Sybil Defense in Crowdsourcing Platforms

Dong Yuan[1,2], Guoliang Li[2], Qi Li[1], Yudian Zheng[3]

[1]Graduate School of Shenzhen, Tsinghua University
[2]Department of Computer Science, Tsinghua University
[3]Department of Computer Science, University of Hong Kong
yuand15@mails.tsinghua.edu.cn,liguoliang@tsinghua.edu.cn,qi.li@sz.tsinghua.edu.cn,ydzheng2@cs.hku.hk

ABSTRACT

Crowdsourcing platforms have been widely deployed to solve many computer-hard problems, e.g., image recognition and entity resolution. Quality control is an important issue in crowdsourcing, which has been extensively addressed by existing quality-control algorithms, e.g., voting-based algorithms and probabilistic graphical models. However, these algorithms cannot ensure quality under sybil attacks, which leverages a large number of sybil accounts to generate results for dominating answers of normal workers. To address this problem, we propose a sybil defense framework for crowdsourcing, which can help crowdsourcing platforms to identify sybil workers and defense the sybil attack. We develop a similarity function to quantify worker similarity. Based on worker similarity, we cluster workers into different groups such that we can utilize a small number of golden questions to accurately identify the sybil groups. We also devise online algorithms to instantly detect sybil workers to throttle the attacks. Our method also has ability to detect multi-attackers in one task. To the best of our knowledge, this is the first framework for sybil defense in crowdsourcing. Experimental results on real-world datasets demonstrate that our method can effectively identify and throttle sybil workers.

1 INTRODUCTION

Crowdsourcing platforms have been extensively deployed and provide a new machnism to address the labeling problem in machine learning. or other compute-hard problems, e.g., image recognition, entity resolution and speech recognition. A key issue in crowdsourcing is quality control that aims to eliminate noisy answers. A number of quality control algorithms have been proposed to eliminate bad workers by quality evaluation, e.g., qualification tests and golden questions [24], majority voting [14], probabilistic graphical models [12, 16, 28, 33, 35], and spammer detection [13, 14, 28].

However all these existing techniques cannot defend against sybil attacks in crowdsourcing. Sybil attacks allow attackers to register many sybil worker accounts so that they can submit similar answers for each question [11] that dominate those of normal workers. The quality control algorithms cannot ensure quality under the attacks. More importantly, worker behavior analysis in crowdsourcing is difficult without any prior knowledge on tasks under attacks and the number of the attackers. Traditional sybil detection in online social networks [1, 2, 23, 34] cannot be applied because of the difficulty in extracting the required features in crowdsourcing.

To tackle these challenges, we propose a cluster-based sybil defense framework (SADU) for crowdsourcing platforms, which detects sybil workers by analyzing similar behaviors of workers without any prior knowledge on the attack. In order to achieve this, we propose a worker similarity function to quantify the similarity of workers' behaviors by evaluating the answers returned by two worker pairs. In particular, we consider reliability of similarity so as to avoid inaccurate clustering incurred by unreliable similarity, e.g., the similarity is not reliable when two workers only have few common questions. Based on the computed similarity, we apply hierarchical clustering to cluster workers into different groups such that workers in the same group have similar behaviors, which enables SADU to accurately identify workers without requiring any prior knowledge on attacked tasks and the number of attackers. Moreover, SADU computes dynamic thresholds in hierarchical clustering according to the expectation of the similarity between two groups of workers, and identify the sybil groups by utilizing a small number of randomly generated golden questions. Furthermore, we devise online algorithms to instantly and efficiently detect sybil workers based on the previous detection results. SADU is robust even if attackers inject noise into the answers to hide their similar behaviors. In particular, in presence of multiple attackers, our method can still detect sybil workers with relatively high accuracy, and thus can ensure quality under sybil attacks. Our experiment results with real datasets demonstrate the high performance of SADU.

To summarize, we make the following contributions.
(1) We propose a sybil defense framework in crowdsourcing, called SADU, which can detect sybil workers and throttle sybil attack. To the best of our knowledge, this is the first sybil defense framework for crowdsourcing.
(2) We devise a worker similarity function to evaluate worker's similarity for clustering workers such that we can utilize a small number of golden questions to easily detect the sybil group.
(3) We develop online detection algorithms to detect the sybil workers, which significantly reduces the cost of sybil detection.

The paper is organized as follows. We first formulate the problem in Section 2. Then we present our sybil attack defense framework in Section 3. The online defense algorithm is introduced in Section 4. We conduct experiments in Section 5. We review related works in Section 6 and conclude the paper in Section 7.

2 PROBLEM DEFINITION

In this paper, we consider single-choice questions that ask workers to select one label from multiple given labels as the answer. Each question is assigned to multiple workers (e.g., three workers) and

CIKM'17 , November 6–10, 2017, Singapore, Singapore

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133039

Table 1: An example of questions, workers and answers. The sybil workers are $\{w_3, w_4, w_5\}$. Each question has three labels ($\mathcal{L} = \{0, 1, 2\}$) and asks workers to select one as the answer (– denotes the worker does not answer the question).

Questions	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
True Answer	1	1	2	0	2	0	0	2
w_1	1	1	2	0	-	-	0	2
w_2	1	1	-	-	2	-	-	-
w_3	2	-	0	2	-	0	-	-
w_4	-	-	0	2	2	0	1	1
w_5	-	2	-	-	2	1	1	1

each worker returns an answer for each assigned question. Then, result inference infers the correct result for each question based on the corresponding answers. Table 1 shows an example of questions, workers and answers. Note that, this paper focuses on addressing the attacks to the objective tasks where each question has only one correct answer. Our method can be extended to address attacks on other types of tasks by analyzing the correlation between labels. Moreover, strategic sybil attacks are not the focus of the paper since such attacks incur a large amount of manual labors to interact with detection and cannot be captured by analyzing answers. The attacks can be captured by analyzing more features (e.g., user profiles) in crowdsourcing platforms.

2.1 Attack Model

In this paper, we consider the sybil attack where an attacker can coordinate multiple robot workers (called sybil workers) to answer questions of the same task. For each same question, all the sybil workers select the same result (we will discuss that even if the sybil workers add some noises, e.g., giving the same answer with 90% probability and returning other answers with 10% probability). If the crowdsourcing platforms utilize the majority voting to infer the correct answer, the answer returned by sybil workers will be inferred as the "correct" answer, since majority workers are sybil workers.

For example, in Table 1, sybil workers, $\{w_3, w_4, w_5\}$, produce the same answer for questions q_3, q_4 and q_5. With majority voting answer inference, the system may infer the answers of q_3, q_4 and q_5 as 0, 2 and 2, respectively. However, the real correct answers are 2, 0 and 2. Thus, only q_5 is correctly inferred. Here the answers by sybil workers are taken as "correct" and the correct answers given by normal workers are treated as wrong. With majority voting answer inference applied in most crowdsourcing platforms, w_1 will be treated as a low quality worker since it only produces two "correct" answers (actually six correct answers). However, sybil worker w_4 generates six "correct" answers (actually only one correct answer) and w_4 will be treated as a high quality worker.

A straightforward detection method is to group sybil workers if they always return the same answers. To make the attacks stealthy, attackers can add noise to the answers so as to evade this detection. Attackers may allow the majorities sybil workers instead of all sybil workers to produce the same answers. For example, sybil worker w_5 generates a different answer for q_6 with other sybil workers. Note the quantity of the noise added by sybil workers is limited. Otherwise, the sybil workers will be treated as low quality workers by the system. Furthermore, there may exist multiple attacks in one task. Different attacks may generate various noises in the answers. So it is challenging to detect sybil attacks in crowdsourcing platforms.

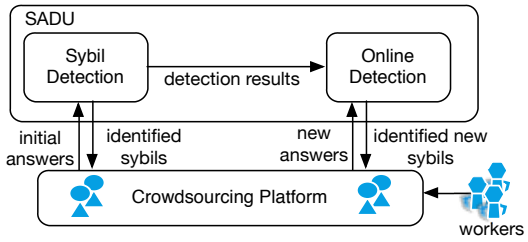

Figure 1: The framework of SADU.

2.2 Our Goal

Sybil defense in crowdsourcing platforms should achieve the following goals: (1) It should detect sybil workers and distinguish the answers produced by sybil workers and those by normal workers in realtime so as to filter out bad answers and improve the accuracy of inferring correct answers for each task. (2) It should monitor tasks and the workers' answers such that it can detect sybil attacks in realtime to filter out the sybil workers to reduce the useless money on sybil workers. (3) It should identify sybil workers even under attacks by multiple attackers. Since crowdsourcing platforms do not have any priori knowledge about the number of attacks in a task, detection of multiple attacks will be very challenging.

2.3 Overview

The framework of SADU is illustrated in Figure 1, which contains two main phases. (i) **Sybil Detection.** SADU collects worker answers (initial answers) from the crowdsourcing platforms, computes worker similarities, and clusters the workers. Then SADU leverages a small number of golden questions to identify groups of sybil workers. (ii) **Online Detection.** It instantly identifies sybil workers and eliminates such workers to save the cost, according to the answers submitted by workers and the results from the sybil detection.

3 SYBIL DETECTION

In this section, we first define the similarity of workers for worker clustering (Section 3.1). Then we discuss how to cluster workers (Section 3.2). Finally, we present sybil detection method based on the worker clustering results (Section 3.3).

3.1 Worker Similarity

Similarity Reliability. If two workers answer very few common questions, they will have a large probability to return the same answer for the same question even if they do not belong to the same type of workers (i.e., normal or sybil). If two workers answer many common questions, the normal worker has a small probability to return the same answer for the same questions to the sybil worker. For example, w_2 and w_4 share only one question and give the same answer. It is not reliable to treat them as the same type of workers since w_2 is a normal worker but w_4 is a sybil worker.

To summarize, the more common questions returned by two workers, the more reliable to evaluate their similarity. When the number of common questions is large, adding one more common

Table 2: Worker Similarity

	w_1	w_2	w_3	w_4	w_5
w_1	\	0.255	-0.373	-0.479	-0.373
w_2	0.255	\	-0.129	0.129	0
w_3	-0.373	-0.129	\	0.373	-0.129
w_4	-0.479	0.129	0.373	\	0.239
w_5	-0.373	0	-0.129	0.239	\

question should not cause much difference in computing the reliability. Hence, as the number of common questions increases, the reliability is expected to increase fast when the number of common questions shared is small, while keeping relatively stable when the number is large. Thereby, we formalize the reliability as below.

Definition 3.1 (Reliability of Similarity). Given two question sets, Q_i, Q_j, which are sets of questions answered by w_i and w_j, respectively, the reliability is defined as:

$$R(w_i, w_j) = \frac{2}{1 + \theta^{-|Q_i \cap Q_j|}} - 1, \tag{1}$$

where θ is a variable to control the change rate of reliability. Reliability ranges from 0 to 1. 0 indicates the reliability of similarity is not reliable at all. 1 indicates the relialibity of similarity is totally reliable. θ ensures that reliability can correctly reflect similarity with respect to different sizes of intersected question sets. For instance, if we set θ to 1.3, the reliability of sharing one question and three questions is 0.13 and 0.37, respectively. If we set the θ to 3, the reliability is 0.5 and 0.93. The reliability is close to the maximum value if there are three questions, and does not change significantly if there are over three questions.

By considering reliability, we can define the similarity of workers with their answers. If two workers answer more questions with same answers, they have larger similarity. Otherwise, more questions with different answers lead to smaller similarity. So, we define the worker similarity as follows.

Definition 3.2 (Worker Similarity). Given two workers w_i and w_j, let Q_i (Q_j) and $\mathcal{A}_{w_i}$ ($\mathcal{A}_{w_j}$) respectively denote the set of questions answered by and the set of answers provided by worker w_i (w_j), and $\mathcal{A}^q_{w_i}$ ($\mathcal{A}^q_{w_j}$) denote the answer given by worker w_i (w_j) for question $q \in Q_i$ ($q \in Q_j$). The similarity of w_i and w_j is:

$$S(w_i, w_j) = \frac{1}{|Q_i \cap Q_j|} \sum_{q \in Q_i \cap Q_j} R(w_i, w_j) \cdot I(\mathcal{A}^q_{w_i}, \mathcal{A}^q_{w_j}), \tag{2}$$

where

$$I(x, y) = \begin{cases} 1 & \text{if } x = y, \\ -1 & \text{if } x \neq y. \end{cases} \tag{3}$$

According to the definition, the range of worker similarity is $(-1, 1)$, where 1 indicates two workers always choose the same answer and -1 indicates two workers always choose different answers. As shown in Table 1, w_1 and w_2 generate the same answers for two questions and produce different answers for the other question and the reliability is 0.37. We can obtain that the similarity between w_1 and w_2 is 0.255 and the similarity between w_3 and w_4 is 0.373. Therefore, we can infer w_3 and w_4 are very likely to be the same type of worker.

3.2 Worker Clustering

Now we can cluster workers according to the computed similarity. Workers that are clustered into the same group should have similar behavior. It is difficult to define the number of groups and cluster workers by leveraging traditional cluster methods since we do not have any prior knowledge on the number of attackers. To address this issue, we extend the average linkage hierarchical clustering [8] to cluster workers. Initially, each worker is treated as a single group and iteratively merge groups. In each iteration, two groups with highest similarity will be merged into a group. The iteration will stop until all workers are clustered into one group or the similarity between all groups (i.e., group similarity) is below a threshold.

Group Similarity. Given two groups of workers G_p and G_q, the larger the similarity of each worker pair between the two groups, the larger the group similarity. So we use the average similarity of worker pairs between the two groups to represent the group similarity. Group similarity is defined as below.

Definition 3.3 (Worker Group Similarity). Given two groups of workers G_p and G_q and the worker pair set $P_w = \{\langle w_i, w_j \rangle | w_i \in G_p, w_j \in G_q, Q_i \cap Q_j \neq \emptyset\}$. The group similarity between G_p and G_q is:

$$S_G(G_p, G_q) = \frac{\sum_{w_i \in G_p, w_j \in G_q} S(w_i, w_j)}{|P_w|}. \tag{4}$$

Recall the example shown in Table 1, as the similarity of w_3 and w_5 is -0.129 and the similarity of w_4 and w_5 is 0.239, we can obtain that the group similarity of groups $\{w_3, w_4\}$ and $\{w_5\}$ is 0.055.

Dynamic Threshold. Now we compute thresholds for clustering, which are dynamically computed according to the clustering results. Different group pairs may have different thresholds with respect to the clustering results. Before computing a threshold, we need to compute the expectation of similarity between two workers, which can be obtained based on the reliability of similarity values. The reliability is computed according to the number of common questions two workers answered, which is independent of the value of answers. Thus, according to Equation (2), we can compute the expectation of similarity by using the following equation:

$$E(S(w_s, w_n)) = R(w_s, w_n) E\left(\sum_{q \in Q_s \cap Q_n} \frac{I(\mathcal{A}^q_{w_s}, \mathcal{A}^q_{w_n})}{|Q_s \cap Q_n|} \right), \tag{5}$$

where $R(w_s, w_n)$ is calculated according to the observed number of common questions shared by two workers. Then, we analyze the remaining expectation part, which can be treated as the expectation of similarity without the reliability. Considering a question with $|\mathcal{L}|$ labels, the probability that a sybil worker correctly returns an answer is $\frac{1}{|\mathcal{L}|}$, and the probability a sybil worker incorrectly returns an answer is $1 - \frac{1}{|\mathcal{L}|}$. Suppose the probability that a normal worker correctly answers a question is α, and then the probability that it incorrectly answers a questions is $1 - \alpha$. Hence, we obtain that the probability that both a normal worker and a sybil worker correctly answer a question is $\frac{\alpha}{|\mathcal{L}|}$ and the probability that both a normal worker and a sybil worker incorrectly answer a question on a specific label is $\frac{1}{|\mathcal{L}|} \frac{1-\alpha}{|\mathcal{L}|-1}$. As there are $|\mathcal{L}|$ labels, the probability that both a normal worker and a sybil worker incorrectly answer a question with all labels is $\frac{1-\alpha}{|\mathcal{L}|}$. Thus, we can obtain that the

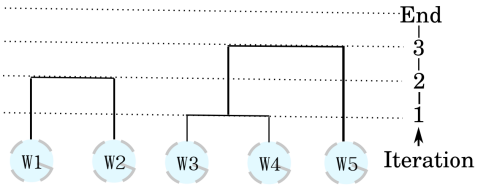

Figure 2: Group Structure of The Example

probability that both a normal worker and a sybil worker return the same answer is $\frac{1}{|\mathcal{L}|}$, and the probability that both a normal worker and a sybil worker return different answers is $1 - \frac{1}{|\mathcal{L}|}$. Based on all possible similarity values, the similarity between a sybil worker and a normal worker is $(\frac{1}{|\mathcal{L}|} - (1 - \frac{1}{|\mathcal{L}|}))$ without reliability. Therefore, we can obtain the expectation:

$$E(S(w_s, w_n)) = R(w_s, w_n) \cdot (\frac{2}{|\mathcal{L}|} - 1). \tag{6}$$

Note that, since the expectation of worker similarity is dynamic with respect to different workers, the computed expectation of group similarity is also calculated dynamically. Then, according to the expectation of worker similarity (see Equation (4)), we can easily get the expectation of group similarity:

$$E(S_G(G_s, G_n)) = \frac{\sum_{w_i \in G_s, w_j \in G_n} E(S(w_i, w_j))}{|P_w|}. \tag{7}$$

Moreover, there may exist noises in the computed expectation of group similarity, which is caused by misjudged workers. To address this issue, we add a positive parameter τ in the reliability and set the threshold to $E(S_G(G_s, G_n)) + \tau$. Intuitively, a large τ may cause some normal worker groups will not be merged, while a small value may cause the sybil group merge with normal worker group. We set τ to 0.10 in our example. The impact of τ is presented in Section 5.3.

Following the example in Table 1, we can cluster groups as follow. Given the similarity of w_3 and w_4 is 0.373, which is the largest similarity and larger than threshold -0.024, w_3 and w_4 are merged into one group. Next w_1 and w_2 are grouped together since their similarity is 0.255, the second largest similarity. Now w_3, w_4, and w_5 are grouped together, and its similarity is 0.055 over the threshold -0.001. Finally, the similarity of rest two group is -0.204, which is below the threshold, i.e., 0.003, and thereby the iteration terminates.

3.3 Sybil Detection

Sybil Group Determination. SADU leverages golden questions to identify sybil groups that mainly contain sybil workers. To achieve this, it randomly assigns k golden questions to workers and computes the quality of groups according to the ground truth. We compute the quality of each group based on the answers on the golden questions and treat a group with low quality as a sybil group. k golden questions are randomly distributed in Q_g questions. For a golden question q_g, if the number of answers given by workers in group G, which match ground truth answers, is larger than the number of the answers that do not match, G has a correct answer for q_g. Then the quality of groups is the ratio of correctly answered golden questions to the total number of answered golden questions. By leveraging a given quality threshold q_τ, we can identify normal groups and eliminate the groups with quality below q_τ. Note that, a small group with single or few workers may not be unidentified. We could identify the identities of such groups by assigning some golden questions to the workers in the groups in the next round of assignment. Since in most cases the number of this kind of group is small, we will label the groups as unidentified groups and detect the identities of workers in the group during online detection. For example, as the example shown in Table 1, if the ground truth of q_2 and q_8 are 1 and 2, respectively, we can easily determine the group $\{w_1, w_2\}$ is the normal group and $\{w_3, w_4, w_5\}$ is the sybil group.

Sybil Worker Detection. We identify sybil by evaluating the identities of workers in the groups. Given the results of clustering, workers in the groups whose qualities are over q_τ are labeled as normal l^n, while workers in groups whose qualities are under q_τ are labeled as sybil l^s. Workers in unidentified small groups will be labeled as uncertain l^u. Note that, in the sybil groups, workers who are assigned with only one or few questions might be normal workers. They are included in sybil groups by mistake because they do not share any answer with other normal workers in the normal worker groups. Therefore, we should not label these workers as sybil. We need to carefully evaluate workers who only answered a small number of questions, e.g., one or two questions. We introduce a threshold τ_{num} in sybil detection to address this issue. We can safely label workers if they answer over τ_{num} questions according to the following principles:

If a worker is in a unidentified group or has answered less than τ_{num} questions, the worker's identity is still undecided and will be labeled as l^u. If a worker who has answered no less than τ_{num} questions is in a group which quality is over q_τ, we can safely label the worker as l^n (normal worker). Otherwise, if a worker who has answered no less than τ_{num} questions is in a group which quality is under q_τ, we can safely label the worker as l^s (sybil worker).

Note that, the worker labeled with l^u will be re-evaluated when it uploads its answers during online detection.

4 ONLINE DETECTION

Now we can detect the sybil workers in runtime according to the results of sybil detection, and disable sybil workers. We propose a method to classify workers online to identify the new workers and workers with uncertain label. Note that, since both new workers and the workers labeled as uncertain only answer a small number of questions, we cannot use the similarity proposed above to detect sybil workers. Otherwise, the reliability of similarity between these workers and other workers will be very small and the value is not reliable to be used to classify the workers. To address this issue, we propose credit-based classification method. For each question answered by w, we calculate the credit for each existing group and classify the worker to the group with the maximum credit value. Here, the credit is calculated according to the answers given by workers from different groups. Intuitively, for a group, if a worker in the group shares the same answer with w, the credit of this group increases by one. If they do not share the same answer, the credit of the group reduces by one. However, there is a corner case. For example, for a question with ten same answers, if there are eight workers from the same group accidentally, the group gets eight credits, which is not acceptable. Thereby, we should limit the credit that a group can achieve for one question.

We calculate the credits of groups for one question as follows: first for each group, we collect the number of workers from this group who have the same answer to w, i.e., n_s, and the number of workers who do not generate the same answer to w, i.e., n_d. Then, we calculate the credit by $|n_s - n_d|$. If $n_s - n_d = 0$, we set the credits to 0. Therefore, we compute the credit for each group as below:

Definition 4.1 (Group Credits). Given a group G, a worker w, the set of questions answered by w, Q_w. For a question q, n_s^q (n_d^q) denotes the number of same (different) answers given by workers in G for q. Credits of a group computed as follows:

$$C_w(G) = \sum_{q \in Q_w} f(n_s^q - n_d^q), \tag{8}$$

where

$$f(x) = \begin{cases} 1 & \text{if } x > 0, \\ -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0. \end{cases} \tag{9}$$

Note that when a worker only answers few questions, we do not need to cluster this worker into any group. Only after the worker finishes τ_{num} questions, we will cluster the worker to the group with the maximum credit. If the maximum credits is below zero, which means the worker should be possibly from another group or a group of spammer workers. We should create a new group to include the worker. After assigning a worker to a group, we can label the workers based on the method in Section 3.

5 EXPERIMENTS

5.1 Experiment Setting

Datasets. We used three well-known real-world assignment datasets, i.e., Amt [26], Wsd [30], Dog [41]. Amt was a dataset where workers posted positive or negative feedbacks on movie reviews. It contained 500 questions, each of which was answered by 20 workers. Wsd asked the most appropriate sense of a word from tweets. Each question had three label options. It contained 177 questions, and each of them was answered by 10 workers. Dog was a dataset to classify the dogs, where each task had four options. It contained 807 questions, and each question was answered by 10 workers. Golden labels were provided by the original papers. These datasets were collected from different types of tasks and had different label set sizes, while the number of answers varied from 1770 to 10,000. We used ChinaCrowds[1] to collect the answers for online detection.

Evaluation Metrics. We used precision and recall to evaluate our sybil detection methods, and used the accuracy of answer to evaluate the final accuracy after removing the answers provided sybil workers. In all figures of accuracy, the line with "Original" denoted the accuracy before account filtering, and the other lines indicated the accuracy after account filtering.

Baseline Approach. UOSN [2] was the malicious account detection method for online social network. We treated answering a question as an action performed by a worker. If two workers gave the same answer to the same question, they performed the same action. If two workers gave the same answer to all the questions they answered, their similarity was one. We varied the similarity threshold from 0.4 to 0.9 to choose the best performance threshold. We found 0.75 is the best value. Note that, since community based method [33] incurred approximately 870 times slower than SADU and had a low quality if under attacks, we did not include the results in the paper.

Parameters Setting. We discussed how to set the parameters to make the parameters generic for different scenarios. We evaluated the impacts of θ and τ in real dataset experiments (Section 5.3), and used these settings in both simulation and real dataset experiments to measure the performance of SADU. According to our observation, we found that SADU could always achieve good detection accuracy with the following settings: θ in similarity computation was set to 1.3, τ was set to 0.10. More normal and golden questions used in computation led to higher precision, but meanwhile incurred more costs of money. We traded off between precision and cost by using the following settings. The minimum number of questions τ_{num} is set to 5, k is set to 10. Moreover, the quality thresholds in different datasets were set as follows: the threshold q_τ on Amt and Wsd datasets was set to 0.7, and that for Dog was set to 0.6, which was consistent with that used in qualification tests in crowdsourcing platforms. Note that, since the questions in Dog were not easy to answer and the average quality of workers was less than 0.7, we set q_τ as 0.6.

5.2 Simulated Experiment

In the simulations, we first considered all sybil workers harnessed by one attacker to construct different attacks, and the proportion of sybil workers was set to 0.6, i.e., 60 percent of workers were sybil workers, and the noise injected by sybil workers was set to 0.1, i.e., the sybil worker had the probability of 0.1 to choose an answer different from the other sybil workers. The total number of workers was set to 100, the number of tasks was set to be 200, the number workers assigned with one task was set to 5, the size of choice set $\mathcal{L}$ was set to 4, and the quality of normal worker was set to 0.85. We evaluated SADU under different scenarios with varying above settings. We run each our algorithm 50 times with randomly selected workers as sybil workers and computed the average results.

5.2.1 Evaluation with Varying Sybil Proportion. We evaluated SADU with tasks that had different proportions of sybil workers. As shown in Figure 3, SADU outperformed UOSN in precision, recall and accuracy. As the sybil proportion increased, the precision of SADU increased. The reason was that more sybil workers had more chances to answer the same question. However, the precision of UOSN increased slowly, because sybil workers may have chosen true answers accidentally and normal workers may also made mistakes. It was difficult to distinguish them using simple Jaccard similarity. Besides, the results of UOSN were not stable since UOSN might combine normal workers with sybil workers because of the defined similarity function. SADU achieved stably high recall, when the sybil proportion varied. The reason was that our similarity function captured the similarity of workers and the sybil workers were assigned to the same group in most of cases. Similarly, UOSN behaved unstable in recall, because sybil workers accidentally shared the same answer with the normal workers and might be clustered into the group of normal workers. When the sybil proportion was low, the precision was relatively low, as the number of sybil workers was significantly smaller than normal workers and many sybil workers did not even have chance to answer the

[1] http://www.chinacrowds.com

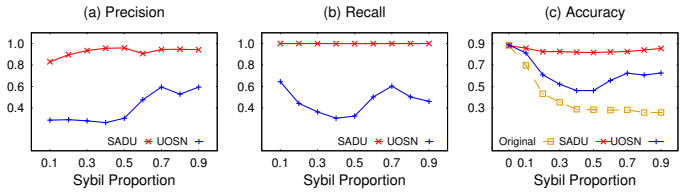

Figure 3: Simulation: Varying Sybil Proportion

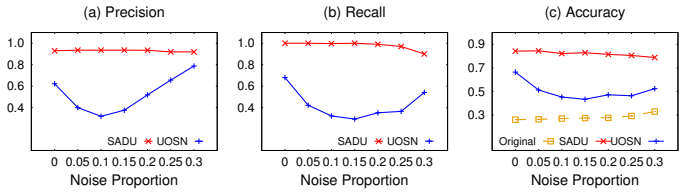

Figure 4: Simulation: Varying Noise Proportion

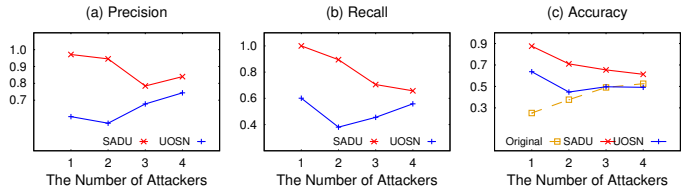

Figure 5: Simulation: Multiple Attackers

Figure 6: Simulation: Varying Normal Worker Quality

same question. In this case, normal workers who accidentally gave the same answer with sybil worker might be wrong clustered. As the sybil proportion grew, SADU kept a high accuracy due to the high recall of SADU. The increase in accuracy of SADU was higher than UOSN since SADU had high precision and recall. According to the above observations, although sybil attack would interfere with interfering correct answers, SADU reduced the effect and had better accuracy than baseline UOSN.

5.2.2 Evaluation with Varying Noise Proportion. We considered noise proportion as the probability that sybil workers chose different choices from other sybil workers for the same question. We evaluated SADU with different noise proportions injected by attackers. As illustrated in Figure 4, SADU maintained high precision and recall. We observed that even after the attacker injected 30% noise, SADU still achieved high precision and recall. The reason was that our similarity function can still capture similar behaviors between sybil workers who gave similar answers. With the increase in the noise proportion, the precision was relatively stable and the recall decreased. The reason was that, as the noise increased, the similarity between sybil workers decreased but the similarity between sybil workers and normal workers increased relatively, which might cause some normal workers be assigned to the sybil group by mistake. Interestingly, as the noise increased from 0.1 to 0.3, the performance of UOSN improved. The possible reason was that the UOSN was unstable and achieved high precision accidentally. Note that SADU still outperformed than UOSN. The accuracy results were shown in Figure 4. We observed that the accuracy of original answers increased when the noise proportion increased. The reason was that more answers were inferred as the one given by normal workers when the noise was injected to sybil workers' answers. As the noise proportion increased, the accuracy of SADU and UOSN decreased because more workers were misidentified due to the noise. The accuracy of UOSN decreased more significant than SADU since its clustering algorithm and similarity function could not cluster the sybil workers well.

To summarize, as the noise added by attacker increased, SADU still maintained high precision, recall and accuracy, which behaved much better the UOSN.

5.2.3 Evaluation with Multiple Attackers. We evaluated the performance of SADU with sybil workers harnessed by different number of attackers. We set the total proportion of sybil workers to 0.8, which could be controlled by several attackers, i.e., it was several groups of sybil workers. For simplicity, the number of sybil workers controlled by each attacker was the same. According to Figure 5, we can see that SADU still outperformed UOSN in precision, recall and accuracy. When the number of attacker was one and two, SADU achieved high precision and recall. SADU detected sybil groups precisely when the number of sybil workers was 80 percent of the total workers. As the number of attacker increased, the precision and recall of SADU decreased since more questions were answered by sybil workers and some sybil accounts might give the same answers to the normal workers. It was difficult to distinguish the sybil workers and normal workers. UOSN achieved low precision and recall with different numbers of attackers. As the number of attackers increased from two to four, the precision and recall of UOSN increased. The reason was that when the number of attackers was small, it was more likely to cluster normal and sybil group into one group. As shown in Figure 5, the accuracy inferred according to original answers increased as the number of attackers increased. Different attackers had different options to answer questions, which increased probability that answers given by normal workers are inferred as true answers. The accuracy of SADU and UOSN decreased as the numbers of attackers increases. Based on the observations, we observed that if the majority are sybil workers generated by multiple attackers, it was difficult to identify workers and infer true answers. However, SADU still achieved much better performance than UOSN.

5.2.4 Evaluation with Varying Worker Quality. We evaluated SADU with different normal worker quality. As shown in Figure 6, we observed that as the normal worker quality increased, the precision, recall and accuracy increased. The reason was that the higher quality of normal workers caused higher similarity between normal workers which indirectly improved the quality of clustering. SADU had a much higher precision and recall then UOSN. Even when normal worker quality was 0.7, SADU still achieved high precision and recall as 0.91 and 1.0, respectively. The accuracy of original answers was stable at around 0.25. Since the 60 percent of workers was sybil workers, the accuracy was almost the same as random guess. SADU improved the accuracy to 0.7-1.0 by accurate detection.

We can conclude the even with low quality normal worker, e.g. 0.7, SADU still provided stable and high precision and accuracy.

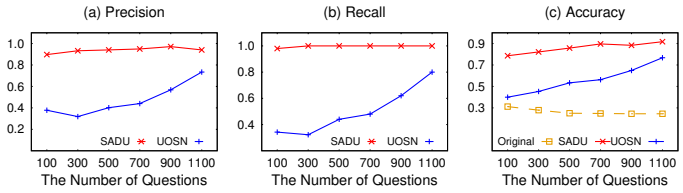

Figure 7: Simulation: Varying #Questions

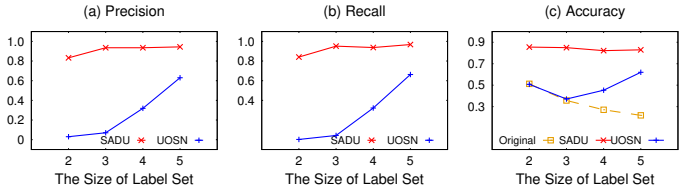

Figure 8: Simulation: Different Size of Choices

5.2.5 Evaluation with Varying #Questions. We evaluated our method by varying the number of questions. As illustrated in Figure 7, SADU maintained high precision, recall and accuracy with different number of questions. Even with only 100 tasks, SADU had a high precision and recall as 0.90 and 0.9, respectively. But the UOSN had low precision and recall. We observed that as the number of questions increased, the precision, recall and accuracy increased. The main reason was that more tasks made the similarity of workers more stable which led to less affected by accidents. Then the workers could be clearly clustered.

To summarize, SADU had high precision and recall even with few tasks and outperformed UOSN with different number of questions.

5.2.6 Evaluation with Varying The Size of $\mathcal{L}$. We evaluated SADU with different size of $\mathcal{L}$. As shown by Figure 8, SADU achieved high precision and recall in different size of choice sets. In the task with only two choices, it was hard to distinguish the normal worker and sybil workers. The main reason was with only two choices, the sybil workers had a probability of 0.5 to choose the right answer. Even the sybil workers chose the wrong answer, the normal workers who accidentally chose the wrong answers shared the same answer with sybil workers. So, it was much harder to identify the sybil workers with small choice options. But SADU still had high precision and recall with small choice set. When the choice set size was 2, the precision and recall of SADU were 0.84 and 0.83, respectively. The accuracy of SADU was 0.85 which was much higher than the accuracy of original answers and UOSN, 0.51. As the size of choice set increased, the precision and recall of SADU and UOSN increased, because normal workers and sybil workers were more likely to choose different answer with larger choice set.The accuracy of SADU decreased slightly as the size of choice set increased. But actually, the improvement of accuracy compare to original answers increased.

With above observations, we concluded that even with small choice set size, SADU still achieved high accuracy improvement and had much higher precision and recall than UOSN. Moreover, SADU performed even better with large choice set.

5.2.7 Evaluating Online Detection. Here we evaluate SADU with different number of questions used to identify one worker in online detection. We considered scenarios with different sybil proportion,

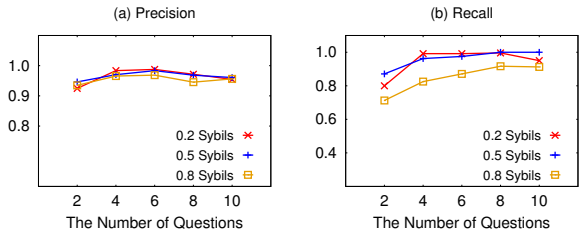

Figure 9: Simulation: Online Detection with Varying #Questions

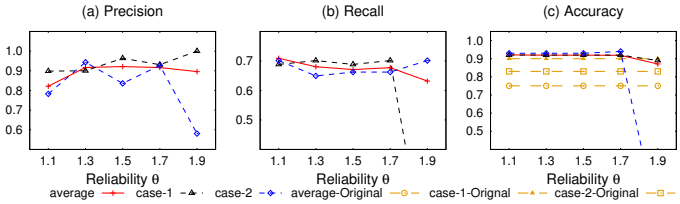

Figure 10: Real Dataset: Varying θ in Amt

e.g., 0.2, 0.5 and 0.8. We measured the precision and recall of detecting workers with uncertain labels from detection and 80 new coming sybil accounts. From Figure 9, even with 2 questions we identified the workers with high precision. When there were 20% sybil workers in the task, the precision was 0.92. But the recall was relatively low, as 0.8. Since only with two questions, it was still hard to identify every coming workers correctly. But as the number of questions increased, the recall increased. We also observed that, comparing to sybil proportion as 0.2 and 0.5, the precision and recall were relatively low when the sybil proportion was 0.8. The reason was since the majority of workers were sybil, it was more likely to misjudge the identifies of workers.

5.3 Real Dataset Experiment

Real dataset experiments with different sybil proportions, numbers of attackers and noise proportions were performed to evaluate SADU. We set the default sybil proportion to 0.6, the default number of attacker to 1, and the default noise proportion to 0.1. We run the algorithm for 50 times and present the average results.

5.3.1 Evaluation with Varying θ. We evaluated SADU with different θ which provided an insight of the impact of θ. We used the 100×20 answers from Amt dataset. The average results of Amt with two representative cases in the experiments were presented in Figure 10. In these two cases, some sybil workers who answered few questions accidentally shared the same answers with normal workers in these questions. In case-1 all workers were clustered into normal group, while in case-2 all workers were clustered into sybil group. As shown in Figure 10, sybil workers and normal workers were clustered into the same group with large θ. When θ was 1.9, the recall of case-1 was 0, and the precision of case-2 was 0.58. With smaller θ, the sybil workers and normal workers were clustered into different groups. On average, we observed that with θ as 1.1, the precision of SADU was relatively low. The main reason was that very small θ value led to the small reliability for the similarity of workers who shared few questions. Due to the similar behavior, most of the sybil workers still were clustered together. In this setting, some normal workers with relatively low quality were clustered into the sybil group. When θ was large than 1.3,

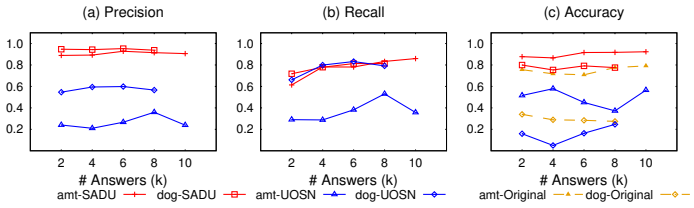

Figure 11: Real Dataset: Varying #Answers

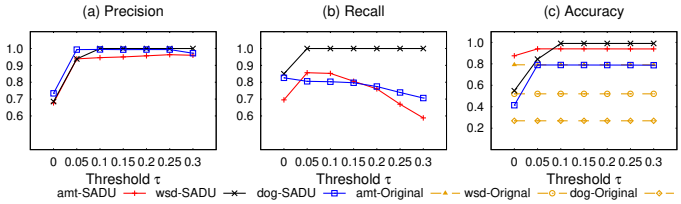

Figure 12: Real Dataset: Varying τ

as the θ increased, the recall decreased. It was mainly because the differences of reliability decreased with the increase in θ, i.e., the similarity of two workers shared few questions can also have relatively high reliability. Thereby, the normal worker group and sybil worker group were more likely to be clustered into one single group. Therefore, according to the above results, we can safely set θ to 1.3 in following experiments so that SADU can achieve good detection results in different experiments and other datasets.

5.3.2 Evaluation with Varying #Answers. We truncated the datasets to evaluate SADU with different number of answers. As shown in Figure 11, SADU had high precision even with 2k answers. The recall was relatively small, because some workers were labeled as uncertain when the number of answers was small. UOSN had low precision and recall. The reason was UOSN did not cluster the groups correctly with single linkage clustering algorithm without considering reliability. Sybil workers and normal workers who answered few questions were linked into the same group. As the number of answers increased, the precision of SADU kept stable and the recall increased. It was mainly because, with more answers, high reliability were achieved. More workers were clustered correctly. SADU improved the accuracy by removing the sybil workers as shown Figure 11(c).

5.3.3 Evaluation with Varying τ. We evaluated SADU with different τ. As illustrated in Figure 12, the precision and recall of three datasets increased when the τ increased from 0 to 0.05. The main reason was with τ as 0, the threshold of clustering was the same as expectation of the similarity. It caused some cases that normal worker group and sybil worker group were clustered into the same group. For Dog and Amt, as τ increased 0.10 to 0.3, the recall decreased. The main reason was that with high threshold, workers were more likely to be clustered into separate groups. But Wsd kept stable even τ was over 0.10. The reason was questions in Wsd were easy and most of workers answered over 20 questions. So even with relatively high threshold, the workers still were clustered accurately. According to the results, 0.10 was recommended as the default value for detection, larger value led to higher precision but lower recall.

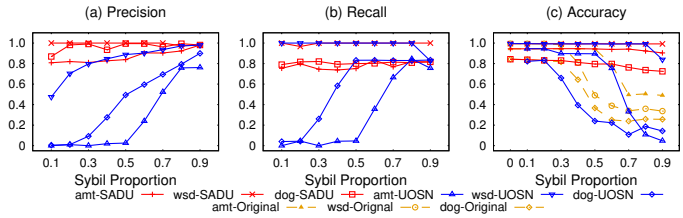

Figure 13: Real Dataset: Varying Sybil Proportion

5.3.4 Evaluation with Varying Sybil Proportion. We evaluated SADU with varying sybil proportion of attackers. As illustrated in Figure 13, SADU maintained high precision, recall and accuracy with different sybil proportions. The precision and recall was relatively small when the sybil proportion was small, e.g. 0.1, because it was hard to cluster the small number of sybil workers into groups. According to Figure 13 (c), we observe that the accuracy did not change significantly. The reason was the accounts we filtered had low quality compared with normal workers. As the sybil proportion increased, the precision and recall of SADU increased. When the sybil proportion was over 0.6, the accuracy of original data decreased severely. But SADU improved the accuracy to almost the same accuracy with no sybil workers. Note that UOSN had a very low precision and recall for Amt and Dog dataset. The reason was UOSN cannot separate the groups with similarity without reliability with single linkage clustering algorithm. We tested UOSN with different threshold, UOSN still assigned all workers to the same group. For Wsd, UOSN maintained high recall which made its accuracy behaved almost the same as SADU. The reason was that the quality of workers in dataset Wsd was very high and each task had over 10 answers, thus it was easy to distinguish the sybil and normal workers. The accuracy of SADU decreased slightly as the sybil proportion increased. The reason was the number of normal workers was too small when the sybil proportion was high, then workers could be easily misjudged. Therefore, we can conclude that even with different numbers of sybil workers in various datasets, SADU precisely detected the sybil accounts and improved accuracy significantly.

5.3.5 Evaluation with Multiple Attackers. We evaluated the performance of SADU with sybil workers harnessed by different number of attackers. We considered the situation that 80 percent of workers are sybil workers, i.e., including sybil accounts from several attackers. For simplicity, every attacker controlled the same number of sybil accounts. For example, if there were four attackers, then the number of sybil workers controlled by each attacker was 20% of the total number of workers. SADU still outperformed UOSN. As illustrated in Figure 14, SADU achieved high precision and recall and improved the accuracy significantly in Amt experiments. For Dog dataset, SADU still achieved high precision and recall. But the improvement of accuracy was not significant when the number of attacker was 4. The main reason was the questions in Dog were hard, it was hard to infer the answers correctly with 4 attackers. UOSN kept invariant in precision and recall, but performed poor in accuracy. The reason was that UOSN clustered most of workers into the same group, including both sybil and normal worker. While in Wsd experiments, SADU and UOSN both significantly improved the accuracy. The main reason was the same as explained in sybil proportion. As the number of attacker increased, SADU kept stable with precision and recall. Because SADU can cluster the workers

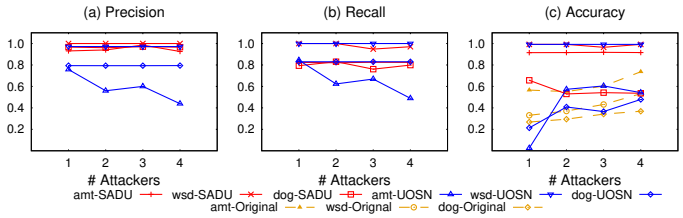

Figure 14: Real Dataset: Multiple Attackers

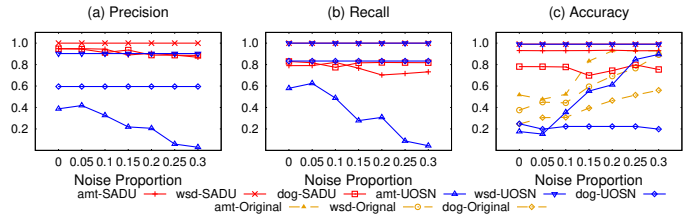

Figure 15: Real Dataset: Varying Noise Proportion

correctly. According the above results, we concluded that if the majority workers are sybil workers, existing methods cannot detect the sybil accounts. However, SADU significantly improved the accuracy of inferred answers.

5.3.6 Evaluation with Varying Noise Proportion. We evaluated SADU with different noise proportions added to sybil workers' answers. As illustrated in Figure 15, SADU outperformed UOSN in three datasets and improved the accuracy significantly with different noise proportion. The precision and recall of UOSN were still relatively low, since normal workers who answered few questions linked the sybil and normal worker group with single linkage clustering. As the noise proportion increased, the precision and of recall of SADU decreased slightly. The main reason was the noise made some sybil workers clustered to normal worker groups. As illustrated in Figure 15 (c), as the noise proportion increased, the accuracy of original answers increased since the answers of normal workers had more influence in the true answer inference. When the noise proportion reached 0.3, the accuracy of original answers reached 0.89 in the Wsd experiments, which means the attack failed since the answers chosen by sybil workers had not been treated as correct answers. UOSN reached almost the same accuracy with SADU in Wsd. The reason was similar to the experiments with varied sybil proportions. The recall of SADU decreased when the noise increased in Dog. However, accuracy of SADU was stable. Even if attackers added different ratio of noise to the sybil workers' answers, SADU still notably improved the accuracy of answers.

5.3.7 Online Detection. We conducted real experiments on ChinaCrowds, and evaluated the performance of our online worker filtering. We successfully identified one task under attack. The size of label set of this task was 2. We grouped five questions as a batch and the price of each was set 0.05$. The task contained 300×5 answers answered by 12 workers. In this dataset, seven workers were sybil workers controlled by one attacker and five workers were normal workers. The clustering algorithm run as 300 answers were collected. 10 workers were involved in these answers. We detected the sybil workers with 100% recall and 85.7% precision. In the sybil group, 6 workers were sybil workers whose bank accounts were the same. For two new coming workers, with 6 questions, we classified them correctly. This showed that our method works in real world with small cost.

6 RELATED WORK

Quality Control in Crowdsourcing. There has been extensive studies in quality control for Crowdsourcing [7, 18–20, 22, 37–40]. Existing crowdsourcing systems, e.g., Amazon Mechanical Turk[2], leveraged the qualification test, which is vulnerable to the proposed attack once the adversary answers the qualification test and copy the answers to robots. Recent works focus on estimating the worker quality with the answer given by workers during the assignment [12–14, 16, 24, 28, 28, 33, 35]. Ipeirotis et al. [12, 16, 27, 33, 35] leveraged probabilistic graphical models to infer the true answers. Whitehill et al. [35] leveraged the EM (Expectation Maximization) to estimate the worker quality, and took the task difficulty into consideration, while the expertise, biase and the community property of workers have been discussed in [16, 27, 33]. Raykar et al. [13, 14, 28] focused on the spammer detection to eliminate the workers that give the random answers. However, all these works are based on the assumption that most of workers are normal workers, which are vulnerable under our proposed attack. Note that, [13, 15, 24] discussed the collusion attack of workers, which are similar to our proposed attack. However, [24] studied the collusion problem in counting question and require a large mount of golden items to be injected into each task. [15] focused on rating task and explores the collusion relationship between worker pairs. [13] proposed hard penalty solution may assign more penalty to the normal worker when there exists a high percentage of malicious workers. Unfortunately, they cannot effectively detect and defend against sybil attacks. Sybil workers can pretend to be benign workers to pass qualification tests. In particular, a large amount of golden tests, e.g., 30% of total questions, are required to test each worker's quality [24]. Sybil workers can generate results dominating that produced by benign workers to evade detection by the inference model and other algorithms that assume majority workers are benign. [31] analyze the problem where the majority could be malicious theoretically. But it focus on extracting the top rating items from answers. If the number of reliable workers is small, e.g. only 10% workers, it required each worker to rate over 1000 items to curate the task, which is not practical in real setting.

Copy Detection in Truth Discovery. In the study of Truth Discovery, the topic of copy detection has been studied for years [4–6, 21]. Dong et al.[6] tried to find the copy relationship between source pairs and later uncovered more complicated copy relations [4], such as copy direction, co-copying and transitive copying. Recently, they proposed a scaled copy detection method [21]. All of these works assumed that most of the sources give right answers, thus their methods cannot be applied to solve our problem.

Sybil Attack in Other Platforms. Sybil attack has been studied in social network [1, 2, 10, 23, 32, 34]. Gang et al. [34] used the clickstreams to analyze the behavior of sybil accounts. Cao et al. [23] analyzed the dynamic connections (friend relationships) between malicious accounts and normal accounts to find the sybil accounts in social network. Boshmaf et al. [1] leveraged the the friend relationship between victim accounts and the sybil accounts to detect sybil accounts. However, such complex clickstreams or friend relationships are hard to obtain in crowdsourcing system. [2] leveraged

[2]https://www.mturk.com/

the similarity of malicious accounts' social network actions (e.g. all workers post pictures in a sustained period of time), to find the malicious accounts. Normal users in this scenario won't behave similarly, which is different from a typical crowdsourcing system. Although the similarity conception in [2] cannot check the task state and monitor online, we apply its similarity method and compare with ours in experiments. Stringhini et al. [32] analyzed the mapping between IP and accounts. It is orthogonal with our work, but adversary can still adjust their IP mapping strategy to bypass the detection.

Similar attack, shilling attack [17], has been studied in recommender system [3, 9, 29, 36]. Shilling attack aims to make a single target maintain high probability to be recommended in the system. This target would receive thousands or more responses for recommendation analysis which is different from the crowdsourcing scenario. What's more, most of these works [3, 29] have an underlying assumption that, the majority of users are honest. Even without this assumption, their methods are application specific, which try to recommend with attack resistant recommendation algorithm, e.g., collaborative filtering [9, 25] or trust-based algorithm [36].

7 CONCLUSION

In this paper we investigate the sybil attack problem in crowdsourcing. We propose a sybil defense framework to throttle the attack. We define a similarity function to evaluate worker behaviors based on their answers, and devise a grouping algorithm to cluster workers. Furthermore, we identify sybil groups using a small number of golden questions, and accurately detect sybil workers online. Experimental results show that our method effectively identified the sybil workers and significantly outperform the baseline approach.

8 ACKNOWLEDGEMENT

This work was supported by 973 Program of China (2015CB358700), NSF of China (61572278, 61632016, 61373024, 61602488, 61422205, 61472198).

REFERENCES

[1] Yazan Boshmaf, Dionysios Logothetis, Georgos Siganos, Jorge Lería, Jose Lorenzo, Matei Ripeanu, and Konstantin Beznosov. 2015. Integro: Leveraging Victim Prediction for Robust Fake Account Detection in OSNs.. In *NDSS*, Vol. 15. 8–11.

[2] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering large groups of active malicious accounts in online social networks. In *CCS*. 477–488.

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

[4] Xin Luna Dong, Laure Berti-Equille, Yifan Hu, and Divesh Srivastava. 2010. Global detection of complex copying relationships between sources. *VLDB* 3, 1-2 (2010), 1358–1369.

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

[6] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. 2009. Truth discovery and copying detection in a dynamic world. *VLDB* 2, 1 (2009), 562–573.

[7] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-Lee Tan, and Jianhua Feng. 2015. iCrowd: An Adaptive Crowdsourcing Framework. In *SIGMOD*. 1015–1030.

[8] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3 (2010), 75–174.

[9] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. 2014. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review* (2014), 1–33.

[10] Rupesh Gunturu. 2015. Survey of Sybil attacks in social networks. *arXiv preprint arXiv:1504.05522* (2015).

[11] Panos Ipeirotis. 2012. http://www.behind-the-enemy-lines.com/2012/01/identify-verification-and-how-to-bypass.html. (2012).

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

[13] Srikanth Jagabathula, Lakshminarayanan Subramanian, and Ashwin Venkataraman. 2014. Reputation-based worker filtering in crowdsourcing. In *NIPS*. 2492–2500.

[14] David R Karger, Sewoong Oh, and Devavrat Shah. 2011. Iterative learning for reliable crowdsourcing systems. In *NIPS*. 1953–1961.

[15] Ashiqur R KhudaBukhsh, Jaime G Carbonell, and Peter J Jansen. 2014. Detecting Non-Adversarial Collusion in Crowdsourcing. In *AAAI Conference on Human Computation and Crowdsourcing*.

[16] Aditya Kurve, David J Miller, and George Kesidis. 2015. Multicategory crowdsourcing accounting for variable task difficulty, worker skill, and worker intention. *TKDE* 27, 3 (2015), 794–809.

[17] Shyong K Lam and John Riedl. 2004. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web*. ACM, 393–402.

[18] Guoliang Li. 2017. Human-in-the-loop Data Integration. *PVLDB* 10, 12 (2017), 2006–2017. http://www.vldb.org/pvldb/vol10/p2006-li.pdf

[19] Guoliang Li, Chengliang Chai, Ju Fan, Xueping Weng, Jian Li, Yudian Zheng, Yuanbing Li, Xiang Yu, Xiaohang Zhang, and Haitao Yuan. 2017. CDB: Optimizing Queries with Crowd-Based Selections and Joins. In *SIGMOD*. 1463–1478.

[20] Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael J. Franklin. 2016. Crowdsourced Data Management: A Survey. *IEEE Trans. Knowl. Data Eng.* 28, 9 (2016), 2296–2319.

[21] Xian Li, Xin Luna Dong, Kenneth B Lyons, Weiyi Meng, and Divesh Srivastava. 2015. Scaling up copy detection. In *ICDE*. 89–100.

[22] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2015. A Survey on Truth Discovery. *SIGKDD Explorations* 17, 2 (2015), 1–16. https://doi.org/10.1145/2897350.2897352

[23] Changchang Liu, Peng Gao, Matthew Wright, and Prateek Mittal. 2015. Exploiting temporal dynamics in Sybil defenses. In *CCS*. 805–816.

[24] Adam Marcus, David Karger, Samuel Madden, Robert Miller, and Sewoong Oh. 2012. Counting with the crowd. In *VLDB*, Vol. 6. 109–120.

[25] Bhaskar Mehta and Thomas Hofmann. 2008. A Survey of Attack-Resistant Collaborative Filtering Algorithms. *IEEE Data Eng. Bull.* 31, 2 (2008), 14–22.

[26] Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Association for Computational Linguistics*. 271.

[27] Guo-Jun Qi, Charu C Aggarwal, Jiawei Han, and Thomas Huang. 2013. Mining collective intelligence in diverse groups. In *WWW*. 1041–1052.

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

[29] Paul Resnick and Rahul Sami. 2007. The influence limiter: provably manipulation-resistant recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*. ACM, 25–32.

[30] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Conference on Empirical Methods in NLP*. 254–263.

[31] Jacob Steinhardt, Gregory Valiant, and Moses Charikar. 2016. Avoiding imposters and delinquents: Adversarial crowdsourcing and peer prediction. In *Advances in Neural Information Processing Systems*. 4439–4447.

[32] Gianluca Stringhini, Pierre Mourlanne, Gregoire Jacob, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. 2015. Evilcohort: detecting communities of malicious accounts on online services. In *USENIX Security*. 563–578.

[33] Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. 2014. Community-based bayesian aggregation models for crowdsourcing. In *WWW*. 155–164.

[34] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y Zhao. 2013. You are how you click: Clickstream analysis for sybil detection. In *USENIX Security*. 241–256.

[35] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*. 2035–2043.

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

[37] Yudian Zheng, Reynold Cheng, Silviu Maniu, and Luyi Mo. 2015. On Optimality of Jury Selection in Crowdsourcing. In *EDBT*. 193–204.

[38] Yudian Zheng, Guoliang Li, and Reynold Cheng. 2016. DOCS: Domain-Aware Crowdsourcing System. *PVLDB* 10, 4 (2016), 361–372.

[39] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth Inference in Crowdsourcing: Is the Problem Solved? *PVLDB* 10, 5 (2017), 541–552.

[40] Yudian Zheng, Jiannan Wang, Guoliang Li, Reynold Cheng, and Jianhua Feng. 2015. QASCA: A Quality-Aware Task Assignment System for Crowdsourcing Applications. In *SIGMOD*. 1031–1046.

[41] Denny Zhou, Sumit Basu, Yi Mao, and John C Platt. 2012. Learning from the wisdom of crowds by minimax entropy. In *NIPS*. 2195–2203.