Identifying Unreliable and Adversarial Workers in Crowdsourced Labeling Tasks

Srikanth Jagabathula

Department of Information, Operations, and Management Sciences
Leonard N. Stern School of Business
44 West Fourth Street
New York University, NY 10012, USA

Lakshminarayanan Subramanian

Department of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York University, NY 10012, USA

Ashwin Venkataraman

Department of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York University, NY 10012, USA

Editor:

Abstract

In this paper, we study the problem of aggregating noisy responses from crowd workers to infer the unknown true labels of binary tasks. Unlike most prior work which has examined this problem under the probabilistic worker paradigm, we consider a much broader class of adversarial workers with no specific assumptions on their labeling strategy. Our key contribution is the design of a computationally efficient reputation algorithm to identify and filter out such adversarial workers in crowdsourcing systems, given only the labels provided by the workers. Our algorithm uses the concept of optimal semi-matchings in conjunction with worker penalties based on label disagreements, to detect outlier worker labeling patterns. We prove that our algorithm can successfully identify low reliability workers, workers adopting deterministic strategies; and is robust to manipulation by worst-case sophisticated adversaries who can adopt arbitrary labeling strategies to degrade the accuracy of the inferred task labels. Finally, we show that our reputation algorithm can significantly improve the accuracy of existing label aggregation algorithms in real-world crowdsourcing datasets.

Keywords: Crowdsourcing, Reputation, Adversary, Outliers

1. Introduction

The growing popularity of online crowdsourcing services like Amazon Mechanical Turk, CrowdFlower etc. has made it easy to collect low-cost labels from the crowd to generate training datasets for machine learning applications. Unfortunately, the collected labels typically are of low quality because of unintentional or intentional inaccuracies introduced by unreliable and malicious workers (Kittur et al., 2008; Le et al., 2010). Determining the correct labels of the tasks from such noisy labels is challenging because the reliabilities or qualities of workers are often unknown. While one may use “gold standard” tasks –
tasks whose true label is already known – to identify the low reliability workers (Snow et al., 2008; Downs et al., 2010; Le et al., 2010), getting access to the true labels for a sufficient number of tasks can be hard and expensive. To address these challenges, the common solution is to use redundancy (Sheng et al., 2008): collect multiple labels for each task and assign multiple tasks to each worker. Given the redundant labels, most existing work makes specific probabilistic assumptions on how individual workers provide labels and proposes techniques to identify low reliability workers, and either filter them out or de-emphasize their contribution. The most common probabilistic model is the following “one-coin” model (Zhang et al., 2014): given a binary task $t$ with true label either $+1$ or $-1$, a worker $w$ provides the correct label with probability $p_w$ and the incorrect label with probability $1 - p_w$. Therefore, the parameter $p_w$ is a measure of the reliability of worker $w$.

While most existing work relies on explicit probabilistic models, recent work (Vuurens et al., 2011; Difallah et al., 2012) and anecdotal evidence show that worker labeling strategies may not be random. Examples include strategies that: (a) uniformly label all tasks $+1$ if it is known that $+1$ labels are more prevalent than $-1$ among the true labels in the corpus of tasks\(^1\); (b) provide accurate labels to the first few tasks but random labels to the remaining tasks; (c) systematically provide $+1$ labels to certain types of tasks and $-1$ to other types. See Vuurens et al. (2011) for real-world empirical evidence of these worker strategies. In addition, workers may be malicious and adopt sophisticated strategies with the explicit purpose of altering the inferred labels for the tasks. For instance, workers have been observed to explicitly alter the popularity of advertisements and phishing articles (Tran et al., 2009) and collaboratively target products on Amazon.com (Mukherjee et al., 2012).

Motivated by the presence of non-random worker strategies, we go beyond the standard random model and study the problem of inferring the task labels under a much broader class of adversarial worker strategies. We distinguish between two types of workers: honest and adversarial. The worker types are latent and our objective is to use only the labels provided by the workers as input to identify the adversaries. More concretely, we focus on the following setting. The tasks have binary true labels in $\{-1, +1\}$. The population of workers is mostly honest, with adversaries comprising a “small” fraction. The honest workers adopt a well-defined probabilistic labeling strategy (e.g. the one-coin model introduced above). However, we make no assumptions on the specifics of the probabilistic strategy. The adversaries adopt strategies different from that of the honest workers, whether probabilistic or not. Further, different adversaries may adopt distinct strategies.

For the above setting, we design a scoring algorithm that assigns a ‘reputation score’ to each worker, which is a measure of the degree to which the worker’s labeling strategy is adversarial. Because we make no specific assumptions on the honest or adversary labeling strategies, our approach is to analyze the labeling patterns of the workers and detect statistical ‘outliers’. These outliers tend to be adversaries because the population is mostly honest and the labeling patterns of the adversaries are different from that of the honest workers. Once the adversaries are identified, their labels can be filtered and the remaining worker labels can be used to infer the task true labels using existing label aggregation algorithms. The choice of the appropriate aggregation algorithm depends on the application at hand. We study the impact of our filtering algorithm in the context of four popular ag-

\(^1\) Note that the “one-coin” model cannot capture this strategy, but the more general “two-coin” model (Raykar and Yu, 2012) can
Identifying Unreliable and Adversarial Workers

ggregation algorithms, as described in Section 5 on numerical results. The generality of our algorithm means that it can be applied in conjunction with any label aggregation algorithm to (potentially) improve its accuracy. Furthermore, the outliers identified by the algorithm may either be discarded or processed separately, depending on the application.

Main results. Our main contribution is a reputation algorithm, designed to identify outlier labeling patterns. The algorithm takes as input the set of workers, the set of tasks, and the binary labels provided by each worker. Each worker may provide labels to only a subset of the tasks. Because the algorithm makes no specific assumptions on the worker labeling strategies, it identifies outliers by penalizing the workers for the number of ‘conflicts’ they are involved in. More precisely, suppose every task receives both +1 and −1 labels. For each task $t_j$, the algorithm maintains the number $d^+_j$ of +1 labels, the number $d^-_j$ of −1 labels, and a penalty budget of 2. Intuitively, if $d^+_j > d^-_j$, then the worker assigning −1 to $t_j$ is “more” of an outlier than a worker assigning label +1. Based on this intuition, the algorithm must make the following decisions: (a) how much of the penalty budget to allocate to each worker for each task and (b) how to aggregate the penalties allocated to each worker to arrive at the final reputation score. We propose two algorithms that differ on how they make these two decisions. The first algorithm (Algorithm 1) does a ‘soft’ assignment: allocates a penalty of $1/d^+_j$ (resp. $1/d^-_j$) to every worker who has provided the label +1 (resp. −1) to task $t_j$ and computes the net penalty as the average of the allocated penalties across all tasks assigned to a worker.$^2$ The second algorithm (Algorithm 2) relies on a ‘hard’ assignment: instead of spreading the penalty score of 1 over all workers who provide the label +1 (resp. −1), it identifies one ‘representative’ worker among the workers who provide the label +1 (resp. −1) to task $t_j$ and allocates the entire penalty of 1 to the representative worker; and the net penalty for the worker is computed by summing the allocated penalties. If the representative worker is chosen uniformly at random, then as intuitively desired, a worker who agrees with the majority is less likely to be chosen and thereby less likely to receive the penalty. However, we show that it is more appropriate to choose the representative worker in a “load-balanced” fashion by using the concept of an optimal semi-matching (Harvey et al., 2003), where the semi-matching is defined on the bipartite graph between the workers and tasks - where every worker is connected to the tasks that she has labeled. For both types of penalty assignments, it is intuitively clear that because there are more honest workers than adversaries, the honest workers are more likely to agree with the majority and thereby receive a lower penalty.

We demonstrate the effectiveness of our algorithm using a combination of strong theoretical guarantees and empirical results on synthetic and real-world datasets.$^3$ For the theoretical analysis, we consider three scenarios. First, we consider the standard setting where all the workers provide labels according to the standard one-coin model. In this setting, we show that when the bipartite graph between workers and tasks is $(l, r)$-regular (i.e., all the workers label exactly $l$ tasks and all the tasks receive exactly $r$ labels), the reputation scores are proportional to the reliabilities of the workers (see Theorem 1). Therefore, our algorithm identifies workers with low reliabilities. This result shows that our definition of

---

2. We do not explicitly define the reputation score, it can be interpreted as the inverse of the net penalty - higher the penalty, lower the reputation and vice-versa.

3. This work expands on the results described in a previous version (Jagabathula et al., 2014)
reputation score is consistent with the notion of worker reliability for the one-coin model, as desired.

Second, we consider the setting in which a fraction of the workers adopt common deterministic strategies such as, what we call, the “uniform strategy” (providing the same label to all assigned tasks) and the “smart strategy” (providing the same high prevalence label to all assigned tasks). For these strategies, we show (see Theorems 2 and 3) that the workers adopting the deterministic strategies receive lower reputation scores than their “honest” counterparts, provided the honest workers have “high enough” reliabilities and the bipartite graph between the workers and the tasks is $(l, r)$ regular.

Third, we consider the setting in which the adversaries adopt sophisticated strategies with the objective of maximizing the number of tasks they affect (i.e., cause to receive labels different from what they would have received in their absence). We allow the adversaries to be unrestricted in their strategies, have infinite computational capacity, and even possess knowledge of the labels provided by the honest workers. For this setting, we provide a lower bound on the minimum number of tasks that $k$ adversaries can affect (i.e. the true label cannot be inferred better than a random guess), irrespective of the label aggregation algorithm employed to aggregate the worker labels (as long as it is agnostic to worker/task identities). The bound depends on the graph structure between the honest workers and the tasks (see Theorem 4 for details). Our result is valid across different labeling patterns and a large class of label aggregation algorithms, and hence provides fundamental limits on the damage that $k$ adversaries can do. Further, we propose a label aggregation algorithm utilizing the worker reputations computed in Algorithm 2 and prove the existence of an upper bound on the worst-case number of affected tasks (see Theorem 6). This combined with the result of Theorem 4 shows that our proposed label aggregation algorithm is optimal (upto a constant factor) in recovering the true labels of the tasks.

Finally, using several publicly available crowdsourcing datasets (see Section 5), we show that our reputation algorithms: (a) can help in enhancing the accuracy of state-of-the-art label aggregation algorithms (b) are able to detect workers in these datasets who exhibit certain ‘non-random’ strategies.

1.1 Related Work

Our work is part of the literature in crowdsourcing that proposes statistical techniques to exploit the redundancy in labels to simultaneously infer the latent reliabilities of workers and the true labels of tasks. In particular, our work is related to three broad streams. The first stream proposes methods for label aggregation when workers adopt specific probabilistic labeling strategies. Our filtering algorithm can work in conjunction with any of these methods. The second stream of work proposes methods to explicitly filter out low-reliability workers and is similar in spirit to our approach. Finally, the third stream focuses on methods to address sophisticated attacks in online settings and is related to our treatment of sophisticated adversaries.

The literature on the methods proposed to infer true labels of tasks under probabilistic labeling strategies is vast. The following are but a few examples: Dawid and Skene (1979); Smyth et al. (1995); Whitehill et al. (2009); Raykar et al. (2010); Welinder et al. (2010); Ghosh et al. (2011); Liu et al. (2012); Zhou et al. (2012); Dalvi et al. (2013); Karger
et al. (2014). Most of these works are based on the worker model proposed by Dawid and Skene (1979) which is a generalization of the one-coin model to tasks with more than two categories. The standard solution is to use the Expectation-Maximization (EM) algorithm to estimate the worker reliability parameters and task true labels. The methods proposed in (Liu et al., 2012; Chen et al., 2013) take a Bayesian approach by assuming a prior over the worker reliability parameters, while Whitehill et al. (2009) include task difficulty as an additional parameter in the model. Welinder et al. (2010) studied a model with multi-dimensional latent variables for each worker such as competence, expertise and bias. All of these works offer no theoretical guarantees on the resulting estimates of the task true labels and the model parameters. The one exception is the recent work by Zhang et al. (2014) who propose a novel initialization scheme for the EM algorithm based on the spectral method and show that their algorithm achieves optimal convergence rate under the Dawid and Skene (1979) model. Ghosh et al. (2011) and Dalvi et al. (2013) propose methods based on Singular-Value Decomposition (SVD) and analyze the consistency of their estimators for the one-coin model. Karger et al. (2014) propose an iterative message-passing algorithm for estimating the true task labels as well as a task assignment scheme that minimizes the total price that must be paid to achieve an overall target accuracy. They show that their algorithm is optimal through comparison with an oracle estimator who knows the reliability of every worker. The common thread in all these works is the imposition of specific models that characterize the worker labels, and studies have shown that they are susceptible to high levels of noise in the worker labels (Hung et al., 2013; Sheshadri and Lease, 2013).

In addition, there are many works that aim to explicitly detect and/or remove unreliable workers based on the observed labels. One approach is to use “gold standard” tasks, i.e. tasks whose true label is already known, to identify low reliability workers (Snow et al., 2008; Downs et al., 2010; Le et al., 2010). However, getting access to task true labels can be hard and might involve additional payment. In this work, we do not assume access to any gold standard tasks and identify adversarial workers based only on the provided labels. Vuuren et al. (2011) define scores customized to specific adversary strategies to identify and remove them. Similarly, Hovy et al. (2013) model the worker population as consisting of two types – one who always provide the correct label and spammers who provide uniformly random labels – and estimate the trustworthiness of each worker using the observed labels. Unlike these works, we allow the adversaries to adopt arbitrary strategies. Ipeirotis et al. (2010) proposed a way of quantifying worker quality by transforming the observed labels into soft posterior labels based on the estimated confusion matrix (Dawid and Skene, 1979). Similar to our work, their approach computes an expected cost for each worker – higher the cost, lower the quality of the worker. Raykar and Yu (2012) propose an empirical Bayesian algorithm to eliminate workers whose labels are not correlated with the true label (called spammers), and estimate the consensus labels from the remaining workers. Both of these works rely on the Dawid and Skene (1979) model, whereas our algorithm does not rely on specific probabilistic assumptions for worker strategies.

Finally, our work is also broadly related to the rich literature of identifying Sybil identities in online social networks. Most of these schemes (Yu et al., 2006, 2008; Danezis and Mittal, 2009; Tran et al., 2011; Viswanath et al., 2012) make use of the graph (or trust) structure between users to limit the corruptive influences of Sybil attacks (see Viswanath et al. (2010) for a nice overview). In our context, there is no information about the network
structure or trust relationships between workers and since most crowdsourcing tasks involve some form of payment, it is harder to launch Sybil attacks by forging financial credentials like credit cards or bank accounts.

2. Setup

We consider the following broad setting. There is a set $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ of $m$ tasks, such that each task $t_j$ is associated with a latent ground-truth binary label $y_j \in \{-1, +1\}$. Our goal is to determine the labels for each of these tasks by eliciting binary responses from a set $W = \{w_1, w_2, \ldots, w_n\}$ of $n$ workers. Each worker is typically assigned only a subset of the tasks. We represent this assignment using a bipartite graph $B = (W \cup T, E)$ with the workers on one side, the tasks on the other side, and an edge $(w_i, t_j) \in E$ indicating that worker $w_i$ was assigned task $t_j$. The graph $B$ is termed the worker-task assignment graph.

We suppose that the assignment $B$ is pre-specified.

Each worker $w_i$ provides a binary response $w_i(t_j) \in \{-1, +1\}$ for each task $t_j$ assigned to her. We encode the responses as the response matrix $L \in \{-1, 0, +1\}^{W \times T}$ such that $L_{ij} = w_i(t_j)$, for all $1 \leq i \leq n$ and $1 \leq j \leq m$, where we set $w_i(t_j) = 0$ for any task $t_j$ not assigned to worker $w_i$. We let $W_j \subseteq W$ denote the set of workers who labeled task $t_j$ and $T_i \subseteq T$ the set of tasks assigned to worker $w_i$. Let $d_j^+$ (resp. $d_j^-$) denote the number of workers labeling task $t_j$ as $+1$ (resp. $-1$).

**Worker model.** As mentioned earlier, the label provided by a worker may not be the true label of the task because workers either make mistakes or deliberately provide incorrect labels. To recover the true labels, we assume the following worker model. The population of workers is comprised of two disjoint classes: honest and adversarial. That is, $W = H \cup A$ with $H \cap A = \emptyset$, where $H$ is the set of honest workers and $A$ is the set of adversarial workers. The class memberships of the workers are latent, so we do not know whether a worker is honest or not. Honest workers provide (noisy) labels according to some probabilistic model (such as the one-coin model introduced in Section 1). Adversarial workers are those whose labeling strategy does not conform to this probabilistic model and they can adopt arbitrary (deterministic or probabilistic) strategies. Examples include: (a) the uniform strategy, in which the worker arbitrarily provides the uniform response $+1$ or $-1$ to all the assigned tasks, irrespective of the true label; (b) the smart strategy, in which the adversary is smart and chooses the uniform label in accordance with the population prevalence of the true labels, so that, if more than 50% of the tasks are a priori known to have the label $-1$, then the worker chooses $-1$ as the uniform label and vice-versa; or (c) the sophisticated strategy, in which the worker adopts strategies specifically designed to cause the maximum “damage” (refer to section 4.2 for details). Note that all of these example strategies cannot be captured by the one-coin model.

We make the following remarks about our model. First, most existing work only focuses on the honest workers and our contribution is also considering adversarial workers. Second, our definition of the “adversary” is intentionally broader than common definitions to accommodate a wide range of labeling strategies. In fact, some of the example adversary strategies described above may be accommodated by extending the standard one-coin model. For instance, the uniform strategy can be accommodated by allowing the worker
parameter \( p_w \) to also depend on the true label of the task. The smart strategy can be accommodated by further allowing the parameters to depend on the population prevalence of the task labels. While such case-by-case extensions are feasible in theory, they result in custom label aggregation algorithms with limited scope of application. In addition, they do not extend to general adversary strategies, including sophisticated strategies specifically designed to inflict the maximum “damage”.

Given the broad definition of adversaries, our approach for label aggregation is to identify and filter out the adversarial workers, and then apply an existing state-of-the-art label aggregation algorithm designed for honest workers to determine the task true labels. More precisely, our objective is to solve the following problem:

**Problem 1** Given a set of workers \( W = H \cup A \), tasks \( T \) and the response matrix \( L \), identify the subset of adversarial workers \( A \).

We describe a reputation-based algorithm that only relies on the response matrix \( L \) to detect the adversaries. The algorithm relies on detecting workers whose labeling patterns are statistical outliers among the population of workers. Once the adversaries are eliminated, we leverage the rich body of work on aggregating labels from probabilistic workers.

For the rest of the paper, we make the following technical assumptions. These assumptions are required only for proving theoretical guarantees, and the algorithm itself can be applied even if the assumptions don’t hold. They also help in motivating the intuition behind our reputation algorithms. Suppose that the honest workers provide labels according to the one-coin model: for task \( t_j \), honest worker \( h_i \) provides the correct label \( y_j \) with probability \( p_i \) and the incorrect label \( -y_j \) with probability \( 1 - p_i \). Define the reliability for honest worker \( h_i \) as \( \mu_i := 2p_i - 1 \) so that \( \mu_i \in [-1, 1] \). We assume that the honest workers are in the majority, \( |H| \geq |A| \), and their reliabilities are sampled from an underlying population with average reliability \( \mu > 0 \). The assumption that \( \mu > 0 \) is required for unique recovery of the true labels of the tasks, even with only honest workers (Karger et al., 2014). Finally, the tasks are sampled from a population in which the prevalence of the positive label is \( \gamma \in [0, 1] \), i.e. there is a fraction \( \gamma \) of tasks with true label +1. Note that our algorithms do not require knowledge of \( \mu \) or \( \gamma \).

### 3. Reputation Algorithms

We now describe the algorithm we propose to identify the adversarial workers, given the response matrix \( L \). We suppose there is no side information on the identities of the workers (such as, say, a social network or worker-level demographic information), so the algorithm must solely rely on the response patterns given by the workers. Our approach is to compute a “reputation” or “trust” score for each worker as a measure of the degree to which their response pattern is a statistical outlier or anomaly. Workers with low reputation scores are significant outliers and are filtered out as adversaries.

To compute the reputation of a worker, the algorithm relies on the number of conflicts the worker is involved in. Broadly speaking, a worker is involved in a conflict if her response to an assigned task is in disagreement with those of other workers. Note that tasks with a consensus opinion, having all +1 labels or −1 labels, do not provide any discriminative information about the workers who labeled the task. In other words, we cannot distinguish
between honest and adversarial workers from just this specific task. Therefore, we focus on the tasks that lack consensus, having both +1 and −1 labels. We term this subset of tasks as the conflict set $T_{cs}$, and workers who respond to tasks in the conflict set are all involved in conflicts. A conflict typically signifies the presence of low reliability honest workers (who tend to make mistakes) or adversaries. In the ideal case when all honest workers are perfect, i.e., have reliabilities $\mu_i = 1$, a conflict necessarily means the presence of an adversarial worker. In this case, the number of conflicts a worker is involved in can serve as a rough indicator of the possibility of the worker being an adversary. However, when honest workers are not perfect and make mistakes, a conflict indicates only a chance of the presence of an adversary. A simple counting of the number of conflicts then, may over-penalize honest workers who label a large number of tasks.

To overcome the issue of over-penalizing honest workers, we propose two penalty aggregation techniques, resulting in two variants of our algorithm: (a) soft-penalty and (b) hard-penalty. We describe these variants in detail next.

### 3.1 Soft Penalty

In the soft-penalty algorithm (see Algorithm 1), for any task $t_j$ in the conflict set, we allocate a penalty of $1/d_j^+$ to all workers who provide the label +1 for $t_j$ and $1/d_j^-$ to all workers who provide the label −1. Then for each worker, we compute the net penalty by averaging the penalties across all assigned (conflict) tasks.

The above allocation of penalties implicitly rewards agreements among worker responses by making the penalty inversely proportional to the number of other workers that agree with a worker. In particular, if a worker agrees with the majority opinion on some task, then she is allocated a lower penalty than a worker who disagrees with the majority. Further, taking the average normalizes for the number of tasks labeled by any worker. The algorithm relies on the following intuition in allocating the penalties: assuming the average reliability of the honest workers $\mu > 0$, we expect that on an average, the honest workers provide the correct response to the assigned tasks. Further, because there are more honest workers than adversaries, we expect the majority response to be the same as the true label of the task, for most of the tasks. Therefore, we expect that the above allocation of penalties assigns lower penalties to high reliability honest workers, and higher penalties to low reliability honest and adversarial workers. We formalize this intuition in Section 4, where we prove theoretical guarantees for the soft-penalty algorithm. We show that the soft-penalty algorithm performs well in identifying low reliability honest workers as well as adversarial workers employing deterministic strategies (refer to Theorems 1, 2 and 3). Our results demonstrate asymptotic consistency of the soft-penalty algorithm in identifying the adversaries, under standard assumptions on the structure of the worker-task assignment graph.

Even though the soft penalty algorithm is successful in identifying adversarial workers adopting certain types of strategies, its performance depends on the complexity of the strategies employed by the adversaries. If the adversarial workers are non-colluding and adopt non-deliberate strategies, then the soft penalty algorithm can identify them from the observed responses. The algorithm however is subject to manipulation by more sophisticated adversaries who can collude together and adapt their labeling strategy to target certain tasks to lower their penalty scores (see Section 4.2.2 for an example strategy).
particular, the soft penalty algorithm treats each task in isolation when assigning penalties and therefore is susceptible to attack by determined adversaries who can cleverly decide their responses based on the honest worker labels and the structure of the worker-task assignment graph, to cause maximum “damage”. In fact, for such worst-case adversaries, we show that (see Theorem 4) for any collection of honest worker responses, there exists a lower bound on the fraction of tasks whose true labels cannot be inferred correctly (better than a random guess), by any label aggregation algorithm (as long as the aggregation algorithm satisfies a few natural properties). To account for such adversarial behavior, we introduce the hard-penalty algorithm next.

3.2 Hard Penalty

To address the case of these sophisticated adversaries, we propose a hard penalty allocation scheme (Algorithm 2) in which the penalty allocation for a particular task takes into account the structure of the worker-task assignment graph and the responses of the other workers on all the other tasks. In particular, instead of distributing the penalty evenly across all the workers that respond to a given task, the algorithm chooses two ‘representative’ workers to penalize per conflict task: one representative worker among those who provide the label +1 and another among those who provide the label −1. The choice of the representative workers is done in a load-balanced manner to “spread” the penalty across all the workers, so that we don’t over-penalize workers who provide labels for a large number of tasks. The net penalty of each worker is the sum of the accrued penalties across all the (conflict) tasks assigned to the worker. Intuitively, such a hard allocation of penalties will penalize workers with higher degrees (i.e. large number of assigned tasks) and many conflicts (who are potential worst-case adversaries), thereby leading to a low reputation.

To choose the representative workers in load-balanced fashion, we use the concept of optimal semi-matchings (Harvey et al., 2003) on bipartite graphs. For a bipartite graph \( G = (V_1, V_2, E) \), a semi-matching in \( G \) is a set of edges \( M \subseteq E \) such that each vertex in \( V_2 \) is incident to exactly one edge in \( M \) (note that vertices in \( V_1 \) could be incident to multiple edges in \( M \)). A semi-matching generalizes the notion of matchings on bipartite graphs. The optimal semi-matching is the semi-matching with the minimum cost - we use the common degree-based cost function, defined as follows: for each \( u \in V_1 \), let \( \text{deg}_M(u) \) denote the degree of \( u \), i.e. the number of edges in \( M \) that are incident to \( u \) and let \( \text{cost}_M(u) \) be defined as

\[
\text{cost}_M(u) = \sum_{i=1}^{\text{deg}_M(u)} i = \frac{\text{deg}_M(u)(\text{deg}_M(u) + 1)}{2}
\]

An optimal semi-matching then, is one which minimizes \( \sum_{u \in V_1} \text{cost}_M(u) \). Intuitively, an optimal semi-matching fairly matches the \( V_2 \)-vertices across the \( V_1 \)-vertices so that the “load” on any \( V_1 \)-vertex is minimized. The above notion of cost is motivated by the load balancing problem for scheduling tasks on machines. Specifically, consider a set of unit-time tasks \( T \) and a set of machines \( P \). Suppose that each task \( t \) can be processed on a subset of the machines, this can be specified as a bipartite graph between \( T \) and \( P \). On any given machine, the tasks are executed one after the other in serial order. An optimal semi-matching can be thought of as an assignment of the tasks to the machines such that
the flow-time, i.e. the average completion time of a task, is minimized. Refer to (Harvey et al., 2003) for more details.

To determine the representative workers for each task, we compute the optimal semi-matching in the following augmented worker-task assignment graph: we split each task \( t_j \) into two copies, \( t^+_j \) and \( t^-_j \), and connect worker \( w_i \) to \( t^+_j \) if the worker labeled the task +1 and to \( t^-_j \) if the worker labeled the task −1. The optimal semi-matching, by definition, yields two representative workers for each task \( t_j \) – one connected to \( t^+_j \) and the other connected to \( t^-_j \). As for the case of soft-penalty, we only consider conflict tasks when creating this augmented bipartite graph. The worker degrees in the optimal semi-matching then constitute their net penalties. The hard penalty algorithm is described in detail in Algorithm 2.

<table>
<thead>
<tr>
<th><strong>Algorithm 1</strong>: <strong>SOFT PENALTY</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1: Input:</strong> ( W, T ) and ( \mathcal{L} )</td>
</tr>
<tr>
<td>**2: For every task ( t_j \in \mathcal{T}<em>{cs} ), allocate penalty ( s</em>{ij} ) to each worker ( w_i \in W_j ) as follows:</td>
</tr>
</tbody>
</table>
| \[ s_{ij} = \begin{cases} 
\frac{1}{d^+_j}, & \text{if } \mathcal{L}_{ij} = +1 \\
\frac{1}{d^-_j}, & \text{if } \mathcal{L}_{ij} = -1 
\end{cases} \] |
| **3: Output:** Net penalty of worker \( w_i \), \( \text{pen}(w_i) = \frac{\sum_{t_j \in \mathcal{T}_i \cap \mathcal{T}_{cs}} s_{ij}}{|\mathcal{T}_i \cap \mathcal{T}_{cs}|} \) |
| **Algorithm 2**: **HARD PENALTY** |
| **1: Input:** \( W, T \) and \( \mathcal{L} \) |
| **2: Create a bipartite graph \( \mathcal{B}^{cs} \) as follows: |
| (i) Each worker \( w_i \in W \) is represented by a node on the left (ii) Each task \( t_j \in \mathcal{T}_{cs} \) is represented by two nodes on the right \( t^+_j \) and \( t^-_j \) (iii) Add the edge \((w_i, t^+_j)\) if \( \mathcal{L}_{ij} = +1 \) or edge \((w_i, t^-_j)\) if \( \mathcal{L}_{ij} = -1 \). |
| **3: Compute an optimal semi-matching \( M \) on \( \mathcal{B}^{cs} \) |
| **4: Output:** Net penalty of worker \( w_i \), \( \text{pen}(w_i) = \text{deg}_M(w_i) \) |

### 3.3 Connection between soft-penalty and hard-penalty algorithms

While the hard and soft-penalty algorithms appear different on surface, the soft-penalty algorithm can be interpreted as the random, normalized variant of the hard-penalty algorithm. Specifically, suppose we choose a random semi-matching \( M \) in the augmented worker-task assignment graph \( \mathcal{B}^{cs} \), defined in Algorithm 2, and assign the penalty \( \text{deg}_M(w_i)/\text{deg}_{\mathcal{B}^{cs}}(w_i) \) to worker \( w_i \), where \( \text{deg}_{\mathcal{B}^{cs}}(w_i) \) is the degree of worker \( w_i \) in \( \mathcal{B}^{cs} \). When the random semi-matching is constructed by mapping each copy \( t^+_j \) (or \( t^-_j \)) of task \( t_j \) uniformly at random to a worker connected to it, the probability that it will be mapped to worker \( w_i \in W_j \) is equal to \( 1/\text{deg}_{\mathcal{B}^{cs}}(t^+_j) \) (or \( 1/\text{deg}_{\mathcal{B}^{cs}}(t^-_j) \)), or equivalently, \( 1/d^+_j \) (or \( 1/d^-_j \)). Therefore, the expected degree \( \text{deg}_M(w_i) \) of worker \( w_i \) in semi-matching \( M \) is equal to \( \sum_{t_j \in \mathcal{T}_i \cap \mathcal{T}_{cs}} s_{ij} \), where \( s_{ij} = 1/d^+_j \) if \( \mathcal{L}_{ij} = +1 \) and \( 1/d^-_j \) if \( \mathcal{L}_{ij} = -1 \). Because the degree \( \text{deg}_{\mathcal{B}^{cs}}(w_i) \) of worker \( w_i \) is equal to \( |\mathcal{T}_i \cap \mathcal{T}_{cs}| \), it follows that the expected penalty of worker \( w_i \) is equal to \( \text{deg}_M(w_i)/\text{deg}_{\mathcal{B}^{cs}}(w_i) = \sum_{t_j \in \mathcal{T}_i \cap \mathcal{T}_{cs}} s_{ij}/|\mathcal{T}_i \cap \mathcal{T}_{cs}| \), which is exactly the penalty allocated by the soft-penalty algorithm. It thus follows that penalties under the above random, normalized variant of the hard-penalty algorithm concentrate around the penalties allocated by the soft-penalty algorithm. When all the workers are assigned the same number of tasks, the normalization does not matter, and hence, the random hard-penalty and the soft-penalty algorithms allocate the same penalties on average.
With the above interpretation of the soft-penalty algorithm, it follows that the hard-penalty algorithm differs from the soft-penalty algorithm in two key aspects: it (a) does not normalize the penalties by degrees and (b) uses optimal semi-matchings as opposed to random semi-matchings. The absence of degree-based normalization of the penalties results in significant penalization of high-degree workers. The use of the optimal semi-matching results in more aggressive load-balancing of penalties by optimizing a global objective function. Both of these effects make the hard-penalty algorithm conservative and robust to sophisticated adversary strategies, as established theoretically in Section 4. The above connection also suggests that the soft-penalty algorithm without degree normalization should have a performance similar to that of the hard-penalty algorithm. We explore this aspect numerically in Section 5.

4. Theoretical Results

Our reputation algorithms are analytically tractable, and we establish their theoretical properties below. The proof of all theorems are in the Appendix.

4.1 Soft-penalty algorithm: common adversary strategies

We analyze the performance of the soft-penalty algorithm under three settings: (i) the classical setting when there are no adversaries ($A = \emptyset$), (ii) when the adversaries adopt the uniform strategy, and (iii) when the adversaries adopt the smart strategy. Both strategies are described in detail below.

The performance of the soft-penalty algorithm depends on (a) the true labels of the tasks (b) the reliabilities of the honest workers and (c) the structure of the worker-task assignment graph $B$. For the purposes of our theoretical analysis, we focus on the following generative model (refer to the discussion towards the end of Section 2):

- Sample the true label $y_j \sim Ber(\gamma)$ for each task $t_j$, where we assume that the Bernoulli distribution is defined on $\{-1, +1\}$.
- Sample reliability $\mu_i \in [-1, 1]$ for each honest worker $h_i$ according to some distribution $f$ such that $E[f] = \mu > 0$.
- Assign tasks to workers using a random $(l, r)$-regular bipartite graph $B$, where each worker is assigned to exactly $l$ tasks and each task is labeled by exactly $r$ workers.
- For each assigned task $t_j$, honest worker $h_i$ provides response $y_j$ with probability $\frac{1 + \mu_i}{2}$ and $-y_j$ otherwise.

The above setting is standard (see, for example, Karger et al. (2014)).

We first focus on the classical setting in which there are no adversarial workers, so that $A = \emptyset$. As stated in Theorem 1, we show that our reputation scores are consistent with honest worker reliabilities, which capture the workers’ propensity to make mistakes.

**Theorem 1 (Reputations consistent with reliabilities)** Under the above generative model, with high probability as $r \to \infty$, for any pair of workers $h_i$ and $h_i'$ such that $\mu_i < \mu_i'$, it follows that $\text{pen}(h_i) > \text{pen}(h_i')$. In other words, higher reliability workers are allocated lower penalties by the soft-penalty algorithm.
Theorem 1 shows that even when there are no adversaries, filtering out low-reputation workers filters out only the workers with low reliabilities.

Next, we consider the case when $A \neq \emptyset$ so that certain workers are adversarial. In particular, suppose that there is a fraction $1 - q > 0$ of adversarial workers. We consider two common strategies: (a) Uniform and (b) Smart.

**Definition 1 Uniform Strategy.** Every adversarial worker provides response $+1$ on all tasks assigned to her.

We can show the following:

**Theorem 2 (Performance under Uniform Strategy)** Suppose there is a fraction $1 - q$ of adversarial workers, who adopt the Uniform Strategy and let $a_{\text{unif}}$ denote an arbitrary adversarial worker. Under the generative model in Theorem 1, if $q > \frac{1}{1 + \mu}$ then with high probability as $r \to \infty$, there exists threshold $\delta_U \in (-1, 1)$ depending on $\gamma, \mu, q$ such that $\mu_i > \delta_U \implies \text{pen}(h_i) < \text{pen}(a_{\text{unif}})$.

The above theorem establishes that when adversaries adopt the Uniform Strategy, the soft-penalty algorithm assigns lower penalties to honest workers whose reliabilities exceed some threshold, as long as the fraction of honest workers $q$ is “large enough”. Because of the symmetry, an analogous result can be proved for the case when adversaries always provide the response $-1$. Next, we consider the Smart Strategy:

**Definition 2 Smart Strategy.** If the prevalence of positive tasks $\gamma > \frac{1}{2}$ (resp. $\gamma < \frac{1}{2}$), then every adversarial worker provides response $+1$ (resp. $-1$) on all tasks assigned to her.

The smart strategy can be motivated by a concrete example: consider the task of differentiating between benign and malignant tumors in medical images. Then it is likely that most of the images contain benign tumors and only a few contain malignant ones. Using this information, a “smart” worker can just label all the assigned images as being benign without carefully looking through each image.

We can prove the following result for the Smart Strategy:

**Theorem 3 (Performance under Smart Strategy.)** Suppose there is a fraction $1 - q$ of adversarial workers, who adopt the Smart Strategy and let $a_{\text{smart}}$ denote an arbitrary adversarial worker. Under the generative model in Theorem 1, if $q > \frac{1}{1 + \mu}$ then with high probability as $r \to \infty$, there exists threshold $\delta_S \in (-1, 1)$ depending on $\gamma, \mu, q$ such that $\mu_i > \delta_S \implies \text{pen}(h_i) < \text{pen}(a_{\text{smart}})$. Further, we have that $\text{pen}(a_{\text{smart}}) \leq \text{pen}(a_{\text{unif}})$, where $\text{pen}(a_{\text{unif}})$ is the penalty allocated to an adversary adopting the Uniform Strategy.

The theorem says that the penalty assigned to a smart adversary is less than or equal to that of a uniform adversary, which is intuitive. In addition, honest workers with reliability $\mu_i < 0$ receive a larger penalty when adversaries adopt the smart strategy as compared to the uniform strategy, so identifying smart adversarial workers becomes harder. The above results establish the following: (i) the soft-penalty algorithm is successful in identifying low reliability honest workers in the absence of adversaries and (ii) adversarial workers adopting common strategies (such as the uniform and smart strategies) can be identified using our soft-penalty algorithm. Further, all of the theorems above extend to the random, normalized variant of the hard-penalty algorithm described in Section 3.3.
Identifying Unreliable and Adversarial Workers

4.2 Hard-penalty algorithm: sophisticated adversary strategies

In the preceding analysis, we focused on common adversary strategies in which the adversaries were not intentionally malicious. However, existing work provides ample evidence for the presence of workers with malicious intent in crowdsourcing systems. These workers are usually hired on the Web by an attacker (Wang et al., 2012) to create fake accounts and manipulate the ratings/reviews with the purpose of altering the aggregate ratings or rankings received by the tasks. Specific examples include: workers on Digg altering the “popularity” of advertisements and phishing articles (Tran et al., 2009), fake review groups collaboratively targeting products on Amazon (Mukherjee et al., 2012), workers providing fake ratings and reviews to alter the aggregate ratings of restaurants on Yelp (Molavi Kakhki et al., 2013), etc. See the recent work by Wang et al. (2014) for more examples. Motivated by these examples, we study settings with sophisticated adversaries, defined as follows:

**Definition 3** Sophisticated adversaries provide responses with the objective of maximizing the number of tasks whose inferred labels are different from the labels they would otherwise have received from any label aggregation algorithm. They are computationally unbounded, colluding, and possess knowledge of the labels provided by the honest workers; therefore, they can adopt arbitrary response strategies.

Our definition allows the sophisticated adversaries to not just be malicious but capable of executing the most complex strategies. In practice, the adversary may adopt feasible strategies with varying complexities, depending on the application context and their objectives. But focusing on the most sophisticated adversary makes our analysis broadly applicable, independent of the application context.

For our analysis, we measure the performance of the hard-penalty algorithm in terms of accuracy, the number of tasks that receive correct labels, as opposed to their ability to identify the adversaries. This is done to gain analytical tractability.

Before we analyze the performance of our algorithm, we prove a lower bound on the number of tasks that will receive incorrect labels, irrespective of the label aggregation algorithm employed to aggregate the worker responses. The lower bound provides a way to assess the optimality of the hard-penalty algorithm.

### 4.2.1 Lower bound on the number of tasks that receive incorrect labels

To state our result, we need the following additional notation. We represent any label aggregation algorithm as a decision rule \( R : \mathcal{L} \rightarrow \{-1,+1\}^m \), which maps the observed labeling matrix \( \mathcal{L} \) to a set of output labels for each task. Because of the absence of any auxiliary information about the workers or the tasks, the class of decision rules, say \( \mathcal{C} \), is invariant to permutations of the identities of workers and/or tasks. More precisely, \( \mathcal{C} \) denotes the class of decision rules that satisfy \( R(PLQ) = R(L)Q \), for any \( n \times n \) permutation matrix \( P \) and \( m \times m \) permutation matrix \( Q \). We say that a task is affected if a decision rule outputs the incorrect label for the task. We define the quality of a decision rule \( R(\cdot) \) as the worst-case number of affected tasks over all possible true labelings of the tasks and adversary strategies given a fixed set of honest worker responses. Fixing the responses provided by the honest workers allows isolation of the effect of the adversary strategy on the accuracy of
the decision rule. Considering the worst-case over all possible task true labelings makes the quality metric robust to ground-truth assignments, which are typically application specific.

To formally define the quality, let \( B_H \) denote the subgraph of the worker-task assignment graph restricted to honest workers \( H \) and \( y = (y_1, y_2, \ldots, y_m) \) denote the vector of true labels for the tasks. We assume that \( B_H \) also encodes the honest worker responses to keep the notation succinct. Finally, let \( S_k \) denote the strategy space of \( k \leq |H| \) adversaries, where each strategy \( \sigma \in S_k \) specifies the \( k \times m \) response matrix given by the adversaries. Since we do not restrict the adversary strategy in any way, it follows that \( S_k = \{-1, 0, +1\}^{k \times m} \).

The quality of a decision rule \( R \in C \) is then defined as

\[
\text{Aff}(R, B_H, k) = \max_{\sigma \in S_k, y \in \{-1, +1\}^m} \left| \left\{ t_j \in T : R^\sigma_{y_j} t_j \neq y_j \right\} \right|,
\]

where \( R^\sigma_{y_j} \in \{-1, +1\} \) is the label output by the decision rule \( R \) for task \( t \) when the true label vector is \( y \) and the adversary strategy is \( \sigma \). Note that our notation \( \text{Aff}(R, B_H, k) \) makes the dependence of the quality measure on the honest worker subgraph \( B_H \) and the number of adversaries \( k \) explicit.

We have the following result:

**Theorem 4 (Lower bound on number of affected tasks)** Suppose that \( |A| = k \) and let \( \text{PreIm}(T') \) denote the set of honest workers who label at least one task in \( T' \subseteq T \). Then, given any honest worker-task assignment graph \( B_H \), there exists an adversary strategy \( \sigma^* \in S_k \), that is independent of any decision rule \( R \in C \), such that

\[
L \leq \max_{y \in \{-1, +1\}^m} \text{Aff}(R, \sigma^*, y) \quad \forall R \in C,
\]

where

\[
L = \frac{1}{2} \max_{T' \subseteq T : |\text{PreIm}(T')| \leq k} |T'|,
\]

and \( \text{Aff}(R, \sigma^*, y) \) denotes the number of affected tasks under adversary strategy \( \sigma^* \), decision rule \( R \), and true label vector \( y \) (with the assumption that maximum over an empty set is zero).

We describe the main idea of the proof. The proof proceeds in two steps: (i) we provide an explicit construction of adversary strategy \( \sigma^* \) that depends only on \( B_H \) and (ii) we show the existence of 2 possible true labelings \( \tilde{y} \neq y \) such that \( R \) outputs exactly the same labels in both scenarios. The adversary labeling strategy we construct uses the idea of indistinguishability, which captures the fact that by carefully choosing their responses, the adversaries can render themselves indistinguishable from honest workers. In the simple case when there is only one honest worker, the adversary simply flips the response provided by the honest worker, so that each task will have two labels of opposite parity. It can be argued that since there is no other discriminatory information, it is impossible for any decision rule \( R \) to distinguish the honest worker from the adversary and hence identify the true label of any task (better than a random guess). We extend this to the general case, where the adversary “targets” atmost \( k \) honest workers and derives a strategy based on the subgraph of \( B_H \) restricted to the targeted workers. The resultant strategy can be shown to result in incorrectly identified labels for at least \( L \) tasks, for some ground-truth label assignment.
Note that Theorem 4 holds for any honest worker-task assignment graph $B_H$. This is particularly remarkable given that the analysis of aggregation algorithms becomes extremely complicated for general graphs (a fact observed in prior works; see Dalvi et al. (2013)).

The bound $L$ itself depends on the structure of $B_H$ and therefore can be hard to interpret in general. It, however, becomes interpretable for an $(r, \gamma, \alpha)$-bipartite expander, defined next.

**Definition 4** An honest worker-task assignment graph $B_H = (H \cup T, E)$, with edges from the honest workers $H$ to the tasks $T$, is $(r, \gamma, \alpha)$-bipartite expander if: (i) $B_H$ is $r$-right-regular, i.e. each task is labeled by $r$ honest workers and (ii) for all $T' \subseteq T$ such that $|T'| \leq \gamma |T|$, the pre-image of $T'$ satisfies $|\text{PreIm}(T')| \geq \alpha |T'|$, where $\text{PreIm}(T')$ is the set of all honest workers who label at least one task in $T'$.

Note that the definition entails that $\alpha \leq r$. We have the following corollary of Theorem 4 when $B_H$ is $(r, \gamma, \alpha)$-bipartite expander.

**Corollary 5 (Lower bound for expanders)** Suppose $B_H$ is $(r, \gamma, \alpha)$-bipartite expander, then $k$ adversary identities can affect at least $L$ tasks such that $\frac{k}{2r} \leq L \leq \frac{k}{2\alpha}$, provided $\gamma |T| > k/\alpha$. Further, given any constant $r$, there exists $\gamma > 0$ such that a uniformly random $B_H$ is $(r, \gamma, r - 2)$-bipartite expander, so that the lower bound $L \approx \frac{k}{2r}$.

The proof is provided in Appendix A.4. The above statement says that if the honest worker-task assignment graph $B_H$ is randomly constructed, then $k$ adversary identities can affect at least $k/(2r)$ tasks. The bound implies that the ability of the adversaries to affect the tasks increases linearly as the number of identities $k$ increases. Further, the damage that $k$ adversaries can do decreases inversely with the number of honest workers $r$ who provide labels for each task. Both implications are intuitive. As can be seen from the proof, the lower bound $k/(2r)$ on $L$ in Corollary 5 holds for all $r$-right-regular graphs, even if they are not expanders.

### 4.2.2 Accuracy of the hard-penalty algorithm

We now analyze the accuracy of the hard-penalty algorithm when there are $k$ sophisticated adversaries. We focus on the hard-penalty algorithm because it can be seen that the soft-penalty algorithm is vulnerable to sophisticated adversaries. In particular, suppose $B_H$ is $(r, \gamma, \alpha)$-bipartite expander, there are $k = cr$ adversaries, for some $c > 1$, and all honest workers are completely reliable (i.e., $\mu_i = 1$ for each honest worker $h_i$). Suppose each adversary provides the incorrect response to all the $m$ tasks. Then, every task has $r$ correct responses, all provided by honest workers, and $k$ incorrect responses, all provided by adversaries, resulting in penalties of $1/r$ for each honest worker and $1/k$ for each adversary (note that the degree of the workers does not affect the penalty because the penalty received from each task is the same). Because $k > r$, the adversaries receive lower penalties than the honest workers. As a result, filtering out $k$ workers with the highest penalties will always filter out the honest workers. Furthermore, natural aggregation algorithms (simple majority or weighted simple majority with penalties as weights) result in incorrect labels for all the tasks. Therefore, under the soft-penalty algorithm, $k = cr$ adversaries can affect all the tasks, which can be much larger than the lower bound $L \approx k/(2r) = c/2$. 

15
For our analysis, we focus on the penalty-based aggregation algorithm (see Algorithm 3), which is a natural extension of the hard-penalty algorithm to also perform label aggregation:

**Algorithm 3** PENALTY-BASED AGGREGATION  
1: **Input:** $W$, $T$ and $L$  
2: Perform steps 2, 3 of the hard-penalty algorithm  
3: For each task $t_j$, let $w_{t_j}^+$, $w_{t_j}^-$ be worker nodes that task nodes $t_j^+$, $t_j^-$ are respectively mapped to in optimal semi-matching $M$ in Step 2  
4: **Output**  

\[
\hat{y}_j = \begin{cases} 
+1 & \text{if } \deg_M(w_{t_j}^+) < \deg_M(w_{t_j}^-) \\
-1 & \text{if } \deg_M(w_{t_j}^+) > \deg_M(w_{t_j}^-) \\
\left\{-1, +1\right\} & \text{otherwise}
\end{cases}
\]

(here $\hat{y}_j$ refers to output label for task $t_j$ and $\left\{-1, +1\right\}$ means $\hat{y}_j$ is drawn uniformly at random from $\left\{-1, +1\right\}$)

**Theorem 6 (Hard-penalty algorithm with sophisticated adversaries)** Suppose that $|A| = k$ and $\mu_i = 1$ for each honest worker, i.e. an honest worker always provides the correct label. Further, let $d_1 \geq d_2 \geq \cdots \geq d_{|H|}$ denote the degrees of the honest workers in the optimal semi-matching on $B_H$. For any true labeling $y$ of the tasks and under the penalty-based label aggregation algorithm (with the convention that $d_i = 0$ for $i > |H|$):  

1. There exists an adversary strategy $\sigma^*$ such that the number of affected tasks is atleast $
\frac{1}{2} \sum_{i=1}^{k-1} d_i$. 

2. No adversary strategy can affect more than $U$ tasks where 

(a) $U = \sum_{i=1}^{k} d_i$, when all but (atmost) one adversary are required to provide incorrect responses  

(b) $U = \sum_{i=1}^{2k} d_i$, in the general case

A few remarks are in order. First, it can be shown that for optimal semi-matchings, the degree sequence $d_1, d_2, \ldots, d_{|H|}$ is unique (see the proof in Appendix A.6) and therefore, the bounds in the theorem above are uniquely defined given $B_H$. Also, the assumption that $\mu_i = 1$ is required for analytical tractability; proving theoretical guarantees in crowdsourced settings (even without adversaries) for general graph structures is notoriously hard (Dalvi et al., 2013). The result of Theorem 6 provides both a lower and upper bound for the number of tasks that can be affected by $k$ adversaries under the penalty-based aggregation algorithm. Our characterization is reasonably tight when all but one adversary are required to provide incorrect responses; in this case, the gap between the upper and a constant factor of the lower bound is $d_k$, which can be “small” for $k$ large enough. However, our characterization is loose in the general case when adversaries can provide arbitrary responses; here the gap is $\sum_{i=k}^{2k} d_i$ which we attribute to our proof technique and conjecture that the upper bound of $\sum_{i=1}^{k} d_i$ also applies to the more general case.
Optimality of Penalty-based Aggregation. We now compare the upper bound $U$ in Theorem 6 to the lower bound $L$ in Theorem 4. We show that (see Appendix A.7) when the degrees $d_1, d_2, \ldots, d_{|H|}$ are all distinct, $L \geq \sum_{i=1}^{k-1} d_i$, which combined with theorem 4 shows that $k$ adversaries can affect at least $\frac{1}{2} \sum_{i=1}^{k-1} d_i$ tasks, irrespective of the label aggregation algorithm used to aggregate the worker responses. We also have from Theorem 6 that under the penalty-based aggregation algorithm, $k$ adversaries can affect at most $U = \sum_{i=1}^{2k} d_i \leq 3(\sum_{i=1}^{k-1} d_i)$ (as long as $k \geq 2$). Therefore, our algorithm achieves constant factor optimality in recovering the true labels of the tasks, irrespective of the structure of the honest worker-task assignment graph and the adversary strategy.

5. Experiments

In this section, we evaluate the performance of our reputation algorithms in recovering the true labels of tasks as well as identifying adversaries, on both synthetic and real datasets. We consider the following popular label aggregation algorithms as benchmarks: (a) simple majority algorithm MV (b) the EM algorithm (Dawid and Skene, 1979; Raykar and Yu, 2012) (c) the KOS algorithm (Karger et al., 2014) and (d) KOS(NORM), a normalized version of the KOS algorithm that is amenable for arbitrary assignment graphs (KOS is designed for random regular graphs), and compare their accuracy in inferring the true labels of the tasks, in isolation and in conjunction with our reputation algorithms. For the precise details of the benchmarks, we refer the reader to Appendix B. We implemented iterative versions of Algorithms 1(SOFT) and 2(HARD), where in each iteration we filter out the worker with the highest penalty and recompute penalties for the remaining workers. We also implemented a third version called USOFT (unnormalized soft) which is identical to the SOFT penalty algorithm except that it does not normalize the penalties by the worker degrees (refer to the discussion in section 3.3).

We first describe the results for real data.

5.1 Real Data

Our objective for the real data evaluation is to demonstrate the improvements in the accuracy of the benchmark algorithms from the use of our reputation algorithms. We considered the following standard datasets: (a) TREC (Tang and Lease, 2011): a collection of topic-document pairs labeled as relevant or non-relevant by workers on Amazon Mechanical Turk (AMT) as part of the TREC 2011 crowdsourcing track. We considered two versions of this dataset: stage2 and task2. (b) NLP (Snow et al., 2008): consists of annotations by AMT workers for different NLP tasks; again we have two versions: (1) rte - which provides binary judgments for textual entailment, i.e. whether one sentence can be inferred from another (2) temp – which provides binary judgments for temporal ordering of events. (c) Bluebird (Welinder et al., 2010) contains judgments differentiating two kinds of birds in an image. The datasets stage2 and task2 contained many workers who provided responses for only a single task – we filtered out workers with less than 3 responses. We ran the benchmark algorithms on the entire dataset and again after (iteratively) removing the 10 workers with the highest penalties, as computed by our reputation algorithms. Table 1 reports the accuracy achieved by the benchmarks in isolation as well as the best accuracy
<table>
<thead>
<tr>
<th>Dataset</th>
<th>MV Base</th>
<th>MV Soft</th>
<th>MV Hard*</th>
<th>EM Base</th>
<th>EM Soft</th>
<th>EM Hard*</th>
<th>KOS Base</th>
<th>KOS Soft</th>
<th>KOS Hard*</th>
<th>KOS(NORM) Base</th>
<th>KOS(NORM) Soft</th>
<th>KOS(NORM) Hard*</th>
</tr>
</thead>
<tbody>
<tr>
<td>rte</td>
<td>91.9</td>
<td>92.1(8)</td>
<td>92.6(3)</td>
<td>93.0</td>
<td>93.0</td>
<td>93.3(5)</td>
<td>49.7</td>
<td>88.8(9)</td>
<td>91.6(10)</td>
<td>91.3</td>
<td>92.7(7)</td>
<td>93.1(6)</td>
</tr>
<tr>
<td>temp</td>
<td>93.9</td>
<td>93.9</td>
<td>94.6(3)</td>
<td>94.1</td>
<td>94.1</td>
<td>94.4(3)</td>
<td>56.9</td>
<td>69.3(4)</td>
<td>93.7(3)</td>
<td>93.9</td>
<td>94.3(7)</td>
<td>94.4(1)</td>
</tr>
<tr>
<td>bluebird</td>
<td>75.9</td>
<td>75.9</td>
<td>75.9</td>
<td>89.8</td>
<td>89.8</td>
<td>89.8</td>
<td>72.2</td>
<td>75.9(3)</td>
<td>75.9(3)</td>
<td>72.2</td>
<td>75.9(3)</td>
<td>75.9(3)</td>
</tr>
<tr>
<td>stage2</td>
<td>74.3</td>
<td>75.3(4)</td>
<td>80.7(3)</td>
<td>70.2</td>
<td>76.8(10)</td>
<td>81.3(6)</td>
<td>74.5</td>
<td>74.4</td>
<td>75.3(3)</td>
<td>75.5</td>
<td>76.9(1)</td>
<td>78.2(2)</td>
</tr>
<tr>
<td>task2</td>
<td>64.2</td>
<td>64.2</td>
<td>67.8(10)</td>
<td>67.0</td>
<td>67.1(5)</td>
<td>68.6(9)</td>
<td>57.4</td>
<td>57.4</td>
<td>65.6(10)</td>
<td>58.3</td>
<td>58.8(4)</td>
<td>67.7(9)</td>
</tr>
</tbody>
</table>

Table 1: **Percentage accuracy** in recovering true labels of benchmark algorithms in isolation and when combined with our reputation algorithms - Hard* denotes the best of HARD and USOFT. For each benchmark, the best performing combination is highlighted in bold. The number in the parentheses represents the number of workers filtered by our reputation algorithm (an absence indicates that no performance improvement was achieved while removing up to 10 workers with the highest penalties). The last row reports the average accuracy across the datasets.

when removing up to 10 workers. The USOFT and HARD versions of our reputation algorithms achieved similar performance, and we report the better outcome amongst the two under the “Hard” column in the table.

The main conclusion we draw is that our reputation algorithms are able to boost the performance of state-of-the-art aggregation algorithms by a significant amount across the datasets: the average improvement in accuracy for MV and EM is around 3%, for KOS(NORM) is 4.7% while for KOS it is 29.4%, when using the hard penalty-based reputation algorithm. The improvement is large for KOS since it is designed for regular graphs and suffers in performance on real world graphs. Second, we can elevate the performance of simpler algorithms such as KOS and MV to the levels of the more general (and in some cases, complicated) EM algorithm. The KOS algorithm for instance, is not only easier to implement, but also has tight theoretical guarantees when the underlying assignment graph is sparse random regular and further is robust to different initializations (Karger et al., 2014). The modified version KOS(NORM) can be used in graphs where worker degrees are skewed, but we are still able to enhance its accuracy. Our results provide evidence for the fact that existing random models are inadequate in capturing the behavior of workers in real-world datasets, necessitating the need for a more general approach. Third, note that all the variants of our algorithm either improve or match the performance of the base algorithm, with the hard penalty-based variant often resulting in the most improvement. Because the workers filtered out by the hard-penalty algorithm are likely to have higher degrees, our results indicate that the adversarial workers in the datasets tend to label a large number of tasks.

Finally, our algorithm was successful in identifying the following types of workers in the real datasets: (1) uniform workers who always label +1 or −1 in temp, task2, stage2, (2) low reliability workers, i.e. \( \mu_i \leq 0.2 \) in bluebird, task2, stage2, rte, temp, and (3) workers who label each task independent of its true label in task2, rte, temp, bluebird.
Identifying Unreliable and Adversarial Workers

(such workers were defined as “spammers” in previous work (Raykar and Yu, 2012). Note that uniform workers are a subset of this class of workers). Therefore, our penalty-based algorithms are able to identify a broad set of adversary strategies in addition to those detected by existing techniques.

5.2 Synthetic Data

Our objective for the simulation experiments is to study the performance of our reputation algorithms as a function of (a) the structure of the worker-task assignment graph and (b) the adversary labeling strategy. In many practical scenarios, the worker-task assignment graph can be formed organically where the workers decide which tasks to label on. To capture these settings, we focus on the Preferential Attachment random graph (Guillaume and Latapy, 2006) model, which results in a skewed (specifically power-law) distribution of worker degrees that is characteristic of many real-world crowdsourcing scenarios (Franklin et al., 2011). Our results demonstrate that our method can result in up to 16% improvement in accuracy, complementing the theoretical results Section 4 for \((l,r)\)-regular graphs.

For our study, we considered two types of adversaries: (a) spammers – who label each task +1 or −1 with prob. \(1/2\) and (b) uniform – who label +1 on all assigned tasks. These strategies are simple and commonly observed in practice (Vuurens et al., 2011).

The broad simulation methodology is as follows: (a) generate a random instance from the ground-truth model class; (b) generate synthetic responses from the workers for a sample of tasks; (c) compute the true labels of the tasks according to different benchmarks; (d) filter out workers with the highest penalties according to our reputation algorithms (e) recompute true labels according to different benchmarks; and (f) compare the accuracy in recovering the true labels as well as in identifying the adversarial workers.

**Setup.** We considered a total of \(n = 100\) workers. The parameter \(q \in (0.5, 1)\) specifies the probability that a worker is honest, we chose \(q = 0.7\) so that in expectation we have 30 adversaries among the 100 workers. The task true labels were sampled from a population with prevalence \(\gamma = 0.5\). The worker degrees were sampled according to a power-law distribution and then we employed the Python networkx\(^5\) library (Hagberg et al., 2008) to generate the worker-task assignment graph. Note that the number of tasks \(m\) is automatically determined as an outcome of the graph generation process. Since the worker degrees are skewed, the accuracy of the algorithms is influenced by the degrees of the adversaries. We considered two scenarios: (a) adversaries have high degrees and (b) adversaries have low degrees. We biased the degrees of the adversaries by biasing the parameter \(q\) when generating ground-truth honest and adversarial workers (refer to Appendix B for precise details). Then, for each adversary strategy, we randomly generated 300 instances as follows: (a) generate a random worker-task assignment graph; (b) classify workers as honest and adversarial; (c) sample the task true labels \(t_j \in \{-1, +1\}\); (d) for each honest worker \(h_i\), sample reliability \(\mu_i\) u.a.r from the interval \([0.6, 1.0]\); (e) generate responses from honest workers according to the sampled reliabilities; (f) generate responses from adversarial workers based on the strategy.

\(^{5}\) https://networkx.github.io/documentation/latest/reference/generated/networkx.algorithms.bipartite.generators.preferential_attachment_graph.html
<table>
<thead>
<tr>
<th></th>
<th>Spammer</th>
<th>Uniform</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low Deg</td>
<td>High Deg</td>
</tr>
<tr>
<td>MV</td>
<td>84.71</td>
<td>77.29</td>
</tr>
<tr>
<td>MV-SOFT</td>
<td><strong>85.30</strong></td>
<td>76.81</td>
</tr>
<tr>
<td>MV-HARD</td>
<td>83.91</td>
<td>78.15</td>
</tr>
<tr>
<td>MV-USOFT</td>
<td>82.88</td>
<td><strong>78.65</strong></td>
</tr>
<tr>
<td>EM</td>
<td>81.42</td>
<td>70.32</td>
</tr>
<tr>
<td>EM-SOFT</td>
<td><strong>82.30</strong></td>
<td>70.43</td>
</tr>
<tr>
<td>EM-HARD</td>
<td>80.57</td>
<td>70.64</td>
</tr>
<tr>
<td>EM-USOFT</td>
<td>78.93</td>
<td>71.21</td>
</tr>
<tr>
<td>KOS</td>
<td>79.76</td>
<td>62.69</td>
</tr>
<tr>
<td>KOS-SOFT</td>
<td>78.76</td>
<td>62.60</td>
</tr>
<tr>
<td>KOS-HARD</td>
<td>78.17</td>
<td>64.50</td>
</tr>
<tr>
<td>KOS-USOFT</td>
<td>75.96</td>
<td><strong>66.46</strong></td>
</tr>
<tr>
<td>KOS(NORM)</td>
<td>76.60</td>
<td>63.30</td>
</tr>
<tr>
<td>KOS(NORM)-SOFT</td>
<td><strong>79.32</strong></td>
<td>67.41</td>
</tr>
<tr>
<td>KOS(NORM)-HARD</td>
<td>76.03</td>
<td>64.05</td>
</tr>
<tr>
<td>KOS(NORM)-USOFT</td>
<td>75.60</td>
<td><strong>66.29</strong></td>
</tr>
<tr>
<td>REL-SOFT</td>
<td>0.28</td>
<td>0.16</td>
</tr>
<tr>
<td>REL-HARD</td>
<td>0.52</td>
<td>0.14</td>
</tr>
<tr>
<td>REL-USOFT</td>
<td>0.47</td>
<td>0.06</td>
</tr>
</tbody>
</table>

Table 2: **Percentage accuracy** in recovering true labels of different algorithms for Preferential Attachment random graphs (refer to text for details of notation). We also report the average reliability of the 15 filtered workers for each of our algorithms (bottom 3 rows). The columns specify the two types of adversaries with Low Deg/High Deg indicating the bias for adversarial worker degrees. The numbers reported are an average over 300 experimental runs. For each benchmark algorithm, the combination with the highest accuracy is highlighted in bold. For some scenarios, none of the combinations are bold - this means that there was no statistically significant difference ($p < 0.1$) between the combinations.

For each instance, we computed the true labels of the tasks according to each of the benchmark algorithms. We then removed 15 workers with the largest penalties as determined by each of our reputation algorithms, and recomputed the true labels. To measure the success in identifying adversarial workers, we report the average reliability of the filtered workers, which we can compute because we have access to ground truth task labels. The theoretical results in section 4.1 showed that the soft-penalty algorithm filters out workers with low reliabilities for the case of $(l,r)$-regular graphs, and so we expect the average reliability of the filtered workers to be low. Further, since both spammers and uniform
workers have reliability close to zero⁶ and honest workers have high reliability (in the range \([0.6, 1.0]\)), a low average reliability of the filtered workers means that most of them are adversaries. Of course, since we only have finitely many labels from the adversaries, the actual reliabilities will be different from zero.

**Results.** Table 2 reports the results, averaged over the 300 instances. There are 4 columns: 2 for each adversary strategy, where Low Deg and High Deg denote the biasing of adversarial worker degrees. The rows MV, EM, KOS and KOS(NORM) correspond to the accuracies obtained by the four benchmark algorithms when using the responses of all workers. The remaining rows correspond to the accuracies obtained after filtering out the responses of 15 workers with the largest penalties. For instance, MV-SOFT is the accuracy when workers are filtered using the SOFT-penalty algorithm and responses are aggregated using the MV algorithm. Similarly, for all the other combinations. The last three rows report the average reliability of the filtered workers under each reputation algorithm.

We draw the following conclusions. First, our reputation-based filtering approach improves the accuracy of the benchmark algorithms – up to 16% in some scenarios – in 13 out of 16 cases, where each case is a combination of adversary strategy, bias of adversarial worker degrees and a label aggregation algorithm. In particular, we always improve the performance of the MV algorithm and achieve the largest improvements for the KOS benchmarks. In two of the remaining three cases, our algorithm matches the performance of the baseline algorithm. Only in one case (when the adversaries are low degree spammers and when the label aggregation algorithm is KOS), filtering workers using our algorithm results in marginally worse accuracy than the baseline algorithm. As expected, we notice larger improvements when the adversaries have higher degrees, and hence higher influence on the label outcome, than when they have lower degrees. Note that we are able to improve the accuracy despite using labels from fewer workers because we (mostly) discard the adversary labels, which do not provide any information about the underlying task true labels. Second, the USOFT and HARD versions of our algorithm have similar performance confirming our intuition in Section 3.3. However, the USOFT algorithm is much faster since it does not require the (expensive) computation of an optimal semi-matching and therefore, can serve as a viable alternative to HARD penalty for large-scale settings.

Third, we see that the SOFT-penalty algorithm is able to achieve higher accuracy (for each of the benchmarks) by filtering low degree adversaries while USOFT (and HARD) improve performance by filtering high degree adversaries. We attribute this difference to the fact that the SOFT-penalty algorithm normalizes by worker degree as opposed to the HARD and USOFT versions. This insight can be used to aid the choice of the reputation algorithm to be employed as a function of expected adversary degrees. For instance, adversaries adopting simple strategies (such as spammers and uniform workers) may be expected to have higher degrees (because they spend less effort per task); in this case, HARD and USOFT may be used. Finally, this observation is also reflected in the average reliability of filtered workers. In particular, the SOFT-penalty algorithm is able to better identify low degree adversaries than the HARD and USOFT versions, since the latter two do not normalize by the degrees and tend to penalize workers with larger degrees (honest workers in this case) more. However, for high degree adversaries, we see that HARD and USOFT outperform the SOFT-penalty algorithm.

---

⁶ This is true for uniform workers since \(\gamma = 0.5\)
algorithm. The average reliability for the case of uniform workers is higher since some honest workers are also filtered out, which is expected from Theorem 2. These findings establish that our reputation algorithms also perform well when worker-task assignment graphs are non-regular, complementing the theoretical results (Theorems 1, 2 and 3) for regular graphs.

6. Conclusions

This paper studies the problem of inferring true labels of tasks using crowdsourced responses against a broad class of adversarial labeling strategies. Our main contribution is the design of a reputation-based worker filtering algorithm that uses a combination of disagreement-based penalties and optimal semi-matchings to identify adversarial workers, using only the submitted responses. We theoretically establish that our reputation scores are consistent with the existing notion of worker reliabilities and can further identify workers providing deterministic, spammy or even malicious responses. To address worst-case adversarial behavior, we introduce the notion of sophisticated adversaries and establish bounds on the minimum “damage” achievable by such workers. Empirically, we show that our algorithm can be applied in real crowdsourced datasets to significantly enhance the accuracy of existing label aggregation algorithms.

To the best of our knowledge, our work is the first to consider general worker strategies in crowdsourced labeling tasks and we hope that future work can extend our analysis to more complex adversary strategies. Both of our penalty-based algorithms rely on task labels being binary, it would be interesting to explore whether similar techniques can be extended for multi-class settings. One issue with our approach is that we throw away responses from adversarial workers so that each task now has a fewer number of responses. Adaptive techniques for recruiting more workers (Ramesh et al., 2012) as well as identifying the right workers (Li et al., 2014) can help in alleviating the issue and presents an opportunity for future work. Finally, applying our outlier detection technique to ensemble learning approaches, where outputs from multiple learning algorithms are combined, could be a promising future direction; indeed there has already been some work in this space (Wang and Yeung, 2014).

References


Appendix A. Proofs of the theorems

We first state a lemma that will be useful for the theorem proofs. We assume that each worker $w_i$ has a reliability $\mu_i$, sampled from a population with average reliability $\mu$, and each task $t_j$ has true label $y_j$.

**Lemma A.0.1** For a given realization of the worker reliabilities and task true labels, we have the following:

$$
E[d^+_j \mid y_j = +1] = \frac{r}{2} + \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}
$$

$$
E[d^+_j \mid y_j = -1] = \frac{r}{2} - \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}
$$

where the expectation is over the randomness in generation of the worker responses. Furthermore, we must have

$$
E[d^+_j \mid y_j = +1, w_i(t_j) = +1] = \frac{1 - \mu_i}{2} + \frac{r}{2} + \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}
$$

$$
E[d^+_j \mid y_j = -1, w_i(t_j) = +1] = \frac{1 + \mu_i}{2} + \frac{r}{2} - \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}
$$

**Proof** We proceed as follows.

$$
E[d^+_j \mid y_j = +1] = \sum_{i' \in W_j} \Pr(w_{i'}(t_j) = +1 \mid y_j = +1) = \sum_{i' \in W_j} p_{i'} = \sum_{i' \in W_j} \frac{1 + \mu_{i'}}{2}
$$

$$
= \frac{|W_j|}{2} + \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}
$$

$$
= \frac{r}{2} + \frac{1}{2} \sum_{i' \in W_j} \mu_{i'}.
$$

It can be shown in a similar fashion that $E[d^+_j \mid y_j = -1] = r/2 - (\sum_{i' \in W_j} \mu_{i'})/2$. 
Coming to the next set of equalities, we have

$$
\mathbb{E}[d^+_j \mid y_j = +1, w_i(t_j) = +1] = 1 + \sum_{\hat{v} \in W_j, \hat{v} \neq i} \mathbb{E}[1[w_{i'}(t_j) = +1] \mid w_i(t_j) = +1, y_j = +1]
$$

$$
= 1 + \sum_{\hat{v} \in W_j, \hat{v} \neq i} \mathbb{E}[1[w_{i'}(t_j) = +1] \mid y_j = +1]
$$

$$
= 1 + \sum_{\hat{v} \in W_j, \hat{v} \neq i} \Pr(w_{i'}(t_j) = +1 \mid y_j = +1)
$$

$$
= 1 + \sum_{\hat{v} \in W_j, \hat{v} \neq i} p_{i'}
$$

$$
= 1 - p_i + \sum_{\hat{v} \in W_j} p_{i'}
$$

$$
= \frac{1 - \mu_i}{2} + \sum_{\hat{v} \in W_j} p_{i'},
$$

where the second inequality follows from the conditional independence of $1[w_i(t_j) = +1]$ and $1[w_{i'}(t_j) = +1]$ given true label $y_j$ for $i \neq i'$. The desired expression is then obtained by noting that $\sum_{\hat{v}' \in W_j} p_{\hat{v}'} = \mathbb{E}[d^+_j \mid y_j = +1]$. The derivation of the expression of $\mathbb{E}[d^+_j \mid y_j = -1, w_i(t_j) = +1]$ follows from a symmetric argument. The result of the lemma now follows.

\[\blacksquare\]

### A.1 Proof of Theorem 1

For a given realization of worker reliabilities, task true labels, and worker responses to tasks, let $s_{ij}$ denote the penalty allocated by task $t_j$ to worker $w_i$. It follows from the soft-penalty algorithm that (recall notation from section 2)

$$
1/s_{ij} = d^+_j 1[w_i(t_j) = +1] + d^-_j 1[w_i(t_j) = -1]
$$

$$
= (d^+_j - d^-_j) 1[w_i(t_j) = +1] + d^-_j
$$

$$
= (2d^+_j - r) 1[w_i(t_j) = +1] + r - d^+_j
$$

First consider the case when $y_j = +1$. Now note that

$$
\mathbb{E}\left[1[w_i(t_j) = +1]d^+_j \mid y_j = +1\right] = \sum_{\hat{v} \in W_j} \mathbb{E}\left[1[w_i(t_j) = +1]1[w_{i'}(t_j) = +1] \mid y_j = +1\right]
$$

$$
= \Pr[w_i(t_j) = +1 \mid y_j = +1] \left(1 + \sum_{\hat{v}' \in W_j, \hat{v}' \neq i} \Pr[w_{i'}(t_j) = +1 \mid y_j = +1]\right)
$$

$$
= \Pr[w_i(t_j) = +1 \mid y_j = +1] \left(1 + \sum_{\hat{v}' \in W_j, \hat{v}' \neq i} \Pr[w_{i'}(t_j) = +1 \mid y_j = +1]\right)
$$

28
where we have used the fact that the random variables $1[w_i(t_j) = +1]$ and $1[w_{i'}(t_j) = +1]$ for $i \neq i'$ are conditionally independent given $y_j$. It then follows from lemma A.0.1 that

$$E[1/s_{ij} | y_j = +1] = \left(2E[d_j^+ | y_j = +1, w_i(t_j) = +1] - r\right) \Pr(w_i(t_j) = +1 | y_j = +1) + r - E[d_j^+ | y_j = +1]$$

$$= \left(1 - \mu_i + r + \sum_{i' \in W_j} \mu_{i'} - r\right) \frac{(1 + \mu_i)/2 + r/2 - 0.5 \sum_{i' \in W_j} \mu_{i'}}{1}.$$ 

The above expression can be simplified to get

$$E[1/s_{ij} | y_j = +1] = E[1/s_{ij} | y_j = +1] - E[1/s_{ij} | y_j = -1].$$

We now make a few approximations based on the concentration of the sum of independent random variables around its mean. In particular, since $|W_j| = r \rightarrow \infty$, it follows (from say Hoeffding’s inequality) that

$$\sum_{i' \in W_j} \mu_{i'} \approx E[\mu_i] = \mu,$$

where we use the notation $X \approx y$ to denote that random variable $X$ concentrates around quantity $y$. We thus have that

$$E[1/s_{ij} | y_j = +1] \approx E[1/s_{ij} | y_j = +1] \approx r/2 + (1 - \mu_i^2)/2 + 0.5r \mu_i \mu.$$ 

In a similar fashion, in the case when $y_j = -1$, we can write (using the sequence of steps above)

$$E[1/s_{ij} | y_j = -1] = \left(2E[d_j^- | y_j = -1, w_i(t_j) = +1] - r\right) \Pr(w_i(t_j) = +1 | y_j = -1) + r - E[d_j^- | y_j = -1]$$

$$= \left(1 + \mu_i + r - \sum_{i' \in W_j} \mu_{i'} - r\right) \frac{(1 + \mu_i)/2 + r/2 + 0.5 \sum_{i' \in W_j} \mu_{i'}}{1}$$

$$= r/2 + (1 - \mu_i^2)/2 + 0.5r \mu_i \mu$$

$$\approx r/2 + (1 - \mu_i^2)/2 + 0.5r \mu_i \mu$$

where we used the concentration approximation in the last step.

Next, recall that $1/s_{ij} = d_j^+ 1[w_i(t_j) = +1] + d_j^- 1[w_i(t_j) = -1]$ and $d_j^+, d_j^-$ are sums over $r$ random variables. Since $r \rightarrow \infty$ we can again apply Hoeffding’s inequality (after conditioning on true label $y_j$) to get:

$$1/s_{ij} | y_j = +1 \approx E[1/s_{ij} | y_j = +1] \quad \text{and} \quad 1/s_{ij} | y_j = -1 \approx E[1/s_{ij} | y_j = -1].$$
Combining all of the above, we have that the net penalty $\text{pen}(h_i)$ for an honest worker $h_i$ concentrates around:

$$
\text{pen}(h_i) = \frac{1}{|T_i|} \sum_{j \in T_i} s_{ij} \approx \frac{1}{|T_i|} \sum_{j \in T_i} \frac{1}{\mathbb{E}[1/s_{ij}]}
$$

$$
= \frac{1}{|T_i|} \sum_{j \in T_i} \frac{1}{\left(\mathbb{E}[1/s_{ij} | y_j = +1] \Pr[y_j = +1] + \mathbb{E}[1/s_{ij} | y_j = -1] \Pr[y_j = -1]\right)}
$$

$$
= \frac{1}{|T_i|} \sum_{j \in T_i} \frac{\gamma + 1 - \gamma}{r/2 + (1 - \mu_i^2)/2 + 0.5r\mu_i\mu}
$$

where recall that $T_i$ denotes the set of tasks that worker $h_i$ is assigned. Thus the penalty received by honest worker $h_i$ concentrates around $\frac{1}{\mathbb{E}[1/s_{ij}]}$, where the function $g(\cdot)$ is defined as $g(x) = r/2 + (1 - x^2)/2 + r\mu x/2$. Note that $g'(x) = -x + r\mu/2$. Thus, for $r$ large enough and $\mu > 0$, we must have that $g'(x) > 0$ for any $x \in [-1, 1]$. Thus, the function $g(\cdot)$ is increasing on the domain $[-1, 1]$, from which we can conclude that the penalty decreases with the increase in the reliability of the worker. The result of the theorem now follows.

A.2 Proof of Theorem 2

The proof is similar to that of Theorem 1 above. First, we need to compute $\mathbb{E}[d_{j}^+ | y_j = +1]$ for any task $t_j$. In particular, we can write

$$
d_{j}^+ = \sum_{a \in A_j} 1[a(t_j) = +1] + \sum_{w_i(t_j) = +1} 1[w_i(t_j) = +1]
$$

$$
= |A_j| + \sum_{w_i(t_j) = +1} 1[w_i(t_j) = +1].
$$

where $H_j$ and $A_j$ denotes the set of honest workers and adversaries who label task $t_j$. Here we use the fact that adversaries always label +1. Observe that $\mathbb{E}[|A_j|] = (1 - q)r$ and $\mathbb{E}[|H_j|] = qr$, and following the sequence of arguments in lemma A.0.1 above:

$$
\mathbb{E}[d_{j}^+ | y_j = +1] \approx (1 - q)r + \frac{rq}{2}(1 + \mu).
$$

In a similar fashion, we can show that

$$
\mathbb{E}[d_{j}^+ | y_j = -1] \approx (1 - q)r + \frac{rq}{2}(1 - \mu).
$$

Similarly, for any honest worker $h_i$ we can write (see lemma A.0.1):

$$
\mathbb{E}[d_{j}^+ | y_j = +1, h_i(t_j) = +1] \approx (1 - q)r + \frac{1 - \mu_i}{2} + \frac{rq}{2}(1 + \mu)
$$

and

$$
\mathbb{E}[d_{j}^+ | y_j = -1, h_i(t_j) = +1] \approx (1 - q)r + \frac{1 + \mu_i}{2} + \frac{rq}{2}(1 - \mu).
$$
Note that \( \mathbb{E}[d_j^+ \mid y_j = +1] \) and \( \mathbb{E}[d_j^+ \mid y_j = -1] \) are respectively \( \mathbb{E}[1/s_{aj} \mid y_j = +1] \) and \( \mathbb{E}[1/s_{aj} \mid y_j = -1] \) for any adversary \( a \) that labels task \( t_j \). We can now compute the penalty assigned by task \( t_j \) to honest worker \( h_i \). Following the sequence of steps in the proof of theorem 1 above, we obtain:

\[
\mathbb{E}[1/s_{ij} \mid y_j = +1] \approx \left(2(1-q)r + 1 + \mu_i + rq + rq\mu - r\right)(1+\mu_i)/2 + r - (1-q)r - rq/2 - r\mu/2
\]

\[
= (1-q)r(1+\mu_i)/2 + (1-\mu_i^2)/2 + rq/2 + r\mu\mu_i/2
\]

\[
= (1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu + 1 - q)/2.
\]

In a similar fashion, we can show that

\[
\mathbb{E}[1/s_{ij} \mid y_j = -1] \approx \left(2(1-q)r + 1 + \mu_i + rq - r\mu - r\right)(1-\mu_i)/2 + r - (1-q)r - rq/2 + r\mu/2
\]

\[
= (1-q)r(1-\mu_i)/2 + (1-\mu_i^2)/2 + rq/2 + r\mu\mu_i/2
\]

\[
= (1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu - 1 + q)/2.
\]

Therefore, the total penalty received by an honest worker \( h_i \) is given by

\[
pen(h_i) = \frac{1}{|T_i|} \sum_{t_j \in T_i} s_{ij} \approx \frac{\gamma}{(1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu + 1 - q)/2} + \frac{(1-\gamma)}{(1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu - 1 + q)/2}
\]

(1)

On the other hand, the total penalty received by an adversary \( a \) is given by

\[
pen(a) = \frac{1}{|T_a|} \sum_{t_j \in T_a} s_{aj} \approx \frac{\gamma}{(1-q)r + rq(1+\mu)/2} + \frac{(1-\gamma)}{(1-q)r + rq(1-\mu)/2}
\]

(2)

In order to derive the condition on \( q \) for honest workers to receive lower penalties than adversaries, consider a perfect worker \( \hat{h}_i \) such that \( \mu_i = 1 \). For such a worker, we have

\[
pen(\hat{h}_i) \approx \frac{\gamma}{r/2 + r(q\mu + 1 - q)/2} + \frac{(1-\gamma)}{r/2 + r(q\mu - 1 + q)/2}
\]

Then, it follows that the penalty assigned to a perfect worker is less than that assigned to an adversary if and only if

\[
pen(\hat{h}_i) < pen(a)
\]

\[
\iff \frac{\gamma}{r/2 + r(q\mu + 1 - q)/2} + \frac{(1-\gamma)}{r/2 + r(q\mu - 1 + q)/2} < \frac{\gamma}{(1-q)r + rq(1+\mu)/2} + \frac{(1-\gamma)}{(1-q)r + rq(1-\mu)/2}
\]

\[
\iff q/2 + q\mu/2 > 1 - q/2 - q\mu/2 \iff q + q\mu > 1 \iff q > 1/(1+\mu).
\]

This establishes the condition on \( q \), the fraction of honest workers. Observe that this holds independent of the prevalence \( \gamma \). For the actual result, we make two observations:

1. \( pen(h_i) \) is decreasing in \( \mu_i \) as long as \( r \) is sufficiently large. This can be verified from the expression above.

2. When \( \mu_i = -1 \), it can be seen that \( pen(h_i) > pen(a) \) and when \( \mu_i = 1 \), we showed above that \( pen(h_i) < pen(a) \) as long as \( q > 1/(1+\mu) \).

The above two conditions guarantee that there exists threshold \( \delta_U \in (-1, 1) \) - depending on \( \mu, \gamma, q \) - such that \( \mu_i > \delta_U \iff pen(h_i) < pen(a) \). The result of the theorem now follows.
A.3 Proof of Theorem 3

The proof follows immediately from that of Theorem 2 above. In particular, if $\gamma > \frac{1}{2}$, then the Smart Strategy reduces to the Uniform Strategy, and in that case we have $\delta_S = \delta_U$. When $\gamma < \frac{1}{2}$, a symmetry argument shows that the penalty allocated to an adversary $a$ following the Smart Strategy is given by:

$$pen_S(a) \approx \frac{\gamma}{(1-q)r +rq(1-\mu)/2} + \frac{1-\gamma}{(1-q)r +rq(1+\mu)/2}$$

which is strictly lower than the penalty in equation (2). On the other hand, the penalty allocated to an honest worker $h_i$ in this case is:

$$pen_S(h_i) \approx \frac{\gamma}{(1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu - 1 + q)/2} + \frac{1-\gamma}{(1-\mu_i^2)/2 + r/2 + r\mu_i(q\mu + 1 - q)/2}$$

Again we can observe that $pen_S(h_i) > pen_S(a)$ when $\mu_i = -1$ and $pen_S(h_i) < pen_S(a)$ when $\mu_i = 1$ as long as $q > \frac{1}{1+\mu_i}$. Since $pen_S(h_i)$ is decreasing in $\mu_i$, this shows the existence of threshold $\delta_S \in (-1,1)$ such that $\mu_i > \delta_S \implies pen_S(h_i) < pen_S(a)$. Note that when $\mu_i < 0$, we have that $pen_S(h_i) > pen(h_i)$ where $pen(h_i)$ is given by equation (1). This shows that identifying adversarial workers adopting the smart strategy can be harder since low reliability honest workers receive higher penalties as opposed to when adversaries adopt the uniform strategy.

A.4 Proof of Theorem 4

We prove the result for the the case when there exists at least one subset $\mathcal{T}' \subseteq \mathcal{T}$ such that $\text{PreIm}(\mathcal{T}') \leq k$. Otherwise, the lower bound $L = 0$ by definition and the result of the theorem is trivially true.

Let $H^*$ denote the set $\text{PreIm}(\mathcal{T}^*)$ where

$$\mathcal{T}^* \overset{\text{def}}{=} \arg \max_{\mathcal{T}' \subseteq \mathcal{T} : |\text{PreIm}(\mathcal{T}')| \leq k} |\mathcal{T}'| .$$

For a given decision rule $R \in \mathcal{C}$, we construct an adversary strategy $\sigma^*$ under which at least $L$ tasks are affected for some true labeling of the tasks. Specifically for a fixed honest worker-task assignment graph $\mathcal{B}_H$ and ground-truth labeling $y$ of the tasks, consider the following adversary strategy (that depends on the obtained honest worker responses): letting $H^* = \{h_1, h_2, \ldots, h_{|H^*|}\}$ and the set of adversaries $A = \{a_1, a_2, \ldots, a_k\}$, we have (recall the notation in Section 2)

$$a_i(t) = \begin{cases} -h_i(t) & \text{if } t \in \mathcal{T}^* \\ h_i(t) & \text{otherwise} \end{cases} \forall i = 1, 2, \ldots, |H^*|$$

In other words, the adversaries label opposite to the honest workers in $H^*$ for tasks in $\mathcal{T}^*$ and agree with them for all other tasks. Note that since $|H^*| \leq k$ by construction, the above strategy is feasible. In addition, if $|H^*| < k$, then we only use $|H^*|$ of the $k$ adversary identities and not use the remaining. Let $\mathcal{L}$ denote the $n \times m$ labeling matrix obtained for this adversary strategy.
Now consider the scenario in which the true labels of all tasks in $\mathcal{T}^*$ are reversed, let this ground-truth be denoted by $\hat{y}$. Let $\tilde{h}(t)$ denote the response of honest worker $h$ for task $t$ in the new scenario. Since the honest workers provide their responses according to a probabilistic model, the following happens with non-zero probability:

$$
\tilde{h}(t) = \begin{cases} 
  h(t) & \forall t \notin \mathcal{T}^* \\
  -h(t) & \forall t \in \mathcal{T}^* \quad \forall h \in H
\end{cases}
$$

(3)

Correspondingly, according to the adversary labeling strategy $\sigma^*$ described above, the adversary responses would also change. In particular, using $\tilde{a}(t)$ to denote the adversary response in this scenario, we have

$$
\tilde{a}(t) = \begin{cases} 
  a(t) & \forall t \notin \mathcal{T}^* \\
  -a(t) & \forall t \in \mathcal{T}^* \quad \forall a \in A
\end{cases}
$$

(4)

Finally, let $\hat{\mathcal{L}}$ denote the labeling matrix corresponding to this new scenario. We now argue that $\hat{\mathcal{L}} = P\mathcal{L}$ for some $n \times n$ permutation matrix $P$. In order to see this, for any worker $w$ (honest or adversary), let $r(w)$ and $\tilde{r}(w)$ respectively denote the row vectors in matrices $\mathcal{L}$ and $\hat{\mathcal{L}}$. We show that $\hat{\mathcal{L}}$ can be obtained from $\mathcal{L}$ through a permutation of the rows. For that, first observe that for any honest worker $h \notin H^*$, we must have by definition of Prelm that $h(t) = 0$ for any $t \in \mathcal{T}^*$. Thus, it follows from (3) that $\tilde{h}(t) = -h(t) = 0 = h(t)$ for any $t \in \mathcal{T}^*$. Furthermore, $\tilde{h}(t) = h(t)$ for any $t \notin \mathcal{T}^*$ by (3). Therefore, we have that

$$
r(h) = \tilde{r}(h) \text{ for any } h \notin H^*.
$$

Next consider an honest worker $h_i \in H^*$ for some $i$. It can be argued that $r(h_i) = r(a_i)$. To see this, for any task $t \notin \mathcal{T}^*$, we have by (4) that

$$
\tilde{a}_i(t) = a_i(t) = h_i(t),
$$

where the second equality follows from our definition of the adversary strategy. Similarly, for any $t \in \mathcal{T}^*$, we have $\tilde{a}_i(t) = -a_i(t)$ by (4) and $a_i(t) = h_i(t)$ (by the adversary strategy). Hence, we must have $\tilde{a}_i(t) = h_i(t)$ for any $t \in \mathcal{T}^*$. Thus, we have shown that the rows $\tilde{r}(a_i) = r(h_i)$ for any $i$. Thus, $\hat{\mathcal{L}}$ is obtained from $\mathcal{L}$ by swapping rows corresponding to $h_i$ with $a_i$ for all $i$.

Now that we have shown $\hat{\mathcal{L}} = P\mathcal{L}$ for some permutation matrix $P$, it follows from the fact that $R \in \mathcal{C}$ that $R(\hat{\mathcal{L}}) = R(\mathcal{L})$. Thus, the label output by $R$ for all tasks in $\mathcal{T}^*$ is the same under both scenarios. As a result, it follows that $\text{Aff}(R, \sigma^*, y) + \text{Aff}(R, \sigma^*, \hat{y}) = |\mathcal{T}^*| = 2nL$, and therefore, either $\text{Aff}(R, \sigma^*, y) \geq L$ or $\text{Aff}(R, \sigma^*, \hat{y}) \geq L$.

This shows that $R$ has non-zero probability of outputting incorrect labels for at least $L$ tasks. Therefore, there exists some ground-truth labeling $y$ for which the number of affected tasks in $L$ is at least $L$, and since we take a maximum over all possible ground-truth labelings, the result of the theorem follows.

**Remark.** Any decision rule that outputs labels randomly in case of ties (i.e equal number of +1 and -1 responses) will achieve the lower bound $L$, including the simple majority decision rule.

A.4.1 Proof of Corollary 5

We first prove that $L \geq \frac{k}{2r}$. For simplicity, assume $k = cr$ for some constant $c > 1$. Since $\alpha \leq r$ we have that $|\mathcal{T}| \geq \gamma |\mathcal{T}^*| \geq \frac{k}{\alpha} \geq \frac{k}{r} = c$. Consider any $\mathcal{T}' \subseteq \mathcal{T}$ and suppose that $|\mathcal{T}'| = c$. Now, since $\mathcal{B}_H$ is $r$-right-regular, the pre-image of $\mathcal{T}'$ in $\mathcal{B}_H$ satisfies $|\text{PreIm}(\mathcal{T}')| \leq \frac{k}{2r}$.
$r|T'| = cr = k$. In other words, any subset of tasks of size $c$ has a pre-image of size at most $k$. By the definition of $L$, we have that $L \geq \frac{c}{2} = \frac{k}{2^r}$. 

For the upper-bound, again suppose that $k = e\alpha$ for some $e > 1$. Consider any $T' \subset T$ such that $|T'| \geq e$. Since $e = \frac{k}{\alpha} < \gamma |T|$, by the expander property we have that $|\text{PreIm}(T')| \geq \alpha |T'| = e\alpha = k$ and therefore, $L \leq \frac{c}{2} = \frac{k}{2^r}$. 

For the second part of the corollary, refer to Theorem 4.4 in Chapter 4 of [EXP].

### A.5 Proof of Theorem 6

Before we can prove the theorem, we need the following definitions and lemmas.

**Definition 5** A bipartite graph $G = (V_1, V_2, E)$ is termed **degenerate** if the following condition is satisfied:

$$|V_1| > |V_2|$$

**Definition 6** A bipartite graph $G = (V_1, V_2, E)$ is termed **growth** if the following condition is satisfied:

$$\forall V \subseteq V_1, \ |V| \leq |\text{Img}(V)|$$

where $\text{Img}(V) = \{v_2 \in V_2 \mid \exists v \in V \text{ s.t. } (v, v_2) \in E\}$, i.e. the set of neighboring nodes of $V$.

**Lemma A.5.1** Any bipartite graph can be decomposed into degenerate and growth sub-graphs where there are cross-edges only from the growth component to the degenerate component.

**Proof** Let $G = (V_1, V_2, E)$ be a given bipartite graph. Define $V^*$ to be the largest subset of $V_2$ such that $|V^*| > |\text{Img}_G(V^*)|$ where $\text{Img}_G$ denotes the image in the graph $G$. If no such $V^*$ exists then the graph is already growth and we are done. If $V^* \subseteq V_1$ then the graph is degenerate and again we are done. Else, we claim that the subgraph $J$ of $G$ restricted to $V_1 \setminus V^*$ on the left and $V_2 \setminus \text{Img}_G(V^*)$ on the right is growth. Suppose not, then there exists a subset $V'$ of nodes on the left such that $|V'| > |\text{Img}_J(V')|$ where $\text{Img}_J(V') \subseteq V_2 \setminus \text{Img}_G(V^*)$ denotes the image of $V'$ in the subgraph $J$. But then, we can add $V'$ to $V^*$ to get a larger degenerate sub-graph in $G$ which contradicts our choice of $V^*$. To see this, consider the set $V^* \cup V'$ on the left and $\text{Img}_G(V^*) \cup \text{Img}_J(V')$ on the right. We have $|V^* \cup V'| = |V^*| + |V'| > |\text{Img}_G(V^*)| + |\text{Img}_J(V')| = |\text{Img}_G(V^* \cup V')|$. Also, note that the only cross-edges are from $V_1 \setminus V^*$ to $\text{Img}_G(V^*)$. This shows that any bipartite graph can be decomposed into degenerate and growth sub-graphs with cross-edges only from the growth to the degenerate sub-graph. 

**Lemma A.5.2** Let $G = (V_1, V_2, E)$ be any bipartite graph and suppose that $M$ is any semi-matching on $G$. Further, let $J = (V_1', V_2', E')$ be a subgraph of $G$. Starting with $M' \subseteq M$, we can use algorithm $A_{SAM2}$ in $[OPT]$ to obtain an optimal semi-matching $N$ for the subgraph $J$. Let the nodes in $V_1$ be indexed such that $\deg_M(1) \geq \deg_M(2) \geq \ldots \deg_M(|V_1|)$ and
Identifying Unreliable and Adversarial Workers

indexed again such that \(\deg_N(1) \geq \deg_N(2) \geq \ldots \geq \deg_N(|V_1|)\). Then for any \(1 \leq s \leq |V_1|\), we have \(\sum_{i=1}^{s} \deg_N(i) \leq \sum_{i=1}^{s} \deg_M(i)\), i.e. the sum of the top \(s\)-degrees can only decrease as we go from \(M\) to \(N\).

**Proof** Note that if we restrict \(M\) to just the nodes \(V_2'\), we get a feasible semi-matching \(M'\) on the subgraph \(J\). Algorithm \(A_{SM2}\) proceeds by the iterated removal of cost-reducing paths. Note that when a cost-reducing path is removed, load is transferred from a node with larger degree (in the current semi-matching) to a node with strictly smaller degree. To see this, let \(P = (v_1^{(1)}, v_2^{(1)}, v_1^{(2)}, \ldots v_1^{(d)})\) be a cost-reducing path (see section 2.1 in [OPT]).

This means that \(\deg(v_1^{(1)}) > \deg(v_1^{(d)}) + 1\). When we eliminate the cost-reducing path \(P\), the degree of \(v_1^{(1)}\) decreases by 1 and that of \(v_1^{(d)}\) increases by 1, but still the new degree of \(v_1^{(d)}\) is strictly lower than the old degree of \(v_1^{(1)}\). In other words, if \(d_1^\text{bef} \geq d_2^\text{bef} \geq \ldots d_{|V_1|}^\text{bef}\) and \(d_1^\text{aft} \geq d_2^\text{aft} \geq \ldots d_{|V_1|}^\text{aft}\) be the degree-sequence before and after the removal of a cost-reducing path, then \(\sum_{i=1}^{s} d_i^\text{aft} \leq \sum_{i=1}^{s} d_i^\text{bef}\) for any \(1 \leq s \leq |V_1|\). Since this invariant is satisfied after every iteration of algorithm \(A_{SM2}\), it holds at the beginning and the end and we have

\[
\sum_{i=1}^{s} \deg_N(i) \leq \sum_{i=1}^{s} \deg_{M'}(i)
\]

Finally, observe that when we restrict \(M\) to only the set \(V_2'\), the sum of the top \(s\)-degrees can only decrease, i.e.

\[
\sum_{i=1}^{s} \deg_{M'}(i) \leq \sum_{i=1}^{s} \deg_M(i)
\]

Combining equations (5) and (6), the result follows.

**Notation for the proofs.** Let \(\mathcal{T}^+\) denote the set \(\{t^+ : t \in \mathcal{T}\}\), and similarly \(\mathcal{T}^-\) denote the set \(\{t^- : t \in \mathcal{T}\}\), these are “task copies”. Now partition the set of task copies \(\mathcal{T}^+ \cup \mathcal{T}^-\) as \(E \cup F\) such that for any task \(t\), if the true label is +1, we put \(t^+\) in \(E\) and \(t^-\) in \(F\), otherwise, we put \(t^-\) in \(E\) and \(t^+\) in \(F\). Thus, \(E\) contains task copies with true labels while \(F\) contains task copies with incorrect labels. Recall the conflict graph \(\mathcal{B}^{cs}\) constructed in Algorithm 2, where we similarly created two copies for each task. Now, since honest workers always provide the correct response, all honest workers have edges only to the set of task copies in \(E\), in \(\mathcal{B}^{cs}\). However, adversaries can have edges to task copies in both \(E\) and \(F\). In addition, it is easy to see that the subgraph of \(\mathcal{B}^{cs}\) restricted to honest workers \(H\) on the left and task copies \(E\) on the right, has exactly the same structure as \(\mathcal{B}_H\). As a result, the optimal semi-matching \(M_E\) over the sub-graph is the same as the optimal semi-matching on the bipartite graph \(\mathcal{B}_H\), which we denote by \(M_H\). Thus, the degrees of the honest workers in \(M_E\) are by hypothesis of the theorem \(d_1, d_2, \ldots, d_{|H|}\). Without loss of generality, suppose that honest workers are indexed such that \(d_1 \geq d_2 \geq \cdots \geq d_{|H|}\) and \(d_h\) denote the degree of honest worker \(h\).

**Part 1. Adversary strategy that affects at least** \(\frac{1}{2} \sum_{i=1}^{k-1} d_i\) **tasks**

The adversaries target honest workers \(\{1, 2, \ldots, k - 1\}\): for each \(i\), adversary \(a_i\) labels opposite to worker \(h_i\) (i.e. provides the incorrect response) on every task that \(h_i\) is mapped
to in the semi-matching \( M_E \). Furthermore, the adversary uses its last identity \( a_k \) to label opposite the true label for every task \( t \in T \) for which one of the first \( k - 1 \) adversaries have not already labeled on. We argue that under the penalty-based aggregation algorithm, this adversary strategy results in incorrect labels for at least \( \frac{1}{2} \sum_{i=1}^{k-1} d_i \) tasks. To see this, first note that the conflict set \( T_{cs} \) is the entire set of tasks \( T \). The bipartite graph \( B_{cs} \) decomposes into two disjoint bipartite graphs: bipartite graph \( B_E \) from \( H \) to \( E \) and semi-matching \( M_F \) from \( A \) to \( F \) that represents the adversary labeling strategy (it is a semi-matching because there is exactly one adversary labeling on each task). Since the bipartite graph \( B_{cs} \) decomposes into two disjoint bipartite graphs, computing the optimal semi-matching on \( B_{cs} \) is equivalent to separately computing optimal semi-matchings on \( B_E \) and \( M_F \). Since \( M_E \) is the optimal semi-matching on \( B_E \) and \( M_F \) is already a semi-matching by construction, the optimal semi-matching of \( B_{cs} \) is the disjoint union of \( M_E \) and \( M_F \). It is easy to see that in the resultant semi-matching, honest worker \( h_i \) and adversary \( a_i \) have the same degrees for \( i = 1, 2, \ldots, k - 1 \). Hence, for every task mapped to honest worker \( h_i \) for \( i = 1, 2, \ldots, k - 1 \) in the optimal semi-matching, the algorithm outputs a random label, and therefore outputs the correct label for atmost half of these tasks. Thus, the above adversary strategy results in incorrect labels for at least \( \frac{1}{2} \sum_{i=1}^{k-1} d_i \) tasks.

**Part 2. Upper Bound on number of affected tasks**

To simplify the exposition, we assume in the arguments below that the optimal semi-matching in the HARD PENALTY algorithm is computed on the entire task set and not just the conflict set \( T_{cs} \).

However, the bounds provided still hold as a result of lemma A.5.2 above. Also, we assume that the adversary labeling strategy is always a semi-matching, i.e. there is at most one adversary response for any task. If the adversary labeling strategy is not a semi-matching, they can replace it with an alternate strategy where they only label for tasks to which they will be mapped in the optimal semi-matching (the adversaries can compute this since they have knowledge of the honest workers’ responses). The optimal semi-matching doesn’t change (otherwise it contradicts the optimality of the original semi-matching) and hence neither does the number of affected tasks.

We first state the following important lemma:

**Lemma A.5.3** For any adversary labeling strategy, let \( B_{cs}(E) \) denote the bipartite graph \( B_{cs} \) restricted to all the workers \( W \) on the left and “true tasks” \( E \) on the right, and \( M \) be the optimal semi-matching on the bipartite graph \( B_{cs} \). Further, let \( M(E) \subseteq M \) be the optimal semi-matching restricted to task copies \( E \). Then, \( M(E) \) is an optimal semi-matching for the sub-graph \( B_{cs}(E) \).

**Proof** Suppose the statement is not true and let \( N(E) \) denote the optimal semi-matching on \( B_{cs}(E) \). We use \( d_w(K) \) to denote the degree of worker \( w \) in a semi-matching \( K \). Note that, \( d_a(N(E)) \leq d_a(M(E)) \leq d_a(M) \) for all adversaries \( a \in A \). For the adversaries who did not agree with any honest worker, they will have degrees 0 in the semi-matchings \( N(E) \) and \( M(E) \) but the inequality is still satisfied. Now, since \( N(E) \) is an optimal semi-matching
and $M(E)$ is not, we have that

$$\text{cost}(N(E)) < \text{cost}(M(E)) \Rightarrow \sum_{h \in H} d_h(N(E))^2 + \sum_{a \in A} d_a(N(E))^2 < \sum_{h \in H} d_h(M(E))^2 + \sum_{a \in A} d_a(M(E))^2$$

Now, consider the semi-matching $N$ on $B^{cs}$ where we start with the semi-matching $N(E)$ and then map the adversaries $A$ to the tasks in $F$ to which they were assigned in the original optimal semi-matching $M$. Now, we claim that $\text{cost}(N) < \text{cost}(M)$ which will be a contradiction since $M$ was assumed to be an optimal semi-matching on $B^{cs}$.

$$\begin{align*}
\text{cost}(M) - \text{cost}(N) &= \sum_{h \in H} d_h(M)^2 + \sum_{a \in A} d_a(M)^2 - (\sum_{h \in H} d_h(N)^2 + \sum_{a \in A} d_a(N)^2) \\
&= \sum_{h \in H} d_h(M(E))^2 + \sum_{a \in A} (d_a(M(E)) + \Delta_a)^2 \\
&- (\sum_{h \in H} d_h(N(E))^2 + \sum_{a \in A} (d_a(N(E)) + \Delta_a)^2) \\
(\text{where } \Delta_a \overset{\text{def}}{=} d_a(M) - d_a(M(E)) \geq 0) \\
&= (\sum_{h \in H} d_h(M(E))^2 + \sum_{a \in A} d_a(M(E))^2 - \sum_{h \in H} d_h(N(E))^2 - \sum_{a \in A} d_a(N(E))^2) \\
&+ 2 \sum_{a \in A} (d_a(M(E)) - d_a(N(E))) \cdot \Delta_a > 0 \\
(\text{since } d_a(M(E)) \geq d_a(N(E)) \text{ as stated above})
\end{align*}$$

Therefore, $M(E)$ is an optimal semi-matching for the sub-graph $B(E)$. \hfill \blacksquare

We are now ready to prove the result for part (a).

**PART 2A. Adversaries only provide incorrect responses or atmost 1 adversary provides correct responses**

We start with the case when adversaries only provide incorrect responses.

**Lemma A.5.4** Suppose that the adversaries never agree with the honest workers. Let $M$ be an arbitrary semi-matching on the bipartite graph $B^{cs}$ and suppose that this semi-matching is used in the penalty-based aggregation Algorithm to compute the true labels of the tasks. Further, let $b_1 \geq b_2 \geq \cdots \geq b_{|H|}$ denote the degrees of the honest workers in this semi-matching where $b_i$ is the degree of honest worker $h_i$. Then, the number of affected tasks is atmost $\sum_{i=1}^k b_i$.

**Proof** It follows from the assumption that adversaries never agree with the honest workers that there are no cross-edges between $A$ and $E$ in the bipartite graph $B^{cs}$. Thus, for any adversary labeling strategy, we can decompose $B^{cs}$ into disjoint bipartite graphs $B^{cs}(E)$ and $B^{cs}(F)$, where $B^{cs}(E)$ is the subgraph consisting of honest workers $H$ and task copies $E$.
and $\mathcal{B}^{cs}(F)$ is subgraph from the adversaries $A$ to the task copies $F$. This further means that the semi-matching $M$ is a disjoint union of semi-matchings on $\mathcal{B}(E)$ and $\mathcal{B}(F)$. Let the semi-matchings on the sub-graphs be termed as $M(E)$ and $M(F)$ respectively. Further, let $T \subseteq \mathcal{T}$ denote the set of tasks that are affected under this strategy of the adversaries and when the semi-matching $M$ is used to compute the reputations of the workers. We claim that $|T| \leq \sum_{i=1}^{k} b_i$. To see this, for each adversary $a \in A$, let $H(a) \subseteq H$ denote the set of honest workers who have “lost” to $a$ i.e., for each worker $h \in H(a)$ there exists some task $t \in \mathcal{T}$ such that $h$ is mapped to the true copy of $t$ in $M(E)$, $a$ is mapped to the false copy of $t$ in $M(F)$, and the degree of $h$ in $M(E)$ is greater than or equal to the degree of $a$ in $M(F)$. Of course, $H(a)$ may be empty. Let $\bar{A}$ denote the set of adversaries $\{a \in A : H(a) \neq \emptyset\}$ and let $\bar{H}$ denote the set of honest workers $\bigcup_{a \in \bar{A}} H(a)$. Now define a bipartite matching between $\bar{A}$ and $\bar{H}$ with an edge between $a \in \bar{A}$ and $h \in \bar{H}$ if and only if $h \in H(a)$. This bipartite graph can be decomposed into degenerate and growth sub-graphs by lemma A.5.1 above. In the growth subgraph, by Hall’s condition, we can find a perfect matching from adversaries to honest workers. Let $(A_1, H_1)$ with $A_1 \subseteq \bar{A}$ and $H_1 = \text{Img}(A_1)$ be the degenerate component. The number of tasks that adversaries in $A_1$ affect is bounded above by $\sum_{h \in \text{Img}(A_1)} b_i$. Similarly, for $A_2 = \bar{A} \setminus A_1$, we can match each adversary to a distinct honest worker whose degree is greater than or equal to the degree of the adversary. We can bound the number of affected tasks caused due to the adversaries in $A_2$ by the sum of their degrees, which in turn is bounded above by the sum of the degrees of honest workers that the adversaries are matched to. Let $H_2$ denote the set of honest workers matched to adversaries in the perfect matching. Thus, we have upper bounded the number of affected tasks by $\sum_{h \in H_1 \cup H_2} b_i$. It is easy to see that $|H_1 \cup H_2| \leq k$. Therefore, $\sum_{h \in H_1 \cup H_2} b_i \leq \sum_{i=1}^{k} b_i$. Therefore, the number of affected tasks $|T|$ is atmost $\sum_{i=1}^{k} b_i$ if the adversaries only disagree with the honest workers.

Since the above lemma is true for any choice of semi-matching $M$, it is true in particular for the optimal semi-matching on $\mathcal{B}^{cs}$. Therefore, it gives us an upper bound on the number of affected tasks when the adversaries only disagree with the honest workers.

Now, consider the case when there is exactly 1 adversary that agrees with the honest workers and all other adversaries only disagree. Let $M$ be the optimal semi-matching on the bipartite graph $\mathcal{B}^{cs}$ resulting from such an adversary strategy and let $a$ denote the adversary who agrees with the honest workers. Observe that we can apply the same argument in lemma A.5.4 above to get a upper bound on the number of affected tasks that the adversaries “win” against the honest workers. Let $T_1$ denote the set of these tasks. In the proof of the lemma above, there are two possible scenarios: either we obtain a perfect matching between the $k$ adversaries and some $k$ honest workers in which case we have accounted for all of the affected tasks. In the other scenario, when the degenerate component is non-empty, we have a total of at most $k - 1$ honest workers on the right and we bound $T_1$ by the sum of the degrees of these honest workers. Note, however that we may be missing out on some of the affected tasks, namely those that the adversary $a$ “loses” against other adversaries. The tasks that we might be missing out on correspond exactly to the task copies in $E$ that the adversary $a$ is mapped to in the optimal semi-matching $M$. Specifically, let $M(E)$ denote the semi-matching $M$ restricted to just the true task copies.
Identifying Unreliable and Adversarial Workers

E. Then it follows that we can bound the number of affected tasks by $|T_1| + d_a(M(E))$ where $d_a(M(E))$ denotes the degree of $a$ in $M(E)$.

Next observe that in both cases, we have bounded the number of affected tasks by the sum of the degrees of some $k$ workers in the semi-matching $M$ restricted to workers $H \cup \{a\}$ on the left and tasks $E$ on the right, i.e. in the semi-matching $M(E)$. Lemma A.5.3 tells us that $M(E)$ is in fact, the optimal semi-matching on the subgraph from workers $H \cup \{a\}$ to task copies $E$. Finally, lemma A.5.2 implies that this sum is at most $\sum_{i=1}^{k} d_i$ (by starting with $M_H$ as a feasible semi-matching) and the bound follows.

Next, we prove the result for part (b).

**Part 2b. Adversaries can provide arbitrary responses**

Finally, consider the general case when any number of adversaries can agree with the honest workers. We further suppose that $|H| \geq 2|A|$, otherwise the upper bound $U$ below becomes $\sum_{i=1}^{2k} d_i$, which is the set of all tasks $T$ and is a trivial upper bound.

First recall that lemma A.5.4 was applicable to any semi-matching and in fact, we can use the same argument even when the adversaries agree with the honest workers. Formally, consider an arbitrary strategy resulting in an optimal semi-matching $M$ on $B^{cs}$. Let $M(E)$ denote the semi-matching $M$ restricted to just the true task copies $E$. Suppose that the set of affected tasks $T$ under this adversary strategy is such that $T = T_H \cup T_A$ where $T_H$ are the tasks that the adversaries “win” against the honest workers and $T_A$ are the tasks that are affected when $2$ adversaries are compared against each other in the final step of the penalty-based aggregation algorithm. We can then utilize the argument in lemma A.5.4 to bound $|T_H|$ by the sum of the degrees of the top $k$ honest workers in the optimal semi-matching $M$. Further, we can bound $T_A$ by the sum of the degrees of the adversaries in the semi-matching $M(E)$.

Let $A_H \subseteq A$ denote the set of adversaries that have non-zero degrees in semi-matching $M(E)$, i.e. they are mapped to some task in $E$ in the optimal semi-matching $M$ on $B^{cs}$. The above sequence of claims implies that we can bound the number of affected tasks $|T|$ by the sum of the degrees of the top $s = k + |A_H|$ workers in the semi-matching $M(E)$. Now, we claim that this itself is upper bounded by the sum of the degrees of the top $j$ honest workers in the optimal semi-matching $M_H$ on the original honest worker-task assignment sub-graph $B_H$. To see this, start with $M_H$ as a feasible semi-matching from workers $H \cup A_H$ to task copies $E$. Lemma A.5.2 tells us that the sum of the degrees of the top $j$ workers in the optimal semi-matching on the subgraph $B^{cs}(E)$ is at most the sum of the degrees of the top $j$ honest workers in $M_H$. Further, lemma A.5.3 tells us that the optimal semi-matching on the sub-graph from workers $H \cup A_H$ to task copies $E$ is precisely the semi-matching $M(E)$. This shows that we can bound the number of affected tasks by $\sum_{i=1}^{j} d_i$. Finally, note that $|A_H| \leq k \Rightarrow j \leq 2k$ and hence, we can bound the total number of affected tasks by $\sum_{i=1}^{2k} d_i$.

**A.6 Uniqueness of degree-sequence in optimal semi-matchings**

In the arguments above, we have implicitly assumed some sort of uniqueness for the optimal semi-matching on any bipartite graph. Clearly its possible to have multiple optimal semi-matchings for a given bipartite graph. However, we prove below that the degree sequence
of the vertices is unique across all optimal semi-matchings and hence our bounds still hold without ambiguity.

**Lemma A.6.1** Let $M$ and $M'$ be two optimal semi-matchings on a bipartite graph $G = (V_1, V_2, E)$ with $|U| = n$ and let $d_1 \geq d_2 \cdots \geq d_n$ and $d'_1 \geq d'_2 \geq \cdots \geq d'_n$ be the degree sequence for the $V_1$-vertices in $M$ and $M'$ respectively. Then, $d_i = d'_i \forall 1 \leq i \leq n$, or in other words, any two optimal semi-matchings have the same degree sequence.

**Proof** Let $s$ be the smallest index such that $d_s \neq d'_s$, note that we must have $s < n$ since we have that $\sum_{j=1}^{n} d'_j = \sum_{j=1}^{n} d_j$. This means that we have $d_j = d'_j \forall j < s$. Without loss of generality, assume that $d'_s > d_s$. Now, $\exists q \in \mathbb{N}$ such that $d''_s > d''_s + \sum_{j=s+1}^{n} d''_j$ and since $d_j = d'_j \forall j < s$, we have that $\sum_{j=1}^{s} d''_j \geq \sum_{j=1}^{s} d'_j > \sum_{j=1}^{s} d''_j$. But, this is a contradiction since an optimal semi-matching minimizes the $L^p$ norm of the degree-vector for any $p \geq 1$ (Section 3.4 in [OPT]). Hence, we have that $d_i = d'_i \forall i$.

**A.7 Relation between Lower bound $L$ and Optimal semi-matching degrees**

We prove here the relationship between the lower bound $L$ in theorem 4 and the honest worker degrees $d_1, d_2, \ldots, d_{|H|}$ in the optimal semi-matching on the honest worker-task assignment graph $B_H$.

**Lemma A.7.1** Let $d_1 > d_2 > \cdots > d_{|H|}$ denote the degrees of the honest workers in the optimal semi-matching $M_H$ on $B_H$. Then the lower bound $L$ in theorem 4 is such that $L \geq \sum_{i=1}^{k-1} d_i$.

**Proof** Let $T_1, T_2, \ldots, T_{k-1}$ denote the set of tasks that are mapped to honest workers $h_1, h_2, \ldots, h_{k-1}$ in $M_H$ and $T := \bigcup_{j=1}^{k-1} T_j$. Now, we claim that for any $t \in T$, the only honest workers that provide responses for $t$ are amongst $h_1, h_2, \ldots, h_k$. In other words, $\text{PreIm}(T) \subseteq \{h_1, h_2, \ldots, h_k\}$. Suppose not, so that there exists $h_i \in \text{PreIm}(T)$ such that $i > k$. This would contradict the fact that $M_H$ is an optimal semi-matching. Specifically, Theorem 3.1 in [OPT] shows that a semi-matching $M$ is optimal if and only if there is no cost-reducing path relative to $M$. A cost-reducing path $P = (h^{(1)}, t^{(1)}, h^{(2)}, \ldots, h^{(d)})$ for a semi-matching $M$ on $B_H$ is an alternating sequence of honest workers and tasks such that $t^{(x)}$ is mapped to $h^{(x)}$ in the semi-matching $M$ for all $1 \leq x \leq d - 1$ and $\deg_M(h^{(1)}) > \deg_M(h^{(d)}) + 1$. Here $\deg_M$ denotes the degree of a node in the semi-matching $M$ (see section 2.1 in [OPT] for precise definition). Since $i > k$, we have that $d_s > d_i + 1$ for all $s \in \{1, 2, \ldots, k - 1\}$, which introduces a cost-reducing path. Therefore, we have that $\text{PreIm}(T) \subseteq \{h_1, h_2, \ldots, h_k\}$. Now

$$|T| = \left| \bigcup_{j=1}^{k-1} T_j \right| = \sum_{j=1}^{k-1} |T_j| = \sum_{j=1}^{k-1} d_j$$

where we have used the property of a semi-matching that a given task is mapped to only one worker. Using the definition of the lower bound $L$, it follows that $L \geq |T| = \sum_{i=1}^{k-1} d_i$.

\[ \]
Appendix B. Experimental Details

In this section we describe the details of our experimental setup discussed in section 5. We start with the benchmark algorithms.

EM algorithm. For the sake of completeness, we describe the setup in Raykar and Yu (2012), also known as the “two-coin” model. For each worker $w_i$, her accuracy is modeled separately for positive and negative tasks. For a task $t_j$ with true label $+1$, the sensitivity (true positive rate) for worker $w_i$ is defined as:

$$\alpha_i := \Pr[w_i(t_j) = +1 \mid y_j = +1]$$

Similarly, the specificity (1- false positive rate) is defined as:

$$\beta_i := \Pr[w_i(t_j) = -1 \mid y_j = -1]$$

Let $\Theta = \{(\alpha_i, \beta_i) \mid i \in [n]\}, \gamma$ denote the set of all parameters. For ease of exposition, we assume that the worker-task assignment graph is complete, i.e. all workers provide responses for all items, but the algorithm can be immediately extended to the case of incomplete graphs. Given the response matrix $L$, the log-likelihood of the parameters $\Theta$ can be written as:

$$\log Pr[L \mid \Theta] = \sum_{j=1}^{m} \log \left( \prod_{i=1}^{n} \alpha_i^{1[\mathcal{L}_{ij} = +1]} \cdot (1 - \alpha_i)^{1[\mathcal{L}_{ij} = -1]} \cdot \gamma + \prod_{i=1}^{n} (1 - \beta_i)^{1[\mathcal{L}_{ij} = +1]} \cdot \beta_i^{1[\mathcal{L}_{ij} = -1]} \cdot (1 - \gamma) \right)$$

The MLE of the parameters can be computed by introducing the latent true label of each task, denoted by the vector $y = [y_1, y_2, \ldots, y_m]$. The complete data log-likelihood can then be written as:

$$\log Pr[L, y \mid \Theta] = \sum_{j=1}^{m} \left( y_j \log(a_j \gamma) + (1 - y_j) \log(1 - \gamma)b_j \right)$$

where

$$a_j = \prod_{i=1}^{n} \alpha_i^{1[\mathcal{L}_{ij} = +1]} \cdot (1 - \alpha_i)^{1[\mathcal{L}_{ij} = -1]}$$

$$b_j = \prod_{i=1}^{n} (1 - \beta_i)^{1[\mathcal{L}_{ij} = +1]} \cdot \beta_i^{1[\mathcal{L}_{ij} = -1]}$$

Each iteration of the EM algorithm consists of two steps:

- **E-step:** Given the response matrix $L$ and the current estimate of the model parameters $\Theta^{(k)}$, the conditional expectation of the complete data log-likelihood is computed as

$$\mathbb{E} \left\{ \log Pr[L, y \mid \Theta^{(k)}] \right\} = \sum_{j=1}^{m} \left( \gamma_j^{(k)} \log(a_j^{(k)} \gamma^{(k)}) + (1 - \gamma_j^{(k)}) \log(1 - \gamma_j^{(k)})b_j^{(k)} \right)$$
where the expectation is w.r.t to $Pr[y | L; \Theta^{(k)}]$ and $\gamma_j^{(k)} = Pr[y_j = +1 | L; \Theta^{(k)}]$. Using Bayes theorem, we can compute

$$\gamma_j^{(k)} \propto Pr[L_1, L_2, \ldots, L_n | y_j = +1; \Theta^{(k)}] Pr[y_j = +1 | \Theta^{(k)}] = \frac{a_j^{(k)} \gamma_j^{(k)}}{a_j^{(k)} \gamma_j^{(k)} + b_j^{(k)} (1 - \gamma_j^{(k)})}$$

- **M-step:** Based on the current posterior estimates of the true labels $\gamma_j^{(k)}$ and the response matrix $L$, the model parameters are updated by maximizing $E\{\log Pr[L, y | \Theta^{(k)}]\}$, which can be shown to be a lower bound on the data log-likelihood (eq 7). The prevalence of positive tasks $\gamma$ is updated as:

$$\gamma^{(k+1)} = \frac{\sum_{j=1}^{m} \gamma_j^{(k)}}{m}$$

Similarly, the parameters $\alpha_i, \beta_i$ are updated as:

$$\alpha_i^{(k+1)} = \frac{\sum_{j=1}^{m} \mathbf{1}[L_{ij} = +1] \gamma_j^{(k)}}{\sum_{j=1}^{m} \gamma_j^{(k)}}$$

$$\beta_i^{(k+1)} = \frac{\sum_{j=1}^{m} \mathbf{1}[L_{ij} = -1] (1 - \gamma_j^{(k)})}{\sum_{j=1}^{m} (1 - \gamma_j^{(k)})}$$

These two steps are iterated until convergence of the log-likelihood $Pr[L | \Theta]$. To initialize the EM algorithm, we use the majority estimate $\gamma_j^{(0)} = \frac{\sum_{i=1}^{n} \mathbf{1}[L_{ij} = +1]}{n}$.

Note that the model we consider in the theoretical analysis is the simpler “one-coin” model where every worker is characterized by only a single parameter $p_i$ - the probability that she labels an assigned task correctly. The EM algorithm for that case can be derived in a similar manner to the one described above.

**KOS algorithms.** We implemented the iterative algorithm presented in Karger et al. (2014) which we replicate below in our notation.

**Algorithm 4 KOS algorithm**

1. **Input:** $\mathcal{L}, \mathcal{B} = (W, T, E), k_{\text{max}}$.
2. For all $(w_i, t_j) \in E$, initialize $m_{i \rightarrow j}^{(0)}$ with random $Z_{ij} \sim N(1, 1)$
3. For $k = 1, 2, \ldots, k_{\text{max}}$,
   - For all $(w_i, t_j) \in E$, update $m_{j \rightarrow i}^{(k)} = \sum_{i' \neq i} L_{ij} m_{i' \rightarrow j}^{(k-1)}$
   - For all $(w_i, t_j) \in E$, update $m_{i \rightarrow j}^{(k)} = \sum_{j' \neq j} L_{ij} m_{j' \rightarrow i}^{(k)}$
4. For all $t_j$, compute $m_j = \sum_{i=1}^{n} L_{ij} m_{i \rightarrow j}^{(k_{\text{max}}-1)}$
5. **Output:** label for task $t_j$ as $\hat{y}_j = \text{sign}(m_j)$

This algorithm was proposed for random regular graphs in the paper and we modified it in the following way for use in non-regular graphs:
Algorithm 5 \textsc{kos(norm)} algorithm

\begin{algorithm}
\begin{algorithmic}
\State \textbf{Input:} $\mathcal{L}, \mathcal{B} = (W, T, E), k_{\text{max}}.$
\State For all $(w_i, t_j) \in E$, initialize $m^{(0)}_{i \rightarrow j}$ with random $Z_{ij} \sim \mathcal{N}(1,1)$.
\State For $k = 1, 2, \ldots, k_{\text{max}},$
\hspace{1em}$\bullet$ For all $(w_i, t_j) \in E$, update $m^{(k)}_{j \rightarrow i} = \frac{1}{\text{deg}_B(t_j)} \sum_{i' \neq i} \mathcal{L}_{ij} m^{(k-1)}_{i' \rightarrow j}$.
\hspace{1em}$\bullet$ For all $(w_i, t_j) \in E$, update $m^{(k)}_{i \rightarrow j} = \frac{1}{\text{deg}_B(w_i)} \sum_{j' \neq j} \mathcal{L}_{ij} m^{(k)}_{j' \rightarrow i}$.
\State For all $t_j$, compute $m_j = \sum_{i=1}^{n} \mathcal{L}_{ij} m^{(k_{\text{max}}-1)}_{i \rightarrow j}$.
\State \textbf{Output:} label for task $t_j$ as $\hat{y}_j = \text{sign}(m_j)$.
\end{algorithmic}
\end{algorithm}

We chose $k_{\text{max}} = 100$ in our experiments.

\textbf{Simulation Details.} Here we discuss how we imposed the degree bias on adversaries in the Preferential Attachment scenario. Given a worker-task assignment graph, let $d_w$ denote the degree of worker $w$ and $d_{\text{min}}, d_{\text{avg}}, d_{\text{max}}$ denote resp. the minimum, average and maximum worker degrees.

- \textit{Adversaries have high degrees.} For each worker $w$, define $q_w = q \cdot \frac{d_{\text{max}} - d_w}{d_{\text{max}} - d_{\text{avg}}}$. Then, each worker $w$ is an adversary with probability $1 - q_w$. First, note that the expected number of honest workers is given by

$$\sum_w q_w = q \cdot \frac{d_{\text{max}} - d_{\text{avg}}}{d_{\text{max}} - d_{\text{avg}}} \sum_w (d_{\text{max}} - d_w) = q \cdot n$$

Next, we can see that workers with higher degrees have a smaller $q_w$, which implies that they have a greater chance of being an adversary. In an analogous manner, we deal with the case of low degrees.

- \textit{Adversaries have low degrees.} For each worker $w$, define $q_w = q \cdot \frac{d_w - d_{\text{min}}}{d_{\text{avg}} - d_{\text{min}}}$. Then, each worker $w$ is an adversary with probability $1 - q_w$. Again, we have that the expected number of honest workers is given by

$$\sum_w q_w = q \cdot \frac{d_{\text{avg}} - d_{\text{min}}}{d_{\text{avg}} - d_{\text{min}}} \sum_w (d_w - d_{\text{min}}) = q \cdot n$$

In this case, lower the degree $d_w$, higher the chance $(1 - q_w)$ that a worker is chosen as an adversary.

\textbf{References}
