$ denotes the size of the dataset, then it submits $\mathcal{M}$ solutions with the highest indicated rewards to the platform.

B. Plain DCSPs

In plain DCSPs that under threshold reward policy, due to the simplicity of the threshold reward policy, all generated solutions are able to reap rewards from DCSPs in all attack sizes in all datasets, while in the honest workers, the average rewards range from 0.67 to 0.90, lower than the generated solutions. The result demonstrates the effectiveness and the stability of the solution probing attack in threshold reward policy, which means that the adversary learned the pattern of rewardable solutions.

Fig. 3 shows the result in plain DCSPs under quality related reward policy, which presents that the average rewards of honest solutions are higher than their honest counterparts in all quality evaluation algorithms and datasets, proving the effectiveness of the solution probing attack. The figure also shows that the average rewards of the generated solutions is on overall not very high. This is caused by the property of quality evaluation algorithms. Notice that with the increasing number of the generated solutions, the average rewards have reduced in some datasets, which is caused by regression toward the mean.

C. Privacy Preserving DCSPs

TABLE I presents the experiment results of different probing and attack sizes in each dataset in privacy preserving DCSPs under threshold reward policy. From the table we can learn that the average rewards of the adversary is higher than the honest workers when it comes to the maximum obtainable rewards except in Emotion Valence in attack sizes, which is due to the fact that adversary can only derive solutions from 10 solution-reward pairs. However, we can also notice that the average rewards of the adversary are not stable as the minimum average rewards are lower than the honest workers except in Weather, which induces the overall average rewards of the adversary lower than the honest workers. The results demonstrate that solutions encryption protects DCSPs to some extent, but the adversary is able to carefully choose its strategies in order to reap rewards from DCSPs.

TABLE I
REWARDS OF THE GENERATED AND HONEST SOLUTIONS IN PRIVACY PRESERVING DCSPS UNDER THRESHOLD REWARD POLICY

Dataset	(Generated, Honest) Solutions		
	Max	Min	Avg
Synthetic	**0.95**, 0.72	**0.02**, 0.62	**0.53**, 0.67
	0.78, 0.67	**0.65**, 0.63	**0.72**, 0.66
Weather	**1**, 0.73	**0.67**, 0.65	**0.94**, 0.69
	1, 0.75	**0.83**, 0.59	**0.96**, 0.67
Emotion Disgust	**1**, 0.95	**0**, 0.89	**0.63**, 0.92
	1, 0.92	**0.67**, 0.89	**0.80**, 0.91
Emotion Valence	**1**, 0.93	**0**, 0.77	**0.33**, 0.86
	0.57, 0.90	**0**, 0.70	**0.21**, 0.80

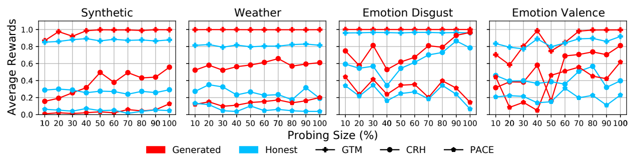

Fig. 4. Solution probing attack in privacy preserving DCSPs with different probing sizes under quality related policy.

Fig. 4 shows that under quality related reward policy, the average rewards of the adversary are higher than the honest worker in most cases in all datasets and quality evaluation algorithms, except when the amount of solution-reward pairs probed are small, rendering the regression model unable to derive the desired solutions from the rewards. Fig. 5 demonstrates that probes half of the solution-reward pairs is able to effectively to generate high quality solutions, as more solutions generated, the average rewards of the adversary are higher than the honest workers, which also demonstrates the effectiveness of the optimizations.

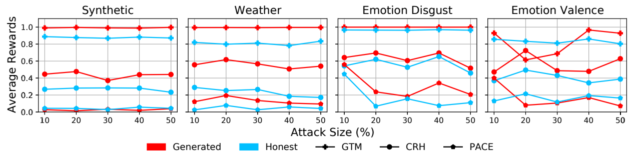

Fig. 5. Solution probing attack in privacy preserving DCSPs with different attack sizes under quality related policy.

The results of TABLE I, Fig. 4 and Fig. 5 demonstrate that even though the solutions are encrypted, the attacker is still able to launch attacks to generate high quality solutions especially in quality related reward policy, which is widely adopted in practice.

VI. Defenses

In this section, we discuss a defense to protect DCSPs from the solution probing attack. The intuition of the defense is to protect the solutions with encryption and the rewards with masks.

A. Mix-and-Match Defenses

To prevent DCSPs from the solution probing attack, we propose a defense based on a *mix-and-match* strategy. The *mix* requires the platform to protect rewards information related to solutions of tasks by adding masks, the *match* refers that each worker is required to participate in at least two crowdsourcing tasks, therefore their added mask can be offset to obtain the aggregated rewards. The constraint that workers are required to complete at least two tasks is practical since in reality most workers concurrently participates in multiple tasks [20], [25].

Generally, we assume that the task set assigned to worker W_k using well-developed schemes [26] is $\boldsymbol{t}_{W_k} = \{T_1, T_2, \ldots, T_m\}$, $k, m \in \mathbb{N}^+$. For security concern, we match each task with V companion tasks in $\boldsymbol{t}_{W_k}$, $V < m$. Since $\boldsymbol{t}_{W_k}$ may be updated as more tasks are published, V companion tasks can always be available for any task in $\boldsymbol{t}_{W_k}$. Assuming that we have companion task set $\boldsymbol{t}_{W_k}^{T_i}$ for task T_i by randomly selection in $\boldsymbol{t}_{W_k}$, W_k is required to accomplish all tasks in $\boldsymbol{t}_{W_k}^{T_i}$ before he can collect $cash_{k,T_i}$.

Specifically, we will mix tasks available to W_k and match another V tasks for T_i when W_k chooses to participate in T_i, $1 \leq V < m$. On W_k's registration, requesters owning these $V+1$ tasks should generate random numbers r_{R_j,W_k}, $j \in [1, V+1]$ using a verifiable random source, which is available on many blockchain implementations [27]. By exchanging random numbers with each other secretly, each requester of a companion task can agree on a mask $s_{T_i,W_k}^{R_l}$ for R_j, satisfying $\sum_{l\in[1,V+1], l\neq j} s_{T_i,W_k}^{R_l} = -r_{R_j,W_k}$. When solutions submitted by W_k are evaluated, R_j sends a masked reward $cash_{k,T_i} + r_{R_j,W_k}$ to DCSPs, which is usually implemented using a transaction to a smart contract. In order to be verified by miners in the blockchain, a proof $\Pi(T_i, R_j, W_k, d_k)$ should also be sent to the platform. If the proof is correct, then W_k will accept R_j's commitment of the reward and continue to finish the other V tasks. If W_k's solution is valid, then the requester R_l of a companion task will send a masked reward $cash_{k,T_l} + s_{T_i,W_k}^{R_l}$ along with the proof $\Pi(T_l, R_l, W_k, d_k)$ to the decentralized platform. Once the platform receives all masked rewards via transactions, a total reward $cash_k = cash_{k,T_i} + r_{R_j,W_k} + \sum_{T_l \in \boldsymbol{t}_{W_k}^{T_i}} cash_{k,T_l} + s_{T_i,W_k}^{R_l}$ will be calculated. Since $r_{R_j,W_k} + \sum_{l\in[1,V+1], l\neq j} s_{T_i,W_k}^{R_l} = 0$, the total reward is a sum of rewards of T_i and other V companion tasks.

TABLE II
Rewards of the generated and honest solutions in mix-and-match DCSPs under threshold reward policy

Dataset	(Generated, Honest) Solutions		
	Max	Min	Avg
Synthetic	**0.94**, 0.72	**0.32**, 0.60	**0.65**, 0.67
	0.92, 0.66	**0.69**, 0.61	**0.83**, 0.63
Weather	**1**, 0.75	**0.48**, 0.65	**0.79**, 0.69
	0.98, 0.76	**0.33**, 0.63	**0.71**, 0.72
Emotion Disgust	**0.90**, 0.95	**0**, 0.90	**0.16**, 0.93
	0.13, 0.96	**0**, 0.92	**0.03**, 0.94
Emotion Valence	**0.33**, 0.95	**0**, 0.77	**0.10**, 0.84
	0.60, 0.95	**0**, 0.53	**0.12**, 0.81

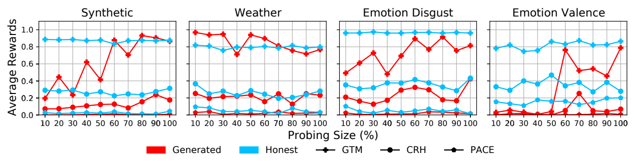

Fig. 6. Solution probing attack in mix-and-match DCSPs with different probing sizes under quality related reward policy.

B. Defense Experiments

The mix-and-match defense experiments are conducted both for the threshold reward policy and the quality related reward policy, in which the masks of companion tasks are offset under modulo $N = 2^{63} - 1$.

TABLE II shows the experiment results of different probing and attack sizes in each dataset of mix-and-match DCSPs under threshold reward policy. From the table we observe that the adversary obtains lower average rewards compared with their privacy preserving counterparts when it comes to the maximum except in Synthetic. The reason is that the Synthetic dataset follows Gaussian distribution, which means if the generated solutions are random, they will coverage to the interval of rewardable threshold, as demonstrated in the average of the generated solutions. However, we can observe that in different attack sizes in Synthetic, the adversary obtains rewards higher than those in privacy preserving DCSPs, which proves the effectiveness of the optimization of the solution probing attacks. The mix-and-match defense destabilize and lower the rewards of the adversary as shown in the Min and Avg columns of the table.

Fig. 6 shows the average rewards of the adversary and the honest workers in mix-and-match DCSPs, in which the adversary has varied probing sizes. It shows that except in

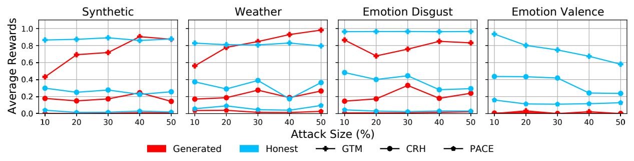

Fig. 7. Solution probing attack in mix-and-match DCSPs with different attack sizes under quality related reward policy.

partial Weather and Synthetic datasets with GTM algorithms, the average rewards of the adversary are lower than honest workers, demonstrating the effectiveness of the mix-and-match defense. However, the average rewards of the adversary decreased compared with ciphertext counterparts in Weather and Synthetic datasets. Fig. 7 shows that when the adversary generates more solutions, it cannot obtain higher average rewards than honest workers except in Weather dataset with GTM algorithms, which is caused by the fact that when more solutions are generated, it will affect the final result of the estimated quality in GTM algorithms. However, when compared with the ciphertext counterparts, the mix-and-match defense effectively decreases the rewards of the adversary.

Based on the results of TABLE II, Fig. 6 and Fig. 7, we conclude that the mix-and-match defense can effectively decrease the rewards of the adversary especially in the quality related reward policy, demotivating it launch the solution probing attack.

VII. Conclusion

This paper reports a new vulnerability of decentralized crowdsourcing platforms built atop blockchain, which leads to the solution probing attack, ruining data privacy and fair trade in crowdsourcing. We identify that the vulnerability lies in unprotected reward information on a public ledger in quality-aware crowdsourcing platforms. We conduct experiments on both synthetic and real-world datasets for the proposed attack in different capacity of the adversary, and demonstrate that the latter is able to obtain higher average rewards than honest workers even when a small ratio of data are probed. Noticing that the key to solve this problem is cutting off the connection between solutions and their corresponding rewards, we propose a defense strategy called mix-and-match. Our evaluation results show that the defense can effectively reduce the rewards the adversary obtained from the attack.

Acknowledgements

The authors would like to thank the anonymous reviewers for the time and efforts they have kindly made. This work was supported in part by the National Key R&D Program of China under Grants 2020YFB1005900, NSFC-61902176, BK20190294, NSFC-61872179, NSFC-61872176, and in part by the Leading-edge Technology Program of Jiangsu-NSF under Grant BK20202001.

References

[1] M. Miceli, M. Schuessler, and T. Yang, "Between subjectivity and imposition: Power dynamics in data annotation for computer vision," *Proceedings of the ACM on Human-Computer Interaction*, vol. 4, pp. 1–25, 2020.

[2] R. Mouawi, I. H. Elhajj, A. Chehab, and A. Kayssi, "Crowdsourcing for click fraud detection," *EURASIP Journal on Information Security*, vol. 2019, no. 1, pp. 1–18, 2019.

[3] W. Shen, X. He, C. Zhang, Q. Ni, W. Dou, and Y. Wang, "Auxiliary-task based deep reinforcement learning for participant selection problem in mobile crowdsourcing," in *ACM International Conference on Information & Knowledge Management*, 2020, pp. 1355–1364.

[4] B. Zhao, S. Tang, X. Liu, and X. Zhang, "Pace: Privacy-preserving and quality-aware incentive mechanism for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1924–1939, 2020.

[5] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han, "Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 1986–1999, 2016.

[6] B. Zhao and J. Han, "A probabilistic model for estimating real-valued truth from conflicting sources," *Proc. of QDB*, vol. 1817, 2012.

[7] X. Gao, H. Huang, C. Liu, F. Wu, and G. Chen, "Quality inference based task assignment in mobile crowdsensing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 10, pp. 3410–3423, 2020.

[8] D. E. Difallah, M. Catasta, G. Demartini, P. G. Ipeirotis, and P. Cudré-Mauroux, "The dynamics of micro-task crowdsourcing: The case of amazon mturk," in *International conference on world wide web*, 2015, pp. 238–247.

[9] J. Zhang, F. Yang, Z. Ma, Z. Wang, X. Liu, and J. Ma, "A decentralized location privacy-preserving spatial crowdsourcing for internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2299–2313, 2020.

[10] C. Li, B. Palanisamy, R. Xu, J. Wang, and J. Liu, "Nf-crowd: Nearly-free blockchain-based crowdsourcing," in *International Symposium on Reliable Distributed Systems*, 2020, pp. 41–50.

[11] Y. Lu, Q. Tang, and G. Wang, "Zebralancer: Private and anonymous crowdsourcing system atop open blockchain," in *IEEE International Conference on Distributed Computing Systems*, 2018, pp. 853–865.

[12] W. Feng and Z. Yan, "Mcs-chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain," *Future Generation Computer Systems*, vol. 95, pp. 649–666, 2019.

[13] H. Ma, E. X. Huang, and K.-Y. Lam, "Blockchain-based mechanism for fine-grained authorization in data crowdsourcing," *Future Generation Computer Systems*, vol. 106, pp. 121–134, 2020.

[14] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. H. Deng, "Crowdbc: A blockchain-based decentralized framework for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 6, pp. 1251–1266, 2018.

[15] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "P2P mixing and unlinkable bitcoin transactions," in *Annual Network and Distributed System Security Symposium*, 2017.

[16] Z. Zhang, J. Yin, Y. Liu, and J. Liu, "Deanonymization of litecoin through transaction-linkage attacks," in *International Conference on Information and Communication Systems*, 2020, pp. 059–065.

[17] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, "Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles," *IEEE communications magazine*, vol. 56, no. 10, pp. 50–57, 2018.

[18] Y. Zhang, M. Simsek, and B. Kantarci, "Self organizing feature map for fake task attack modelling in mobile crowdsensing," in *IEEE Global Communications Conference*, 2019, pp. 1–6.

[19] Y. Zhang and M. Van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *Proceedings IEEE INFOCOM*, 2012, pp. 2140–2148.

[20] M. Fang, M. Sun, Q. Li, N. Z. Gong, J. Tian, and J. Liu, "Data poisoning attacks and defenses to crowdsourcing systems," in *Proceedings of the Web Conference*, 2021, pp. 969–980.

[21] Y. Zhao, X. Gong, F. Lin, and X. Chen, "Data poisoning attacks and defenses in dynamic crowdsourcing with online data quality learning," *IEEE Transactions on Mobile Computing*, 2021.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[23] B. Efron, "Bootstrap methods: another look at the jackknife," in *Breakthroughs in statistics*. Springer, 1992, pp. 569–593.

[24] L. D. Laure Berti-Equille. Data sets for data fusion experiments. [Online]. Available: https://lunadong.com/fusionDataSets.htm

[25] R. Snow, B. O'connor, D. Jurafsky, and A. Y. Ng, "Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks," in *Proceedings of the conference on empirical methods in natural language processing*, 2008, pp. 254–263.

[26] L. Wang, D. Yang, X. Han, D. Zhang, and X. Ma, "Mobile crowdsourcing task allocation with differential-and-distortion geo-obfuscation," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 2, pp. 967–981, 2019.

[27] B. David, P. Gaži, A. Kiayias, and A. Russell, "Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2018, pp. 66–98.