International Journal of Data Science and Analytics (2019) 7:179–199
https://doi.org/10.1007/s41060-018-0139-5

REGULAR PAPER

Sloppiness mitigation in crowdsourcing: detecting and correcting bias for crowd scoring tasks

Lingyu Lyu[1] · Mehmed Kantardzic[1] · Tegjyot Singh Sethi[1]

Received: 25 July 2017 / Accepted: 8 June 2018 / Published online: 29 June 2018
© Springer International Publishing AG, part of Springer Nature 2018

Abstract

Due to different expertise levels, personal preference, or fatigue from long working of the crowd workers, the data obtained through crowdsourcing are usually unreliable. One big challenge is to obtain true information from such noisy data. Sloppiness, which represents the phenomena of observed labels which fluctuate around the true labels, is one type of the errors that has rarely been discussed in research. Moreover, most existing approaches try to derive truths in binary labeling tasks. In this paper, we deal with the sloppiness in a crowd scoring task, to obtain high-quality estimated labels. Crowd scoring task consists of ordinal and multiple labels, instead of just two labels. The worker in crowdsourcing can exhibit sloppiness, which can lead to unreliable scoring. We show that sloppy workers with biases, who constantly give higher (or lower) answers compared with true labels, can be effectively utilized to improve the quality of the estimated labels. To make use of the labels from crowd workers with biased sloppy behavior, we propose an iterative two-step model to infer the true labels. The first step identifies the biased workers and corrects the biases. The second step uses an optimization-based truth discovery framework to derive true labels from high-quality observed labels and the corrected labels from first step. We also present a hierarchical categorization for different types of crowd workers. Experiments on synthetic data as well as real-world datasets are conducted on the proposed model. The effectiveness of the proposed framework is demonstrated by comparing results with baseline models such as majority voting and expectation maximization-based aggregating algorithm; up to 16% improvement could be obtained for the accuracy.

Keywords Crowdsourcing · Sloppiness · Reliability · Bias · Scoring · Truth discovery

1 Introduction

The scale and diversity of data from various sources leads to information explosion and challenge of big data in recent years. These data are generated with wide variety, i.e., social networks such as Twitter, Facebook, or Linkedin; business or entertainment platforms, such as Amazon, IMDb, or Netflix; and many other Internet resources, such as the Massive Open Online Classes (MOOCs). Many of the data processing tasks require human intelligence, such as image labeling, video annotation, natural language processing, machine translation and product recommendation [18,28,31,38]. This stimulates the emerge of crowdsourcing systems, which is human-powered problem-solving methodology for collecting labeled data. Crowdsourcing has been demonstrated as an effective and important approach among others by Amazon Mechanical Turk, reCAPTCHA [30], Duolingo, and the ESP Game, etc. [29,33]. Different from traditional way of solving problems through experts, crowdsourcing mitigates the expensiveness of time and financial cost for large-scale tasks. However, despite the promises, key challenge within the crowdsourcing systems exists: quality of the collected data is unknown and is highly noisy in many cases. This is due to the fact of wide-ranging expertise levels of workers and difficulty levels of tasks. To obtain high-quality labels or answers for the crowdsourcing tasks, it is crucial to identify the trustworthiness of the workers.

The trustworthiness defines the reliability of a worker. In order to aggregate different answers provided by crowd

✉ Lingyu Lyu
l0lv0002@louisville.edu

Mehmed Kantardzic
mehmed.kantardzic@louisville.edu

Tegjyot Singh Sethi
t0seth01@louisville.edu

[1] Computer Engineering and Computer Science, University of Louisville, Louisville, KY, USA

workers, it is intrinsic to concentrate more on the reliable workers instead of untrustworthy ones. The factors which influence the reliability of a worker include expertise level, personal preference, understanding of the tasks, and worker's interests [33]. Due to the tedious and low reward nature of crowdsourcing tasks, errors are common even for workers who make an effort [18]. Ipeirotis and Gabrilovich [15] mentioned in their research that the monetary incentive, which is the case in many crowdsourcing platforms, is a mixed blessing: it might attract workers, but has the probability to make things worse [13,14]. Similar to truth discovery tasks [8,10,20,24,37], it is desired to discover true answers and worker reliabilities from multiple crowd answers. Table 1 presents the definitions of truth discovery tasks and crowdsourcing aggregation tasks, their similarities and differences [11].

A lot of research has been proposed to investigate the problems of inferring true labels and user reliabilities in crowdsourcing for binary labeling tasks [6,16,34,36]. The real-world applications, however, are not always with just yes or no answers. There are many sophisticated labeling tasks with more than two choices available. Here, we focus on ordinal crowd labeling tasks, which is also called crowd scoring tasks, with ordinal and multiple categories of labels. For example, grading project submissions from students in a class could be considered as a labeling task. Assume the scale of the labels is from 1 to 5. In this case, 4 is closer to 5 than 2. That means, giving label 4 to a submission, whose true label is 5, is different from providing label as 2. To obtain the true labels and worker reliability, a vast variety of techniques have been proposed on the basis of principle that reliable workers tend to provide true labels, and truth should be reported by many reliable workers [22]. Most of the existing approaches measures worker reliabilities according to their accuracy (e.g., inverse of variance [5]). The labels from more reliable workers contribute more to truth computation. However, these types of methodologies ignore one group of workers: highly biased and hence inaccurate workers. Biased workers are referred to those who constantly provide higher (or lower) label values compared with true labels. In addition, the labels obtained from the biased workers have the some patterns which could be used to extract useful information. As an example, assume the true labels for a set of items which need to be labeled are $\{3, 3, 3, 3, 3, 3\}$. We obtained labels for this set of items from two different workers. $w1$ as one of them provides corresponding labels as $\{3, 3, 3, 3, 3, 2\}$. $w2$ as the other one offers $\{2, 2, 2, 2, 2, 3\}$. Although $w1$ provides higher-quality labels in this case, $w2$ actually offers equal amount of information as $w1$. In this case, the pattern shown in labels from $w1$ is 1 scale higher than true labels in most of the observed data. By correcting the labels provided by $w2$ through adding 1, the accuracy of $w2$ equals to $w1$. As what Passonneau and Carpenter [23] proposed in their work, a highly biased and hence inaccurate annotator can provide as much information as a more accurate annotator.

This type of highly biased workers belongs to the crowd group which we call workers with sloppy behaviors. The term sloppy is based on the work of [32], which describes a worker who "views the question and data, but maybe insufficiently precise in their judgments" as sloppy worker. The similar definitions could also be found in [19]. Sloppiness is the phenomena of observed labels fluctuating around the true labels. The fluctuation could be caused by personal preference, misleading task description, or just fatigue after long time working, etc. Different from the spam labels, which are labels independent from truths and provide no useful information [23,25], the labels with sloppiness could still be utilized in the process of inferring the truth. A lot of research has been done to recognize or filter the spam/useless labels provided by crowd workers. As an example, the answers which are randomly generated by workers are one type of spam labels. In contrast, we found few discussions about the sloppiness within the crowd. Our goal in this research is to identify the workers with sloppy behaviors, especially the highly biased workers, and extract useful information from biased labels in order to get estimated labels as close as possible to the true labels. The true labels, which are also called as gold truths, are defined as the labels provided by experts in our research. It is, however, not reasonable to always have the gold truths available. For example, in a relevance judgment task, which requires to rate the relevance of (query, URL) pairs, there might be millions of pairs need to be rated. It would be too expensive and time-consuming to hire experts to sit down and do all the judgments in order to get the gold truths. With the absence of the gold truths, it is challenging to distinguish different behaviors of workers and thus difficult to infer truths from the observed labels. For example, when the true grade is known for a student submission in a class is A, on a scale of $\{A, B, C, D, F\}$. It is easy to justify a grader is "reliable" or not by just comparing the observed grade with the true grade. A reliable worker is defined as "Performs the tasks as requested. Reads the question and data and judges sufficiently precise" [32]. However, without knowing the true grade (in this case, assume true grade A is unknown), we could not simply claim whether a worker provides high-quality grades or not.

To deal with the problem of unavailability of true labels, as well as making use of the labels provided by crowd workers with biased sloppy behavior, we propose the iterative self-correcting truth discovery algorithm to infer the true labels and worker reliability from the crowd data. The approach (a) effectively identifies the biased workers: a bias score is calculated for each worker, in order to determine whether he/she belongs to biased worker group; (b) correcting the labels obtained from biased workers: according to the identified bias pattern, we de-bias the observed labels.

Table 1 Truth discovery and crowdsourcing aggregation: definitions, similarities, and differences [11]

	Truth discovery	Crowdsourcing aggregation
Definition	Integrates multi-source noisy information by estimating the reliability of each source	Aggregate noisy answers contributed by crowd workers to obtain the correct answers
Similarities	1. Both are trying to find trustworthy and accurate information from multiple sources 2. Their goals are to improve the quality of aggregation results 3. They have similar principles: reliable sources (workers) tend to provide high-quality information; sources are reliable if they provide accurate information 4. Techniques used are similar, and ground truth is usually unavailable in both cases	
Differences	Passive: data are already generated, and it is available when we find it	Active: user is able to choose what and how much data to generate
	Data crawled from online Web or collected from databases might have various types and may change dynamically	More information might be accessed on the source features, such as workers' location, accuracy on historical tasks, and education background

Bias pattern is recognized as the behavioral feature of the workers. For example, the positive (or negative) bias pattern discussed in our work indicates the feature of worker providing labels constantly higher (or constantly lower for negative pattern) than the truths; and (c) utilizing the truth discovery framework to iteratively update the truths and worker reliabilities. Here, the computed worker reliability is obtained after removing the bias, which reflects the actual information a worker could provide. The reliabilities are utilized to weight the labels provided by each worker to obtain estimated true labels. More reliable workers are assigned with higher weight. Experiments on synthetic and real-world datasets showed the effectiveness of the proposed methodology. At the same time, we also implemented some prevalent and state-of-the-art approaches for aggregating labels from crowdsourcing. The results of the proposed method are compared with outcome of these approaches.

There are several contributions of our work, and they are presented as follows:

(1) Different from many researches which focus on binary labeling problems, we highlight obtaining high-quality estimated true labels from the crowdsourced ordinal labeling tasks. Crowdsourcing researches usually targeted on simple problems such as choosing either 0 or 1 in a task. It is not always the case in real world. We concentrate on more complicated tasks with ordinal labeling. This type of task is much more like a scoring problem in which answers are chosen from a scale which consists of multiple ordinal labels.

(2) A hierarchical categorization of the crowd workers is introduced. The workers provide biased noisy labels are separated from reliable workers and the workers who provide useless labels.

(3) We propose an efficient method to recognize the biased sloppy workers. Due to the sparsity nature of crowdsourced data, we estimate the worker biases through calculating their expected error rate.

(4) We propose truth discovery-based approach to infer truths from both reliable workers and the corrected biased workers. Gold truth is not required to compute the estimated labels and worker reliabilities in our methodology.

The paper is organized as follows: Sect. 2 gives the discussion of related work. Section 3 describes the proposed methodology for recognizing highly biased workers and inferring true labels. Section 4 shows the experimental results for proposed approach and other prevalent methods for aggregating crowd labels. In Sect. 5 we draw the conclusions and present the future work.

2 Related work

Relevant work on aggregating crowd labels has been reported by other research [4,7,9,26,35,40]. The simplest aggregating method is majority voting (MV). MV assumes high-quality workers are majority among the crowd and they work independently from each other. It assigns the same weight to all the workers and then updates the truths. ZenCrowd [7] was proposed by Demartini, Difallah, and Cudr-Mauroux. It was an extension of MV, which weight the workers' answers by their corresponding reliability. The approach uses expectation maximization (EM) to simultaneously estimate the true labels and worker reliability. Dawid and Skene's [7] approach models a confusion matrix for each worker, as well as the class prior. They proposed to use EM to estimate true labels, confusion matrix, and the prior in their work. Snow et al. [28] utilize a similar model for human linguistic annotation. They consider the fully supervised case of machine learning estimation. Whitehill et al. [35] proposed GLAD, which stands for Generative model of Labels, Abilities, and Difficulties, to simultaneously infer the expertise of each worker, the difficulty of each item, and the most probable label of each item. Raykar et al. [26] use a Bayesian approach to add worker-specific priors for each class. Their algorithm evaluates the different experts and gives an estimate of the actual hidden labels by using an automatic classifier. Raykar's approach requires the feature representation of the items, however, which is not always available. If such feature representation does not exist, the method falls back to maximum-a-posteriori (MAP) estimation. Zhou et al. [40] utilize a minimax entropy principle to estimate the true labels from the crowd answers. Their method assumes that labels are generated by a probability distribution over worker, items, and labels. By minimizing the Kullback-Leibler (KL) divergence between the probability and unknown truth, they infer the item confusability and worker expertise. Zhou's method is a natural extension to Dawid and Skene's work [4], and the essential difference is that the minimax entropy takes into account item confusability, in addition to worker expertise. Ertekin et al. [9] present an algorithm called "CrowdSense" that works in an online fashion to dynamically sample subsets of workers based on exploration/exploitation criterion. The algorithm produces a weighted combination of the subset of workers' votes to approximate the crowd's opinion.

All the work mentioned above belong to either iterative methods or probabilistic graphical model (PGM)-based methods to infer the true labels. As an example, MV and "CrowdSense" are iterative methods. MV approach is a special case of iterative weighted voting algorithm, where weights of workers are identical, and thus only one iteration is required. The remaining presented algorithms are PGM-based methods. Li et al. [21] provided a survey on these methods in their work. Here is a summary of the iterative aggregating algorithms: in iterative methods, truth computation step and weight estimation step are iteratively conducted until convergence. In truth computation step, truths are inferred, while worker weights are assumed to be fixed. In weight estimation step, the workers' weights are updated based on the current estimated truths. PGM models, however, incorporate the principle of truth discovery: if a worker provides trustworthy labels frequently, he will be assigned a high reliability; if a label is provided by a reliable worker, it will have higher probability to be chosen as truth. The corresponding likelihood of a PGM model is presented in formula (1).

$$\prod_{s\in S} p(\omega_s|\beta) \prod_{o\in\varnothing} \left(p(v_o^*|\alpha) \prod_{s\in S} p(v_o^s|v_o^*, \omega_s)\right) \tag{1}$$

where v_o^s is the labels provided by the worker s for item o, v_o^* is the true label for the item, and ω_s is the worker's weight. α and β are the hyperparameters correlated to the truth and worker reliability. The graphical representation of the general model is shown in Fig. 1. To infer the hidden true labels and worker weights, techniques such as expectation maximization (EM) and Gibbs sampling could be adopted.

In addition to the inference and PGM models, another type of algorithm that could be utilized to generate truths from crowd answers—optimization-based truth discovery methods [1,20,22]. This type of approach captures the true labels by solving the optimization problem, which is in the following formulation:

$$\arg\min_{\{\omega_s\},\{v_o^*\}} \sum_{s\in S} \omega_s \sum_{o\in\varnothing} d(v_o^s, v_o^*) \tag{2}$$

where $d(\cdot)$ is the loss/distance function between the crowd answer and identified truth. By minimizing Eq. (2), the aggregated results (v_o^*) will be closer to answers from workers with higher weights. Meng et al. [22] proposed an effective optimization-based truth discovery framework to infer the truths for crowd sensing of correlated entities. Aydin et al. [1] investigate a novel weighted aggregation method to improve accuracy of crowdsourced answers for multiple-choice questions. They deploy the optimization-based truth discovery algorithm, as well as the lightweight machining learning (ML) techniques for building more accurate crowdsourced question answering systems. Li et al. [20] propose to identify the true information among multiple sources of data by using an optimization framework. Their model treats the truths and source reliability as unknown variables. The objective is to minimize the overall weighted deviation between truths and observations. They also discussed different types of loss functions which could be incorporated into the framework.

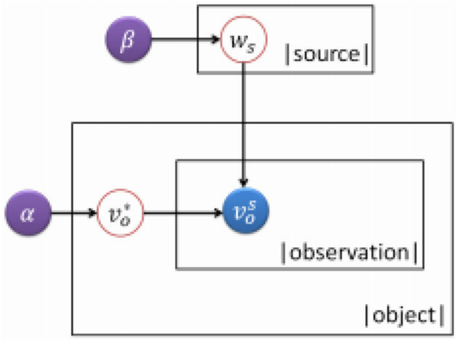

Fig. 1 The general probabilistic graphical (PGM) model [21]

As stated in [21], there are differences between the three types of aggregation or truth discovery methodologies; however, we do not claim that one of them is better than another. It is, however, possible to see the advantages of different approaches in various cases. Li et al. [20] proved superiority of the optimization-based methods in their work by comparing the results of the proposed framework with some Bayesian analysis-based approaches, on heterogeneous data. In terms of interpretability, iterative methods are easier to understand than others. PGM models and optimization-based approaches could take into account the prior knowledge about true labels and workers compared to iterative algorithms. We utilized an optimization-based framework in our research since it is easier for us to correct the highly biased crowd workers in coordinate descent process. In addition, it will be easier to extend to multiple data type cases in the future, such as crowd answers contain both numerical and categorical ordinal data.

Much of the literature has accounted for the labeler bias [6,16,26,28,33,34,36]. As Wauthier and Jordan [33] stated, the data collected from crowdsourcing services is often very noisy: unhelpful labelers may provide wrong or biased answers which may greatly degrade the learning algorithms. Bias may be caused by personal preference, systematic misunderstanding of the labeling task, lack of interest, or malicious intents. When the levels of bias are low, some of the consensus or aggregating algorithms could still work well, but could become unreliable when the quality of workers varies greatly [27]. For example, when the existence of extreme biased workers in a binary labeling task, the EM model proposed by Dawid and Skene [4] is able to flip the labels to achieve higher accuracy. However, if too many of the workers are highly biased, the model cannot separate the noise from the true labels [23].

Snow et al. [28] proposed a multinomial model similar to Naive Bayes for modeling labels and workers. They estimate the worker quality in a Bayesian setting by comparing with the gold standard and apply the weighted voting rule which give highly biased workers negative votes. In this way, they correct the bias in categorical data. Wauthier and Jordan [33] presented Bayesian bias mitigation for crowdsourcing (BBMC), a Bayesian model to capture the sources of bias by describing workers as influenced by shared random effects. These effects are also known as the latent features which depict the preferences of the workers. Raykar et al. [26] utilized a Bayesian approach to capture the mislabeling probabilities by assigning latent variables to workers. Ipeirotis et al. [16] proposed to separate error and bias for the workers. They pointed out that a biased worker is still more useful than a worker who reports labels at random. Dekel and Shamir [6] presented a two-step process to pruning the low-quality worker in a crowd. They first remove the workers by how far they disagree with the estimated true label, and then they reuse the cleaned dataset to build the model. Yan et al. [36] employed a coin flip observation model to learn the worker bias and then optimally selecting new training points and workers. Welinder et al. [34] modeled the worker in an image labeling process as a multidimensional entity with variables representing competence, expertise, and bias. Their work generalizes and extends the research of [35] by introducing worker bias. The authors in [17] introduced and evaluated probabilistic models that can detect and correct task-dependent bias automatically. Zhang et al. [39] proposed an adaptive weighted majority voting (AWMV) algorithm to handle the issue of biased labeling. Their work is based on the statistical difference between the labeling qualities of the two classes.

Most of the research mentioned above assume that the task is binary labeling. Although many of them claim it could be generalized to more than two labels, it is difficult and have high computational complexity. Some of the work tackles the bias workers; however, they did not really separate the biased ones from the spam workers, those who randomly select labels for items. The authors in [23] give discussions about the features of the biased annotators. They, however, solely utilized expectation maximization (EM)-based probabilistic model for a word sense annotation task as a case study. They do not handle the biased labeling in [23]. The model proposed by Snow et al. [28] requires the gold truths to be compared to observed labels in order to recognize biased workers. In [16], the methodology would work only if each worker provides at least a specified number (20 to 30) of labels, which is difficult to fulfill in many cases. The BBMC model proposed by Wauthier and Jordan [33] captures the sources of labeler bias through shared random effects. Different from their work, we take into account of the overall effect of the bias to obtain high-quality estimated labels. In the work of [17], task-specific bias, such as confusing a specific class with another, is captured by utilizing the task features. Our work, however, accounts for the biases of workers through the observed labels. In other words, we do not rely on the features of the labeling task to account for biases. In [39], the authors adapted majority voting algorithm to consider the bias of worker through bias rate in a binary labeling task. This paper, unlike [39], proposed an approach to model biases on an optimization-based truth discovery framework for ordinal labeling tasks. The authors in [34] modeled the label assigned by a worker according to a linear classifier. The classifier is parameterized by a direction and a bias parameter, and the model is developed under the assumption that the labels are binary. In contrast, our work deals with the worker separation problems in a labeling task with multivalued ordinal labels. Instead of the workers without information offered in the labels, we focus on correcting the biased workers to improve the quality of the estimated truths.

3 Methodology

In this section, we explain the framework of the iterative self-correcting truth discovery model, which recognizes the highly biased workers, and computes the truth and weights from the bias-corrected workers. The model consists of two steps: (1) bias correcting: compute a bias score for each crowd worker, and according to the score, determine whether the worker belongs to the highly biased group. If he/she is highly biased, de-bias the worker. (2) Truth discovery: formulate the truth computation problem as an optimization problem which models the truths as weighted voting of the biased corrected labels from step 1. We also give discussion that the model could be easily generalized to numerical labeling tasks.

3.1 Problem formulation

We give detailed description of the problem which we will solve in our work, and then the proposed framework will be presented. Before giving definition of the problem, we introduce the different types of workers in a labeling task to give better understanding of the worker behaviors.

3.1.1 Background

There are two general types of workers: reliable workers and unreliable workers. Reliable workers are ones who would be able to provide high-quality labels. The definition of this type of worker is similar to the concept of "proper worker" from [29]: the worker which "performs the tasks as requested. Reads the question and data and judges sufficiently precise." Unreliable workers, however, give labels which would have an uncontrolled effect, or even negative influences on obtaining truths, by utilizing learning/aggregating algorithms. We use the term "noisy worker" in our work to represent unreliable workers, due to the fact that the labels provided by this type of workers are highly noisy. Based on the different behavior patterns of the noisy workers, many researches have been proposed to categorize these workers. For example, Vurrens et al. [32] categorized noisy workers into random spammer, uniform spammer, and sloppy worker; Kazai et al. [19] defined a topology of noisy workers as sloppy workers, incompetent workers, and spammers; and Passonneau and Carpenter [23] proposed spam annotators, biased annotators, and adversarial annotators. Although different literature gives different names or definitions, the categories of noisy workers could be generalized into two different groups:

(i) *Spam workers* This type of workers provides useless labels, which means not so much information could be extracted and utilized for the aggregating process. Instead, they would greatly degrade the estimated true labels. For example, random spammers and uniform spammers in [32], spammers in [19], and spam annotators in [23] all belong to this group of worker.

(ii) *Useful low-quality workers (sloppy workers)* This type of workers is very special. They provide low-quality labels, but after processing the labels, we could obtain high-quality results. For example, in a binary labeling task, flipping a worker's provided labels, whose accuracy is 0.3, could result 0.7 accuracy on the flipped labels. The adversarial annotators, which is also extreme biased workers, mentioned in [23] falls into this group of worker. The sloppy workers and incompetent workers presented in [19] also belong to this category, according to their definitions. For the easier reference, we call this type of workers as sloppy workers. Our work concentrates to deal with sloppy workers in this research.

We give visual presentation between reliable workers and noisy workers in Fig. 2. The x-axis denotes the deviation between the observed labels from workers and the true labels, and the y-axis is the probability of the deviation. In Fig. 2, the reliable worker showed higher accuracy compared with noisy workers. A perfect worker would have probability as 1 at the point of $deviation = 0$. Four different examples of noisy workers were presented. Among them, "noisy worker_1," "noisy worker_2," and "noisy worker_3" are the spam workers, in which they do not provide useful information in the labels. For example, the "noisy worker_1" has low accuracy, and errs on both sides of deviations. The distribution of the deviations spreads over the x-axis. While "noisy worker_2" mainly has random errors toward the right side (positive) of the deviations. The "noisy worker_3," however, tends to give random labels which lead to negative deviations. The "noisy worker_4" in the figure belongs to the sloppy worker group. Similar to spam workers, sloppy worker has low accuracy. Unlike spam workers, sloppy worker does not give random labels. In Fig. 2, the sloppy worker provides labels close enough to truths.

It is much easier to distinguish between the spam workers and sloppy workers in a binary labeling task. This is due

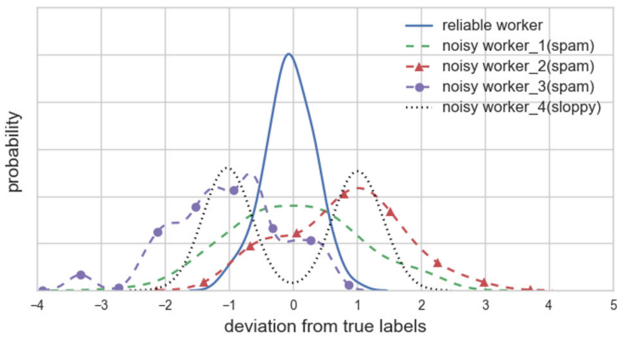

Fig. 2 Illustration of reliable workers and noisy workers

to the nature that the accuracy decision threshold of spam worker is fixed, which is 0.5, for binary labeling tasks [27]. As an example, if worker's accuracy is higher than 0.5, the worker could be classified as a reliable or good worker. If a worker's accuracy is less than 0.5, by flipping the labels he/she provided, we could still get acceptable results. Only when the probabilities of choosing one of the two labels are equal, the worker is considered as a spammer. In real-world applications, the scale of the labels actually entails more than two levels. As an example, a student's test grade is not always only fail or pass; the scale is usually like "A, B, C, D, F ." In this case, the threshold 0.5 is no longer applicable due to the fact that we cannot just simply flip the labels. To make use of the labels from sloppy workers, we need to come up with a different strategy.

Since the accuracy measure could not be used to separate sloppy workers from spam workers, we investigate one important attribute discovered by Dawid and Skene [4]. According to the research in [4], mostly, even though reliable workers make errors, their error will be on the diagonal of the true labels. As an example, Table 2(a) presents the confusion matrix of a reliable worker. Instead of giving the results as the number of instances, the matrix shows the ratio of different observed labels regarding true labels. The italic emphasis gives accuracies based on truths, such as 0.7 in the cell with True Labels as "A" and Observed Labels as "A" is calculated as:

$$\begin{aligned}\mathrm{Acc}_{\mathrm{A}} &= \Pr(\mathrm{obsLabels} = \mathrm{A} | \mathrm{tLabels} = \mathrm{A}) \\ &= \frac{|\mathrm{es}_{\mathrm{A}}|}{|\ \mathrm{ts}_{\mathrm{A}}|}\end{aligned} \tag{3}$$

where "obsLabels" is the observed answers from workers, and "tLabels" represents the true labels. The ts_{A} is a set with all the tasks have true labels as "A," and $\mathrm{es}_{\mathrm{A}} \subseteq \mathrm{ts}_{\mathrm{A}}$, which contains the tasks with observed labels as "A." It could also be interpreted as: if the task has true label as "A," the worker has the possibility of 0.7 to provide the correct label ("A") to this task. Similarly, the cell (A, B) gives the possibility of the worker provide grade "B" for a task with true grade as "A." As indicated in Table 2(a), the worker only errs one scale up or down from the true labels. As an example, when the true label is "B," although the worker might make error, he/she only shows possibility > 0 of labeling the task as "A" or "C." The possibility of observed grade as "D" or "F" is 0.

Table 2 Examples of confusion matrix of workers

Observed labels	A	B	C	D	F
(a) *Diagonal of correct labels for reliable workers* [4]					
True labels					
A	*0.70*	0.30	0.00	0.00	0.00
B	0.30	*0.60*	0.10	0.00	0.00
C	0.00	0.00	*0.80*	0.20	0.00
D	0.00	0.00	0.40	*0.50*	0.10
F	0.00	0.00	0.00	0.30	*0.70*
(b) *Diagonal of correct labels for sloppy workers*					
True labels					
A	*0.40*	0.60	0.00	0.00	0.00
B	0.10	*0.20*	0.70	0.00	0.00
C	0.00	0.10	*0.30*	0.60	0.00
D	0.00	0.00	0.00	*0.20*	0.80
F	0.00	0.00	0.00	0.50	*0.50*

However, let us investigate another example of worker w's confusion matrix as shown in Table 2(b). Comparing the confusion matrix of reliable worker in Table 2(a), it could be seen that w in Table 2(b) provides lower-quality labels. However, the same as shown in 2(a), w also presented the diagonal of correct labels attribute. After combining the characteristics of different types of workers and the discovery, we give a detailed definition for sloppy worker in our work: sloppy workers have lower accuracy than reliable workers; at the same time, they are like reliable workers: they only err on the diagonal of the true labels. The "noisy worker_4" in Fig. 2 showed the interesting fact that the deviations are chosen from $\{0, -1, +1\}$. "Err on the diagonal" in this example indicates the worker makes error either as $+1$ or as -1. For different real-world applications, the diagonal attribute might be as: errors/deviations can be selected from $[-a, +a]$ instead of just $+1$ or -1. For example, while grading a student's homework from class, a reliable worker might make errors between $[-5, +5]$ on a grading scale from 0 to 100. We call the $[-a, +a]$ as the fault tolerance range. That means, if a worker only errs on this range, he/she did have the "diagonal attribute." The following example gives an intrinsic motivation of defining the range for sloppy workers: assume a multivalued ordinal labeling task, and the possible label set is $\{1, 2, 3, 4, 5\}$. There are 10 items need to be labeled, with truths as [2, 3, 3, 4, 3, 2, 3, 5, 4, 3]. Suppose two workers w_1 and w_2 worked on the task. The labels provided by w_1 as [3, 4, 4, 5, 4, 3, 3, 5, 5, 4], and labels from w_2 are [5, 5, 5, 5, 5, 5, 5, 5, 5, 5]. Both of the workers in the example have low accuracies $w_1 = w_2 = 0.2$. In addition, both of them tend to have positive deviations for most of their labels compared to truths. The amount of information, however, provided by w_1 and w_2 is different. By correcting the labels of w_1 through subtracting 1, accuracy as 0.9 could be obtained, while w_2' s labels offer no information regarding approximating the truths. From the example, we could see that by defining the fault tolerance range, it would be possible to utilize the labels from w_1, while removing w_2 as spam worker.

To determine the value for a, which is used in the fault tolerance range, here are some guidelines:

(1) As discussed above, we constrain the deviation range of sloppy worker under acceptable range, which is similar to reliable workers. One way to choose a would be applying binary division heuristic approach to decide the value. Here is how binary division heuristic approach could be conducted. Assume the possible ordinal label set as L.

   (a) First set $a_0 = |L|/2$, where $|L|$ is the total number of possible ordinal labels. Calculate the estimated truths and selected performance metrics such as accuracy or F measure, by utilizing the setting of a.
   (b) Then set $a_1 = a_0/2$, calculate the estimated truths and performance metrics using a_1.
   (c) Set a_0 equal to a_1. Keep repeating (b) step until there is no significant difference between the calculated metrics.

(2) For multivalued ordinal labeling tasks, usually the possible labels would be less than 10 in real-world applications, and the typical value chosen for a would be 1 as illustrated in our work.
(3) Another way to decide a in some cases is to request an expert to define the value regarding the task. For example, while labeling side effect level of a new type of medicine, a doctor who is highly skilled in similar medical domain should be defining the value of a.

For the purpose of easier explanation and experimentation, we set a just as 1 in the context of the paper.

The curve shown in Fig. 2 indicates a sloppy worker with the error (deviation from true labels) perfectly evenly distributed between -1 and 1. That means, the probability of worker gives error as -1 is exactly the same as the probability error as 1. We call this type of worker as perfectly erred sloppy worker. As an example, a perfectly erred sloppy worker j might have probability of 0.6 giving label the same as the true label, 0.2 for having deviation as $+1$, and 0.2 for giving deviation as -1. It is not necessary that every sloppy worker be perfectly erred. In most cases, the probability of sloppy worker showing error as -1 is not equal to probability of error as 1. For example, the possibility of deviation as -1 could be greater than probability of deviation of $+1$. Figure 3 presents examples for this type of sloppy workers. The curve as "sloppy worker_1" gives higher probability for deviation as 1 than -1, and "sloppy worker_-1" shows the worker tends to have deviation as -1 more than as 1. We could further divide the unevenly erred sloppy workers as follows:

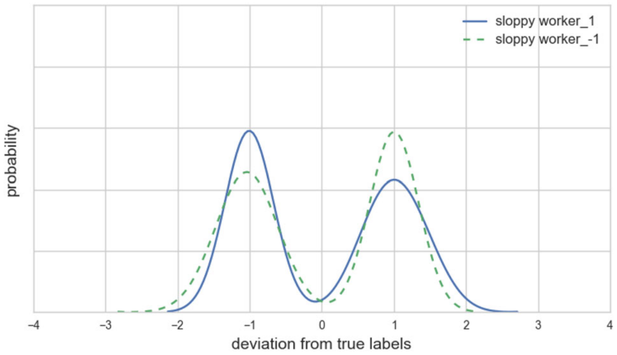

Fig. 3 Examples of biased sloppy workers

(1) *Randomly erred sloppy worker* This type of sloppy workers might have probability of giving deviation on one side slightly more than the other. For example, a worker j has the probability of 0.4 for providing label the same as the true label, 0.4 for having deviation as $+1$, and 0.2 for deviation as -1. Randomly erred sloppy workers are quite similar to perfectly erred sloppy workers as to the amount of information they could provide. Thus in our experimentation section, we only give the results of the influence of perfectly erred sloppy workers on different models, such as majority voting, GLAD algorithm, optimization-based truth discovery method, etc.
In our work, The randomly erred and perfectly erred sloppy workers are not removed while applying aggregating algorithms. One reason is that both of their errors are within the fault tolerance range $[-a, +a]$; thus, they do not affect the estimated truths much. In addition, their weights in aggregating algorithms would be much lower compared to weights of reliable workers and corrected biased sloppy workers. Overall, the labels of randomly and perfectly erred sloppy workers would have small influence on approximating the truths.
(2) *Biased sloppy worker* Different from randomly erred sloppy workers, we are able to extract useful information from the biased sloppy workers. This type of sloppy workers is the same as the highly biased annotator mentioned in [23]. Our proposed algorithm focuses on identifying the biased sloppy workers. The reasons we do not deal with the perfectly erred sloppy workers are: (1) it is difficult to extract information from their provided labels. (2) It could be possible to cancel some errors between this type of workers in some extent, by just utilizing average aggregation algorithm (proved in theoretical in Sect. 3.1.2). The reasons also apply to randomly erred sloppy workers. The confusion matrix listed in Table 2(b) is an example of biased sloppy worker. We can further divide the workers into two subcategories: positive biased sloppy workers, who tend to

give much higher probability for deviation as + 1 comparing to − 1. Negative biased sloppy workers. They biased more toward − 1 rather than + 1.

Table 3 gives the summarization of the definitions of different types of workers in this work. The statistic traits column shows the characteristics we used in this work for categorizing the crowd. To separate the randomly erred sloppy workers from the biased sloppy workers, a threshold value could be defined. Similar method while fixing the value of a for fault tolerance range could be utilized here: we first set the threshold value as 0.5 to recognize the randomly erred workers from biased sloppy workers. Then binary search a value within range $[0.5, 0.9]$ as the threshold till convergence. Another way could be easily implemented to approximate the threshold. We call it incremental heuristic way: test the values in $\{0.5, 0.6, 0.7, 0.8, 0.9\}$ on the collected label set, separately. The one which gives the best performance is chosen as the threshold. The incremental method, which is efficient in some cases, is computationally less complex for obtaining the threshold value compared to the former one.

3.1.2 Formalization of sloppiness

We define the mathematical notations for the problem of making use of sloppy workers' labels. Suppose there are N items that need to be labeled and M workers for the task. Let $L = \{1, 2, \ldots, C\}$ is the set of labels in which workers could choose from, $y_i^j \in L$ is the label assigned to ith item by worker j, and y_i is the true label of item i. $\hat{y_i}$ is the estimated true label for item i, which is called the consensus label. Assume $\varepsilon (= y_i^j - y_i)$ is the deviation/error between the observed label (obtained from workers) and true label. Table 4 summarizes the important notations used in the paper. Define the following probabilities:

$$P_\varepsilon^j = \Pr(y_i^j = c + \varepsilon | y_i = c) \tag{4}$$

According to the definition of a perfectly erred sloppy worker, we know that $\varepsilon \in \{0, 1, -1\}$, and $P_1^j = P_{-1}^j$, where

$$P_1^j = \Pr(y_i^j = c + 1 | y_i = c) \tag{5}$$

$$P_{-1}^j = \Pr(y_i^j = c - 1 | y_i = c) \tag{6}$$

$$P_0^j = \Pr(y_i^j = c | y_i = c) \tag{7}$$

$$\sum_{\varepsilon \in \{-1,0,1\}} P_\varepsilon^j = 1 \quad \text{and} \quad 0 \le P_{-1}^j, P_0^j, P_1^j < 0.5 \tag{8}$$

We first give proof that under some conditions, the existence of perfectly erred sloppy workers has no significant influence on the consensus labels. Let the calculated expectation of consensus label for item i be $E(\hat{y_{ic}})$. Here we also give

Table 3 Definitions of different types of workers

Worker types				Definition	Statistic traits
Reliable worker				Worker who provide high-quality labels	Accuracy > 0.5
Noisy worker (provide low-quality labels)	Spam worker			Worker who provide useless labels	Accuracy ≤ 0.5, and having deviations outside the fault tolerance
	Sloppy worker (make errors which are mostly equal to + 1 or − 1)	Perfectly erred sloppy		The errors evenly distributed between + 1 and − 1	Accuracy ≤ 0.5, and Pr(error=1) equals to Pr(error= − 1)
		Randomly erred sloppy		The prob. of errors between observed labels and true labels on one side (e.g., + 1) is slightly different from the other (e.g., − 1)	Accuracy ≤ 0.5 and 0≤Pr(error=1)≤0.5 and 0≤Pr(error= −1)≤0.5
		Biased sloppy	Positive biased	The distribution of deviation between observed labels and true labels as +1 >as − 1	Accuracy<0.5 and Pr(error=1)>0.5
			Negative biased	The distribution of deviation between observed labels and true labels as +1 <as − 1	Accuracy<0.5 and Pr(error= −1)>0.5

Table 4 Frequently used notations

Symbol	Definition
M	Number of workers
N	Number of items
L	Set of labels worker could choose from
$Y^{(*)}$	True labels set of all items
$Y^{(j)}$	Set of labels obtained from worker j
$Y^{'(j)}$	Set of labels after bias correction from worker j
W	The set contains all worker weights
Θ	The set contains all worker accuracies
$E(\hat{y}_{ic})$	Expected consensus label for item i, whose true label is c
y_i	True label of item i
y_i^j	Label assigned to item i by worker j
$\hat{y}_i$	Consensus label for item i
ε_i^j	Deviation between label from worker j and truth for item i
ω_j	Weight of worker j
y_{ic}^j	Label assigned by worker j to item i, whose true label is c
P_ε^j	Probability of worker j has deviation from true label as ε
θ_j	Accuracy of worker j
α_j	Conditional probability of worker j has error as $+1$
β_j	Conditional probability of worker j has error as -1
η_a	Marginal error threshold on worker is highly biased or not
η_b	Bias score threshold for biased workers

assumption that the estimated labels are obtained by averaging the observed labels from workers, and every worker is independent from each other. Also, suppose the true label for item i as c, which means $y_i = c$. Thus,

$$E(\hat{y}_{ic}) = E\left[\sum_{j=1}^{M} y_{ic}^j / M\right] = \frac{1}{M} E\left[y_{ic}^1 + y_{ic}^2 + \cdots + y_{ic}^M\right]$$
$$= \frac{1}{M}\left(E\left[y_{ic}^1\right] + E[y_{ic}^2] + \cdots + E[y_{ic}^M]\right) \tag{9}$$

where y_{ic}^j is the label given to item i whose true label is c, and $\hat{y}_{ic}$ denotes the estimated label for item i, for which true label is c. The $E[y_{ic}^j]$ is calculated as:

$$E(y_{ic}^j) = \sum_{k \in \{c-1, c, c+1\}} \left[y_{ic}^j \times \Pr(y_{ic}^j = k)\right]$$
$$= (c-1) \times P_{-1}^j + c \times P_0^j + (c+1) \times P_1^j \tag{10}$$
$$= c - (P_{-1}^j - P_1^j) \quad or \quad c + (P_1^j - P_{-1}^j)$$

Formula (10) is obtained based on the condition $\sum_{\varepsilon \in \{-1,0,1\}} P_\varepsilon^j = 1$ in 8. The hypothesis here is that worker j is a perfectly erred sloppy worker, which indicates that $P_1^j = P_{-1}^j$. The result could be obtained as $E(y_{ic}^j = c)$. We then could get $E(\hat{y}_i)$ in (9):

$$E(\hat{y}_{ic}) = \frac{\sum_{j=1}^{M} c}{M} = c \tag{11}$$

Till now, we proved that existence of perfectly erred sloppy workers has no significant influence on the expected consensus labels, under the conditions that workers are independent from each other and using the average aggregating algorithm. However, there are two problems in real-world applications:

(1) Mostly, it is too ideal to have perfectly erred sloppy workers: a worker should perfectly avoid the true grades and evenly distribute labels only one scale up and down around the true grades. In real-world scenarios, the sloppy workers would probably have a higher possibility toward either one of the errors (ε).

(2) It is unrealistic to obtain the exact expected consensus label as indicated in formula (9) and (11): while inferring Eq. (11) from (9), we first calculated $E(y_{ic}^j)$, which is c for perfectly erred sloppy workers. Only when a worker labeled the same task for large enough time, we could claim the expectation value. However, in many applications, only one label is provided from each worker on the same item. We call this fact as "worker-item-uniqueness." Due to this uniqueness, the final aggregated label usually is not as good as what we expected.

These problems lead to the difficulty in making use of perfectly erred workers, since they are randomly biased and we cannot cancel the errors inside the labels. However, it is possible to use the labels of the biased sloppy workers. Passonneau and Carpenter [23] proposed and validated in their work that a highly biased worker, although inaccurate, can provide as much information as a more accurate labeler. They also mentioned that weighted voting schemes are not proper approaches to infer true labels. We first examine prevalent weighted voting approaches to show why this is the case. From the definition, we could see that an observed label y_i^j from a sloppy worker equals to the sum of true label y_i of item i and ε_i^j. Thus, the generic form for weighted learning is presented as follows:

$$\hat{y}_{ic} = \sum_{j=1}^{M} \omega_j (y_i + \varepsilon_i^j) = \sum_j \omega_j y_i + \sum_j \omega_j \varepsilon_i^j,$$
$$\text{where} \quad \sum_j \omega_j = 1 \tag{12}$$

where ε_i^j is the bias made by worker j on item i, ω_j denotes the weight of worker j, and it also represents the reliability of the worker. In state-of-the-art approaches, ω_j is usually set proportional to worker' s accuracy, such as the inversion of worker' s variance $1/v_j$. In truth discovery algorithms, the weights are chosen based on the deviation function (loss function) between true grades and observed grades. In this case, the worker' s weight is actually still in proportion to accuracy. These methods give more weight to workers with higher $\Pr(y_i^j = c|y_i = c)$.

As we know, biased workers usually are inaccurate, which means their labels have low accuracy. In order to deal with the scenarios with highly biased sloppy workers, we propose the iterative self-correcting approach to estimate the worker reliability and true labels.

3.2 Bias detection on sloppy worker

Before we explain the details of the iterative self-correcting algorithm, we first introduce some terms and measurement metrics. For each worker j, assume the accuracy of j as θ_j, and it is calculated as $\theta_j = \Pr(y_i^j = y_i) = \sum_{i=1}^N \Pr(y_i^j = c|y_i = c) \times \Pr(y_i = c)$. If we know the truth labels for all the items, we can try to estimate the worker' s accuracy with the ratio $\theta_j = a/(a+b)$, where a is the number of correct and b is the number of incorrect answers from the worker. Since the workers are assumed to make errors between $+1$ and -1, we use α_j and β_j to denote the conditional probability of j has error/deviation as $+1$ and -1, respectively. In other words, we assign parameters to each worker: if this worker makes errors while labeling an item, the probability of getting deviation $+1$ is α_j, and deviation as -1 is β_j.

$$\begin{aligned}\alpha_j &= \Pr(\varepsilon = 1|y_i^j \neq y_i)\\ \beta_j &= \Pr(\varepsilon = -1|y_i^j \neq y_i)\end{aligned} \tag{13}$$

It is also true that $\alpha_j + \beta_j = 1$. When $\alpha_j >> \beta_j$, it indicates the worker tends to give positive deviations. While if $\beta_j >> \alpha j$, the worker is more likely to have negative deviations. Similarly, once we know the estimated true labels set, we could approximate α_j and β_j. Due to the sparsity problem while approximating all these parameters, it is necessary to fix another way to estimate θ_j, α_j, and β_j. Based on the work of [12,15], we use vanilla Bayesian estimation strategy for estimating them. We treat θ_j as a distribution and presume the conjugate prior for $\Pr(\theta_j)$ as uniform distribution in the [0, 1] interval. Thus, after collecting a correct and b incorrect answers from the worker, the posterior probability $\Pr(\theta_j)$ follows a beta distribution $B(a+1, b+1)$:

$$\Pr(\theta_j) = [\theta_j]^a[1-\theta_j]^b \frac{1}{B(a+1, b+1)} \tag{14}$$

Similarly, for each worker j, we set the prior distribution over the bias (α_j and β_j) as uniform distribution. Then the posterior would be a beta distribution with hyperparameter $b = (b_1, b_{-1})$ for α_j and β_j, where b_1 is the prior $+1$ bias count, and b_{-1} is the prior -1 bias count.

$$\Pr(\alpha_j) = [\alpha_j]^{b_1}[\beta_j]^{b_{-1}} \frac{1}{B(b_1+1, b_{-1}+1)} \tag{15}$$

Since $\alpha_j + \beta_j = 1$, so $\beta_j = 1 - \alpha_j$, then we get:

$$\Pr(\alpha_j) = [\alpha_j]^{b_1}[1-\alpha_j]^{b_{-1}} \frac{1}{B(b_1+1, b_{-1}+1)} \tag{16}$$

To measure the highly biased worker' s reliability, we define a bias score for each of them. Higher bias score indicates the worker with higher bias; thus, the more information he/she would be able to provide as those accurate workers. The bias score (BS) is derived from the information gain:

$$\mathrm{BS}(\alpha_j) = H(\mathrm{bias}|0.5) - H(\mathrm{bias}|\alpha_j) \tag{17}$$

where $H(\mathrm{bias}|0.5)$ denotes the entropy of choosing error as $+1$ with probability of 0.5 and -1 with probability of 0.5. $H(\mathrm{bias}|\alpha_j)$ represents the entropy for the specific worker j with error probability α_j and β_j. Since $\beta_j = 1 - \alpha_j$, we present the entropy based on α_j.

$$\begin{aligned}H(\mathrm{bias}|0.5) &= -[0.5\log 0.5 + 0.5\log 0.5] = -\log 0.5\\ H(\mathrm{bias}|\alpha_j) &= -[\alpha_j \log\alpha_j + (1-\alpha_j)log(1-\alpha_j)]\end{aligned} \tag{18}$$

When $\alpha_j = 0.5$, then $\beta_j = 0.5$, BS$= 0$. It indicates that the worker is not biased. In this case, the worker is actually perfectly erred. When $\alpha_j \to 1$ or $\beta_j \to 1$, $H(\mathrm{bias}|\alpha_j) = 0$; thus, BS$=$ log2. Due to the fact that we treat the worker' s error/bias as distribution, the expected $\mathrm{BS}(\alpha_j)$ can be calculated as (α_j is a random variable):

$$E(\mathrm{BS}(\alpha_j)) = \int \Pr(\alpha_j) \cdot \mathrm{BS}(\alpha_j) \mathrm{d}\alpha_j \tag{19}$$

We know $\Pr(\alpha_j)$ from Eq. (16). After some algebraic manipulations for Eq. (19), we could obtain the expectation of $\mathrm{BS}(\alpha_j)$ as:

$$\begin{aligned}E(\mathrm{BS}(\alpha_j)) &= \log 2 - \Psi(b_1 + b_{-1} + 1)\\ &+ \frac{b_1\Psi(b_1+1) + b_{-1}\Psi(b_{-1}+1)}{b_1 + b_{-1}}\end{aligned} \tag{20}$$

where $\Psi(x)$ is the digamma function. By setting a threshold η_b, we could correct the highly biased workers with

$E(\mathrm{BS}(\alpha_j)) > \eta_b$. However, it is also necessary to take into account the worker' s accuracy, before we correct the bias/error. This is because the error α_j and β_j are assumed as conditional probability in this work. The true error probability (marginal probability) of a worker will depend also on the accuracy of the worker. For example, if a worker' s accuracy is 0.9, thus the possibility of the worker making errors is only at 0.1. Although it is possible that the worker might have a 0.9 possibility with + 1 error for any item he made mistake, we should not correct his labels (by subtracting 1). If we do so, all the items with correct answers will be changed also, thus degrading the final results.

In order to determine whether to correct bias behavior of a worker, we proceed in the following way: as mentioned before, the worker' s accuracy θ_j is assumed as a distribution, and set beta distribution as its conjugate prior. Thus, the error rate of worker is $1 - \theta_j$. We give the posterior mean of the marginal probability of errors, which is the best point estimation for $\alpha_j(1-\theta_j)$ and $\beta_j(1-\theta_j)$: $E(\alpha_j(1-\theta_j)) = \int_{\theta_j=0}^{1}\int_{\alpha_j=0}^{1}((1-\theta_j)\times \Pr(\theta_j))(\alpha_j \Pr(\alpha_j))\mathrm{d}\theta_j \mathrm{d}\alpha_j$, so:

$$E(\alpha_j(1-\theta_j)) = \left[1 - \frac{a+1}{a+b+2}\right] \cdot \frac{b_1+1}{b_1+b_{-1}+2} \tag{21}$$

Similarly, estimation for $E(\beta_j(1-\theta_j))$ could be obtained.

By setting threshold for expected error rate of the worker as η_a, we would be able to determine whether take into account the bias behavior. Only if when $E(\alpha_j(1-\theta_j)) > \eta_a$ or $E(\beta_j(1-\theta_j)) > \eta_a$, we will then consider correcting errors made by the workers based on their bias score. The η_a could be chosen through incremental heuristic way or binary search (binary division heuristic) approach, both were presented while choosing the threshold for separating the randomly erred sloppy workers from biased workers. The value which has the best performance will be selected as the threshold η_a.

Algorithm 1 Iterative self-correcting truth discovery algorithm.

```
Input: Observations from $M$ workers: $\{Y^{(1)}, Y^{(2)}, \ldots, Y^{(M)}\}$, threshold $\eta_a$, and $\eta_b$.
Output: Estimated true labels for $N$ items $Y^{(*)} = \{y_i\}_{i=1}^{N}$, worker weights $W = \{\omega_j\}_{j=1}^{M}$, estimated worker accuracies set $\Theta = \{\theta_j\}_{j=1}^{M}$, and workers bias correcting factor $B$.
1: Initialize the truths $Y^{(*)}$;
2: repeat
3:   Remove spam workers first from the crowd;
4:   for $j = 1 \ldots M$ do
5:     $a \leftarrow$ number of correct labels
6:     $b \leftarrow$ number of incorrect labels
7:     $b_1 \leftarrow$ number of $+1$ error ($y_i^j - yi$ as $+1$)
8:     $b_{-1} \leftarrow$ number of $-1$ error ($y_i^j - y_i$ as $-1$)
9:     Calculate $E(BS(\alpha_j))$ from equation (20)
10:      if $E(BS(\alpha_j)) < \eta_b$ then
11:        $Y^{'(j)} \leftarrow Y^{(j)}$
12:        Continue
13:      else
14:        Calculate $max(E(\alpha_j(1-\theta_j), E(\beta(1-\theta_j)))$
15:        if $maximum > \eta_a$ then
16:          if $\alpha_j > \beta_j$ then
17:            {Correcting $Y^{'(j)}$ by subtracting 1}
18:            $Y^{'(j)} \leftarrow Y^{(j)} - 1$
19:          else
20:            {Correcting $Y^{(j)}$ by adding 1}
21:            $Y^{'(j)} \leftarrow Y^{j} + 1$
22:          end if
23:        end if
24:      end if
25:   end for
26:   {Calculate the worker' s weight}
27:   Update worker' s weight $W$ using equation (24) to infer worker reliability after bias correction, based on the estimated truths.
28:   {Calculate the estimated truths}
29:   for $i = 1 \ldots N$ do
30:     Update the truth of $i^{th}$ item $y_i$ based on observations and current weight estimations, from the worker who contributed to this item, according to equation (25).
31:   end for
32: until Convergence
33: {Calculate the worker' s bias}
34: for $j = 1 \ldots M$ do
35:   Compare the $Y^{'(j)}$ and $Y^{(j)}$, get the bias of worker $j$, insert the bias into B.
36: end for
37: Return $Y^{(*)}$, $W$, $\Theta$, and $B$
```

3.3 Proposed iterative self-correcting framework

The iterative self-correcting algorithm is detailed in Algorithm 1. The algorithm is based on the general principle of optimization-based truth discovery models. As Li et al. [20] concluded in their work, the general principle of truth discovery is captured through the optimization formulation in formula (2).

The proposed framework is an optimization-based method for bias-corrected observations. In Algorithm 1, line 4–25 represents the process of detecting and correcting the biased sloppy workers. After obtaining the de-biased answers $Y^{'(j)}$ for each worker, we discover the truths through the following formulation:

$$\min_{\{\omega_j\},\{y_i\}} f(Y^{(*)}, W) = \sum_{i=1}^{N}\sum_{j=1}^{M} \omega_j d(y_i^j, y_i),$$
$$\text{where} \quad y_i^j \in Y^{'(j)} \quad \text{s.t.} \sum_{j=1}^{M} \exp(-\omega_j) = 1 \tag{22}$$

where the $d(\cdot)$ is called the loss function as mentioned in [20]. Since originally the sloppy worker only errs + 1 or − 1 around the ground truths, it would be the same to either use 0–1 loss or (normalized) squared loss. However, after correcting the bias for the worker, it is possible that worker might err +2 or −2 from the truth answers, for example, if a highly biased worker has 80% of his answers with + 1 error. According to Algorithm 1, all the observed answers from the worker are corrected by subtracting 1 from $Y^{(j)}$. Thus, if the worker has any answer with − 1 error, then the de-biasing would result err −2 from the truths. Thus we propose to use the 0–1 loss in this case:

$$d(y_i^j, y_i) = \begin{cases} 1, & \text{if } \; y_i^j \neq y_i \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

To learn the truth answer set Y and worker' s weights and biases, we optimize the objective function in Eq. (22) by applying the block coordinate descent approach [2]. The approach iteratively conducts two-step procedure until convergence:

(i) Update worker weights while fixing truths of the items $Y^{(*)} = \{y_1, y_2, \ldots, y_N\}$: in this step, the true label for each item is fixed, and we estimate the worker weights W based on the difference between true label and corresponding observed answer. The weight of each worker is calculated through formula (28).

$$W \leftarrow \arg\min_W f(Y^{(*)}, W) \quad \text{s.t.} \sum_{j=1}^{M} \exp(-\omega_j) = 1 \tag{24}$$

(ii) Update the truths y_i while fixing the worker weights and every other item' s truths $\{W, y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_N\}$: in this step, the true label of each item is updated by minimizing the weighted differences between the true labels and observations.

$$y_i \leftarrow \arg\min_{y_i} f(Y^{(*)}, W) \tag{25}$$

This two-step approach corresponds to line 26–31 in Algorithm 1. Step (i) is the worker weight updating process from line 26 to 27. Line 28–31 is the estimated truths updating procedure as described in step (ii).

To derive the source weight in step (i) of the block coordinate descent method, the Lagrange multiplier approach can be utilized on optimization problem in (22):

$$L(W, \lambda) = \sum_{j=1}^{M} \omega_j \sum_{i=1}^{N} d(y_i^j, y_i) + \lambda(\exp(-\omega_j) - 1) \tag{26}$$

By taking the partial derivative with respect to ω_j be 0, it would be able to obtain:

$$\lambda = \frac{\sum_{i=1}^{N} d(y_i^j, y_i)}{\exp(-\omega_j)} \tag{27}$$

From the constraint, we know that $\sum_{j=1}^{M} \exp(-\omega_j) = 1$, thus,

$$\omega_j = -\log \frac{\sum_{i=1}^{M} d(y_i^j, y_i)}{\sum_{j'=1}^{M} \sum_{i=1}^{N} d(y_i^{j'}, y_i)} \tag{28}$$

In order to give an overview of the proposed framework, we present the diagram of the procedure in Fig. 4. We first initialize the truths for all the items. The initialization values are randomly selected from the prior uniform distribution we assumed for the truths. Then we iteratively estimate the true labels as indicated in Algorithm 1, line 2–32. The spam detection approach used in our work for real-world dataset is based on the mean squared error measurement. The algorithm is adopted from the work of Vuurens et al. [32]. The authors proposed a function called $RandomSep$ to recognize the workers who provide random labels. The $RandomSep$ is defined as:

$$RandomSep = \frac{\sum_{v \in V} \varepsilon_r^2}{|V|} \tag{29}$$

where ε_r is the error which represents the ordinal difference between the label a worker given and the estimated true label. V is the collection of all labels offered by each worker. By setting threshold for $RandomSep$, we could detect the spam worker. For example, through setting the threshold value as 1, we could filter out the workers with $RandomSep > 1$ and leave the workers only err + 1 or − 1 around the estimated truths.

The bias correcting process corresponds to line 4–25 in Algorithm 1. Finally, the labels after spam removing and bias correction are used to estimate the truths through coordinate descent algorithm (line 26–31 in Algorithm 1).

4 Experiments

In this section, we present the performance measures for the proposed framework. The experimental results are shown on both synthetic and real-world datasets. We also compare our method with some baseline approaches.

4.1 Experimental setup

4.1.1 Performance measures

In the experiments, the inputs for the models are observations for the items from crowd workers. The aggregating algorithms are applied to output the estimated truths and worker weights. Our proposed methodology and some of the state-of-art methods are conducted in an unsupervised manner, and

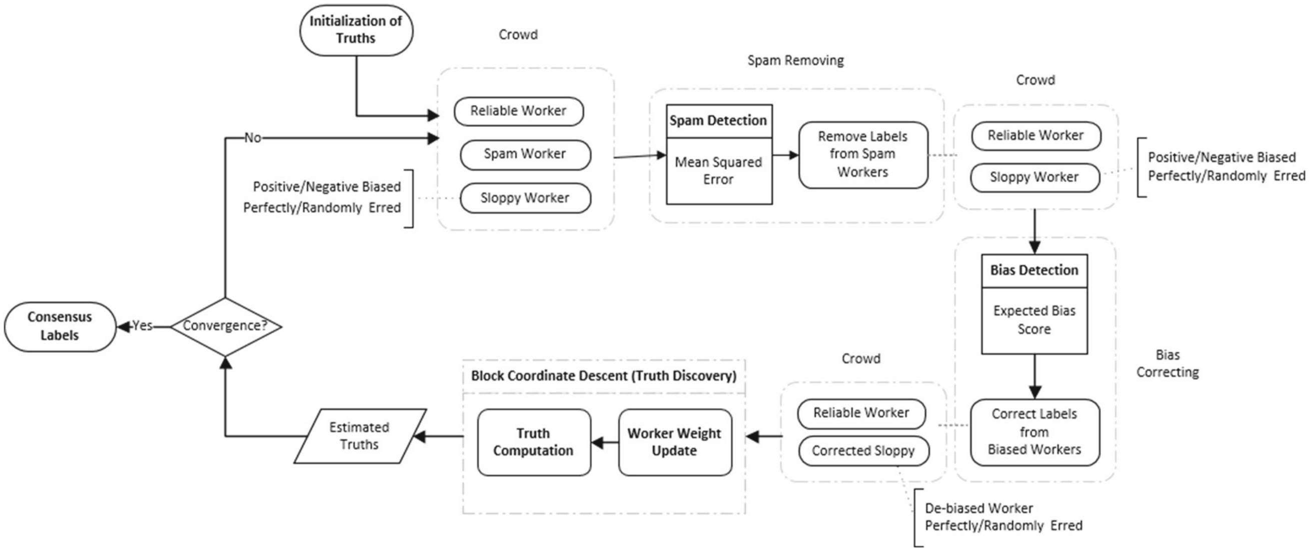

Fig. 4 Diagram of the proposed ITSC-TD framework

the ground truths will be only used for evaluating the performance. Here, we focus on categorical type of data; thus, we adopt the following measures.

- *Accuracy* Accuracy is computed as the percentage of the approach' s output that is the same as the ground truths.
- *F1 measure* It is a weighted average of precision and recall of an approach. *F* measure is different from the accuracy, which equally weights the positive and negative results.

Accuracy and *F1* measure how well the model estimate the truths. The higher the measures, the closer the method' s outputs to the ground truths, and thus the better it performs.

4.1.2 Baseline and proposed methods

The proposed iterative self-correcting truth discovery method in the experiment is denoted as ITSC-TD. We compare our model with the following methods which are designed to infer truths and worker weights in crowdsourcing systems:

Majority voting (MV) simply takes the majority label as the truth for an item. If there is a tie while applying MV, we randomly select an answer from the voted results in our work. Expectation maximization [4] (EM) model proposed by Dawid and Skene (EM-DS) models a confusion matrix for each worker and using EM to estimate the true labels. GLAD [35] models the expertise of each worker through a single parameter, and estimates truths via EM algorithm. Optimization-based truth discovery (TD) [20] is also a weighted voting scheme to infer the true labels.

We implement all the baselines, and the parameters in the models are set according to corresponding papers. The experimental results are compared using different datasets. In the proposed ITSC-TD methodology, we need to set the accuracy (η_a) and bias score (BS) threshold (η_b) to determine a highly biased sloppy worker. Here, we initially set the η_a equals to 0.5, and a_j as 0.5 to obtain η_b. Then 10 values are selected randomly for η_a and a_j respectively, in the range [0.5, 0.7], for parameter sensitivity investigation. The final output are obtained through averaging the results of the different settings.

4.2 Experiments on synthetic dataset

In this section, we focus on experimenting the influence of sloppy workers to the quality of estimated truths. The synthetic data are simulated based on the work of Alfaro and Shavlovsky [5]. We consider 50 workers and 50 items in the simulated environment. Assume each item is labeled by 6 workers, and the labels are chosen from the scale: $\{1, 2, 3, 4, 5\}$. Suppose we would like to simulate the grading process for a set of students' submissions, which are the items, in class. A latent variable model is utilized to approximate the gold truths: let the true quality of each item i denoted as y_i. To simulate all the y_i, we assume the existence of a real-valued latent variable q, where q is normally distributed with mean $\mu = 0$ and standard deviation $\sigma = 1$. The variable y_i results from an "incomplete measurement" of q, where one only determines the interval into which q falls:

$$y_i = \begin{cases} 1, & \text{if} \quad q < \mu - 2\sigma \\ 2 \quad \text{or} \quad 3 \quad \text{or} \quad 4, & \text{if} \quad q \in [\mu - 2\sigma, \mu + 2\sigma] \\ 5, & if \quad q > \mu + 2\sigma \end{cases} \tag{30}$$

In Eq. (30), while $q \in [\mu - 2\sigma, \mu + 2\sigma]$, y_i could choose from 2, 3, and 4. In order to assign one of the labels to y_i, we assume another latent real-valued variable u which is uniformly distributed on [0, 3]. The label of y_i is determined as follows:

$$y_i = \begin{cases} 2, & \text{if} \quad u \in [0, 1) \\ 3, & \text{if} \quad u \in [1, 2) \\ 4, & \text{if} \quad u \in [2, 3] \end{cases} \tag{31}$$

Combining Eqs. (30) and (31), we could obtain the formula for determining y_i:

$$y_i = \begin{cases} \text{if} \quad q < \mu - 2\sigma, \quad 1 \\ \text{if} \quad q \in [\mu - 2\sigma, \mu + 2\sigma], \quad \begin{cases} \text{if} \quad u \in [0, 1), & 2 \\ \text{if} \quad u \in [1, 2), & 3 \\ \text{if} \quad u \in [2, 3], & 4 \end{cases} \\ \text{if} \quad q > \mu + 2\sigma, \quad 5 \end{cases} \tag{32}$$

Two types of workers are simulated: reliable and sloppy worker. Each worker j was assigned a specific accuracy. For reliable workers, the accuracies are uniformly distributed on (0.5, 0.9]. For sloppy workers, their accuracies are uniformly distributed on [0.1, 0.5]. It is also necessary to allocate a bias rate for sloppy workers. The bias rate is the percentage of positive (negative) errors existing among incorrect answers for positive (negative) biased worker. A perfectly erred sloppy worker should have a bias rate as 0.5. The items are then randomly distributed to workers for labeling: each item is required to be labeled by 6 workers, and each worker needs to label 6 items. The observations which are labels obtained from workers are simulated in the following way:

- *If the worker i is reliable* According to the accuracy of i, determine whether to assign true label to the item or not. If an incorrect answer should be given, randomly select label one scale up or down (if possible) of the true label for the item.
- *If the worker i is sloppy* Accuracy is still first utilized to decide to provide a correct or incorrect answer. If wrong answer is supposed to be given, the sloppy worker type (positive, negative, or perfectly erred) and bias rate are then applied to provide the observed label.

Once all the simulated observations are generated, we are able to apply the aggregating models to estimate truths and worker weights from them. For each setting, the data are simulated through 100 runs, and the results are reported as the average over the 100 runs.

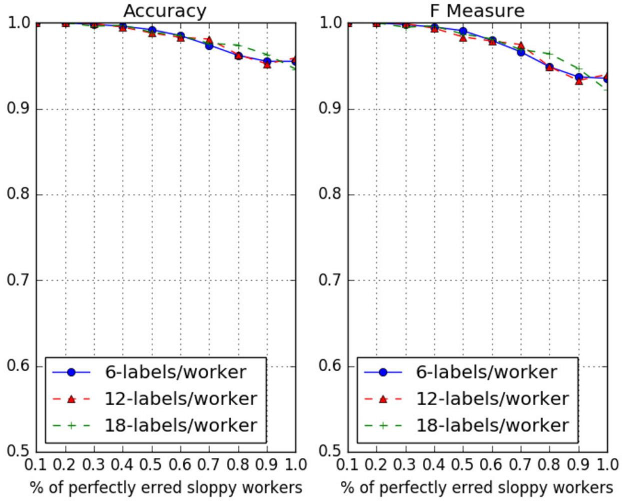

Fig. 5 Influence of perfectly erred sloppy workers with multiple labels/worker on consensus labels

4.2.1 Simulation on perfectly erred sloppy workers

First of all, we investigate the perfectly erred sloppy workers' influence on the consensus results in ideal occasions. Assume each worker provides multiple labels (k) for the item which is assigned for them. Three different settings for k were experimented here: 6, 12, and 18 labels were generated by each worker. That means, if item i is assigned to worker j to get observed labels, j is responsible to provide k labels for i. We assume that workers are independent from each other, and the k labels from the same worker are also independently given. Figure 5 shows the outcome. The consensus labels are given through averaging all the observations. In Fig. 5, the x-axis represents the proportion of the perfectly erred sloppy workers among the crowd, and y-axis shows accuracy and *f1* measure, respectively. Although the metrics start decreasing at the point of 0.5 for the proportion of sloppy workers, we could still obtain high-quality results. For example, at the worst case scenario, with proportion equals to 1, the accuracy is > 0.95, and *f1* measure > 0.9. The results proved that given perfectly erred sloppy workers, under the specific conditions, it is still possible to obtain high-quality consensus results, as we mentioned in Sect. 3.1.2.

Due to the "worker-item-uniqueness," it is necessary to investigate the influence of sloppy workers on aggregating results with only one observation per worker for item i. Figure 6 presents the metrics calculated for the results with single label provided per worker for one item. Comparing to the accuracies and *f1* measures with average (AVG) as aggregating algorithm in Figs. 5 and 6, we can see that without multiple labels per worker, the perfectly erred sloppy workers degrade the aggregated results greatly. The reason that performance in Fig. 6 is worse than Fig. 5 is: while a worker independently provides multiple labels/item, the

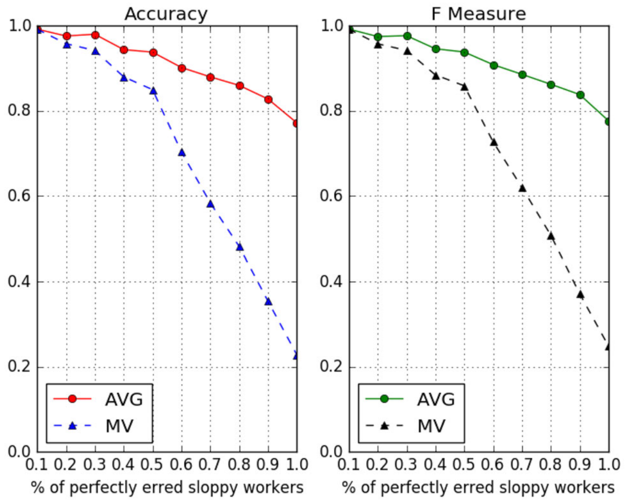

Fig. 6 Influence of perfectly erred sloppy workers with one label/worker on consensus labels

sloppiness within their labels is most likely be canceled with each other (probability of negative error = probability of positive error = 0.5). When only one label is given for each item/worker, the label is randomly given and independent from each worker; thus, the overall probability of obtaining error as + 1 is not necessarily equal to the probability of error − 1. Average algorithm could cancel some of the errors among the grades; however, it is usually used for numerical data instead of categorical ordinal data. Majority voting (MV) is the simplest and one of the prevalent aggregating methods for getting consensus labels for categorical ordinal data. The dotted line with triangle sign in Fig. 6 gives results of majority voting. MV showed even worse results compared to AVG method. While setting the percentage of perfectly erred workers as 100%, the difference of accuracy between AVG and MV could reach around 60%.

4.2.2 Simulation on biased sloppy workers

In order to show the influence of biased sloppy workers on the consensus results, we first conduct the experiments on either one of the two types of biases. Table 5 gives the metrics calculated for the proposed and baseline methods for different proportions of positive biased sloppy workers. In Table 5, when the proportion of positive biased sloppy workers is smaller or equals to 0.5, the selected evaluation metrics show good results among all the listed methods. Among the four baseline models, EM-DS method gives relatively better results compared to other three. If we compare the accuracy and *F* measure of EM-DS with the proposed algorithm while the ratio of positive biased sloppy workers within range [0.1, 0.5], the improvement by utilizing ITSC-TD is only from around 0 to 3%. After increasing the proportion till > 0.5, the proposed ITSC-TD method shows a significant improvement compared to other baseline models for both accuracy and *F* measure. As an example, 14% accuracy improvement can be obtained by applying ITSC-TD compared to EM-DS approach.

Similar process and results could be obtained while experimenting on negative biased sloppy workers. Figure 7 shows the calculated measurements. The solid line with the triangle symbol represents the metrics results for proposed ITSC-TD methodology. The “knee point” in Fig. 7 is the point at the proportion of negative biased sloppy workers $= 0.5$, and it means while the proportion > 0.5, there is a significant improvement on the calculated measurements for proposed algorithm compared to other approaches.

The experiments above assumed the existence of only one type of biased sloppy workers: either positive biased or negative biased. In order to examine the influence of mixture of positive and negative biased sloppy workers, we do simulations on various ratios of these two types of workers. The results are presented in Table 6. The ITSC-TD method has the best performance among all the algorithms applied. As

Table 5 Accuracy and *F* measure calculated for different proportions of positive biased sloppy workers

Method	Metric	Proportion of positive biased sloppy workers									
		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
MV	Accuracy	0.98	0.96	0.92	0.88	0.82	0.62	0.46	0.34	0.24	0.20
	F1	0.98	0.97	0.93	0.89	0.84	0.63	0.52	0.38	0.25	0.23
TD	Accuracy	1.00	0.96	0.96	0.94	0.86	0.66	0.52	0.36	0.26	0.24
	F1	1.00	0.97	0.97	0.95	0.87	0.68	0.57	0.40	0.27	0.27
GLAD	Accuracy	0.98	0.96	0.94	0.94	0.92	0.66	0.56	0.38	0.26	0.22
	F1	0.98	0.97	0.95	0.93	0.92	0.68	0.60	0.38	0.27	0.24
EM-DS	Accuracy	0.98	0.96	0.98	0.95	0.93	0.70	0.60	0.38	0.26	0.24
	F1	0.98	0.96	0.98	0.95	0.94	0.72	0.60	0.42	0.26	0.24
ITSC-TD	Accuracy	1.00	0.98	0.98	0.96	0.95	0.76	0.74	0.50	0.36	0.34
	F1	1.00	0.99	0.99	0.96	0.96	0.80	0.76	0.56	0.39	0.35

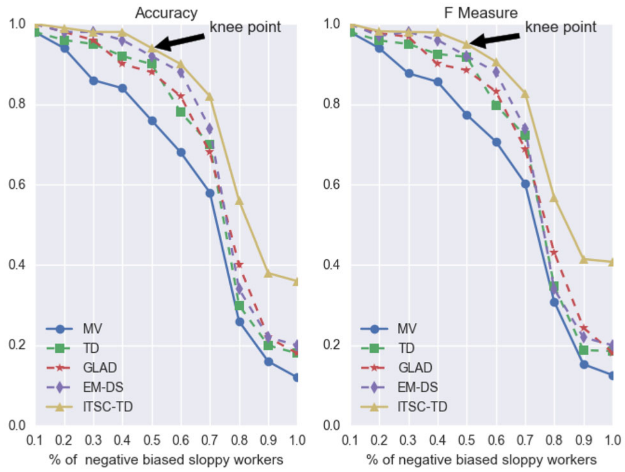

Fig. 7 Results of accuracy and *F* measure for different proportions of negative biased sloppy workers

the results shown in Table 6, when the proportion of sloppy workers > 50%, the proposed approach has significant difference on the performance compared to other methods. The results are consistent with what we obtained with the existence of only one type of biased sloppy workers.

By comparing the results in Tables 5 and 6, we could see that there are differences between them even with the same proportions of sloppy workers. The explanation of the distinctions is that for each table, we utilize the approach mentioned at the beginning of Sect. 4.2 to simulate the sloppy workers through 100 runs. The average over the 100 runs is then used as final outcome, which is presented in the table.

Although the percentage of the biased sloppy workers might be set same in both of the tables, each worker' s individual bias rate could be different. Thus the final results vary. For example, assume a worker with 60% of her labels positively biased, and another worker with 80% of her labels positively biased. The result by taking into account the former worker will be different from the result of using the latter one. In order to see the differences of the performance between only using positive biased workers and having both positive and negative biased workers, we could set one essential baseline for each table. As an example, we set the GLAD approach as baseline model for Tables 5 and 6. The accuracies and *F* measures obtained for ITSC-TD, with high proportion of biased sloppy workers, have significant improvement in contrast to the metrics calculated for GLAD in both of the tables. We could conclude that the proposed methodology is efficient in both scenarios: when there is only one type of biased workers, or both types of biased sloppy workers exist among the crowd.

4.2.3 Simulation on mixed sloppy workers

Before showing the results for datasets with crowd includes all types of sloppy workers, we would like to first investigate the performance of the proposed approach on the worker set with the mixture of perfectly erred sloppy workers and reliable workers. Like all the other baseline models we selected, the proposed ITSC-TD method utilize the labels given by perfectly erred sloppy workers without correcting or removing them. Figure 8 presents the outcome of the experiments.

Table 6 Results for various ratios of mixture of positive and negative biased sloppy workers

Method		MV		TD		GLAD		EM-DS		ITSC-TD	
Prop. of sloppy workers (%)	Ratio of positive and negative biased	Accuracy	*F1*	Accuracy	*F1*	Accuracy	*F1*	Accuracy	*F1*	Accuracy	*F1*
20	0.5:0.5	0.96	0.96	0.98	0.97	0.98	0.98	0.98	0.97	0.98	0.98
	0.8:0.2	0.96	0.97	0.98	0.98	0.92	0.91	0.98	0.98	1.00	1.00
	0.4:0.6	0.97	0.96	0.98	0.98	0.98	0.97	0.98	0.98	0.98	0.98
40	0.5:0.5	0.84	0.85	0.92	0.92	0.90	0.89	0.96	0.96	0.98	0.98
	0.8:0.2	0.80	0.81	0.94	0.93	0.89	0.86	0.94	0.94	0.96	0.96
	0.4:0.6	0.80	0.81	0.88	0.89	0.84	0.83	0.90	0.90	0.95	0.94
50	0.5:0.5	0.75	0.76	0.84	0.84	0.83	0.82	0.90	0.90	0.92	0.91
	0.8:0.2	0.76	0.78	0.88	0.87	0.82	0.81	0.88	0.88	0.94	0.94
	0.4:0.6	0.72	0.77	0.80	0.83	0.78	0.80	0.86	0.86	0.90	0.91
60	0.5:0.5	0.60	0.68	0.80	0.81	0.78	0.79	0.86	0.86	0.92	0.92
	0.8:0.2	0.58	0.57	0.78	0.78	0.69	0.67	0.74	0.74	0.88	0.87
	0.4:0.6	0.60	0.59	0.68	0.67	0.68	0.66	0.78	0.78	0.82	0.80
80	0.5:0.5	0.46	0.49	0.56	0.58	0.54	0.55	0.58	0.58	0.72	0.71
	0.8:0.2	0.32	0.36	0.46	0.48	0.38	0.40	0.44	0.43	0.64	0.64
	0.4:0.6	0.44	0.46	0.56	0.56	0.52	0.51	0.56	0.55	0.66	0.67

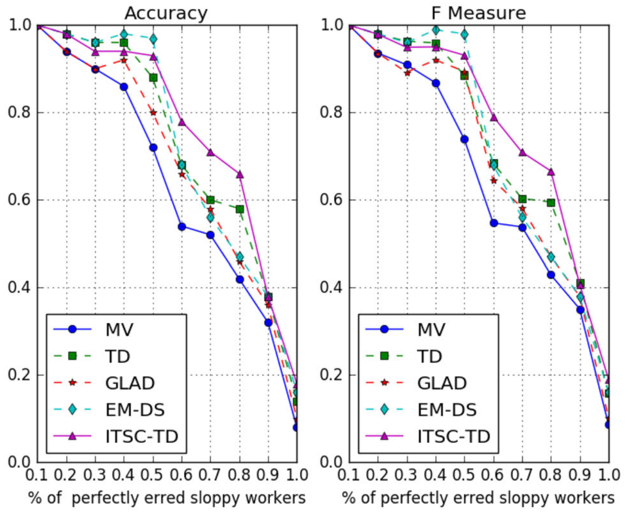

Fig. 8 Results of accuracy and *F* measure for mixture of perfectly erred sloppy workers and reliable workers for proposed and baseline methods

The proportion of the perfectly erred sloppy workers ranges from 0 to 100% as shown in x-axis. From the figure, we could see that compared to other baseline approaches, the proposed method has no significance difference between them. This is due to the fact that the ITSC-TD algorithm does not tackle with perfectly erred sloppy workers.

While varying the ratio of the perfectly erred sloppy workers, we could observe the influence of this type of workers on the quality of the estimated truths, which is presented in Fig. 8. When the percentage $< 50\%$, most of the baseline models (except MV) and our proposed methodology give estimated labels precise enough. In other words, the perfectly sloppy workers have very small influence on weighted voting aggregating algorithms with their proportion less than 50%. One reason is that they make errors within the fault tolerance range, which denotes that they give labels close enough to truths. The other reason is in weighted voting, the weights given to workers are proportional to their accuracies. The perfectly erred sloppy workers thus would have low weights assigned to them. If we change the percentage till most of the workers are perfectly erred sloppy workers ($> 50\%$), the results are greatly degraded as shown in Fig. 8. Especially when the proportion of perfectly erred sloppy workers $\geq 90\%$, all the used weighted voting algorithms' performance is almost same as MV. In order to achieve higher accuracies, it might be useful to filter out this type of workers and remove their labels. This leads to removal of most workers from the crowd. It leaves many items remain unlabeled. Due to the fact that perfectly erred sloppy workers only have errors within the fault tolerance range, in many occasions, it is acceptable to utilize their labels to approximate the truths when there is not enough precise labels available. Based on the analysis, we do not remove the perfectly erred sloppy workers. For randomly erred sloppy workers, same conclusion could be drawn.

Next, we present the results obtained for simulations with different types of sloppy workers incorporated. Table 7 gives the metrics calculated for the mixture of different proportions of positive biased sloppy workers, perfectly erred sloppy workers, and reliable workers. In order to show how biased and perfectly erred sloppy workers influence the performance of the ITSC-TD algorithm, we also present results of the crowd consists of positive biased and reliable workers, as well as the crowd with perfectly erred and reliable workers. Since the results for the mixture of negative biased workers with reliable workers are quite similar to positive biased workers, we only present the calculated metrics for PB:RE.

In Table 7, the highlighted cells represent the methods with best performance. By comparing the experimental results, the ITSC-TD methodology shows superiority than other baseline approaches with existence of biased sloppy workers among the crowd. As an example, while the crowd contains only perfectly erred sloppy workers and reliable workers, TD algorithm gives best performance with PE:RE = 0.4:0.6. Although while PE:RE = 0.8:0.2, ITSC-TD has the largest accuracy value, only 1.9% improvement could be obtained compared to TD approach. It could be further verified by comparing the metrics calculated for $PB : RE$ and $PB : PE : RE$. For example, in Table 7, when the sloppy worker : reliable worker as 0.4: 0.6, if we compare the accuracy obtained for EM-DS and ITSC-TD: 2.1% improvement was achieved by applying ITSC-TD, with PB:RE = 0.4:0.6. However, there was 0% improvement while PB:PE:RE = 0.2:0.2:0.6. This indicates that with same proportion of sloppy workers, better results could be obtained with higher ratio of biased sloppy workers. Same conclusion could be drawn when sloppy worker: reliable worker as 0.8: 0.2. The reason that there is 0% improvement comparing accuracy of EM-DS and ITSC-TD, while PB:PE:RE = 0.2:0.2:0.6, is because no significant improvement could be obtained while the proportion of sloppy workers < 0.5.

4.3 Experiments on real-world dataset

We use real-world data to investigate the effectiveness of the proposed ITSC-TD methodology. Two publicly available datasets are utilized to evaluate the models, namely TREC and AdultContent2 (AC2). The original TREC dataset, which used in [3], has AMT ordinal graded relevance judgments for pairs of search queries and URLs (web pages). Each (Search query, Web page) pair was provided to workers, to ask for topical relevant assessment. The relevance judging has multiple-choice responses as "very relevant (2)," "relevant (1)," and "not relevant (0)." It consists of 98,453 ratings corresponding to 766 workers, 100 queries, and 20,232 (query,

Table 7 Results for the crowd with mixture of positive biased sloppy workers, perfectly erred sloppy workers, and reliable workers

Method	Metric	Ratios of different types of workers					
		PB : RE 0.4 : 0.6	PB : PE : RE 0.2 : 0.2 : 0.6	PE : RE 0.4 : 0.6	PB : RE 0.8 : 0.2	PB : PE : RE 0.6 : 0.2 : 0.2	PE : RE 0.8 : 0.2
MV	Accuracy	0.80	0.88	0.82	0.42	0.48	0.44
	F1	0.83	0.89	0.84	0.43	0.49	0.49
TD	Accuracy	0.92	0.92	**0.96**	0.46	0.54	0.54
	F1	0.92	0.93	**0.97**	0.47	0.56	0.58
GLAD	Accuracy	0.88	0.90	0.92	0.46	0.52	0.48
	F1	0.89	0.89	0.91	0.46	0.53	0.49
EM-DS	Accuracy	0.94	**0.96**	0.94	0.50	0.60	0.52
	F1	0.93	**0.96**	0.94	0.50	0.60	0.52
ITSC-TD	Accuracy	**0.96**	**0.96**	0.94	**0.66**	**0.71**	**0.55**
	F1	**0.95**	**0.96**	0.95	**0.65**	**0.73**	**0.57**

PB Positive biased, *RE* reliable, and *PE* perfectly erred

Table 8 Some statistics about the TREC and AC2 datasets

	Label levels	Instances	Workers	Ratings	Reliable	Spam	Biased	Perfectly/randomly erred
TREC	3	3275	722	19,699	433	0	143	146
AC2	4	333	269	3324	190	19	44	16

URL) pairs. The ratings were on a scale of $\{-2, -1, 0, 1, 2\}$, where - 1 means missing ground truth label, and -2 corresponds to broken link. We processed this dataset by filtering the ratings with value -2 and only take into account the data with gold ground truth. The final dataset contains 3275 (query, URL) instances, 722 workers and 19,699 collected labels. The ratings were mapped from $\{0, 1, 2\}$ to $\{1, 2, 3\}$. This mapping does not affect the values of accuracy and F measure, it is for just easier implementing of all the models.

The AC2 dataset was originally used in [16] includes AMT judgments for websites for the presence of adult content on the page. The judgments are ordinal ratings on G, P, R, X: G for no adult content, P refers to content requires parental guidance, R means content mainly for adults, and X for hardcore porn. The original AC2 dataset consists of 97271 ratings from workers. We filter the data by only taking into account the items with gold truths. This leads to a subset of the data which consists of 3324 ratings from 269 workers for 333 websites. The ratings are mapped from $\{G, P, R, X\}$ to $\{1, 2, 3, 4\}$. The mapping has no influence on the measurement metrics. Due to the fact that we would want to deal with sloppy workers in our work, we further filter the data by choosing the ratings within the set gold truth, gold truth$+$ 1, gold truth$-$ 1. The final AC2 dataset used has 3057 ratings corresponding to 333 items, 265 workers.

There are only three levels for the rating task in TREC dataset, so it might be difficult to define and filter spam workers in this dataset. Table 8 gives some of the statistics for these two datasets. The "Instances" column indicates the number of items to be labeled. The last four columns present the number of different types of workers in the crowd. We give two types of experimental analysis on the real-world datasets: (i) focus on dealing with the sloppy workers. In this analysis, we remove the spam workers by comparing the observed labels with gold truths. In other words, we only keep the worker judgments that are at most one level away from the gold labels. The results are presented in Table 9. (ii) We investigate the impact of both spam and sloppy workers on our proposed methodology. The AC2 dataset is used here for the analysis. As mentioned above, TREC dataset incorporates only three levels of labels, in which no spam worker is defined. The results are shown as Table 10.

Table 9 Comparison between different methods on TREC and AC2 dataset (without spam workers included)

	TREC		AC2	
Method	Accuracy	*F1*	Accuracy	*F1*
MV	0.476	0.481	0.763	0.741
TD	0.483	0.496	0.765	0.743
GLAD	0.498	0.501	0.761	0.739
EM-DS	0.502	0.506	**0.772**	**0.748**
ITSC-TD	**0.513**	**0.524**	0.770	0.746
Detected biased	96		29	

Bold values indicate the best results obtained in each column

Table 9 summarizes the performance of all the methods in terms of accuracy and F measure on TREC and AC2 dataset without spam workers included. The last row of the table

Table 10 Results for AC2 dataset with both spam and sloppy workers incorporated

Method	Accuracy	*F1*
MV	0.756	0.725
TD	0.759	0.735
GLAD	0.752	0.731
EM-DS	0.759	0.736
ITSC-TD	0.762	0.741
Detected spam	16	
Detected biased	21	

gives the number of biased workers recognized while applying ITSC-TD framework. The proposed method in our work showed the best performance compared to other baseline approaches on TREC dataset. Although for AC2, EM-DS out performs the proposed methodology, there is no significant difference between all the methods applied. The different effectiveness of the proposed method on the two datasets can be explained through the statistics presented in Table 8. There are more biased workers (around 23%) in TREC compared to AC2 dataset (around 16%). Thus the ITSC-TD approach gives best quality outcome compared to other baseline models for TREC, while it does not show superiority in AC2 due to the low proportion of biased workers. In addition, most workers (more than 70%) in AC2 belong to reliable worker group, which makes even the MV approach already good enough for approximating the truths. Finally, the experimental results are consistent with conclusion drawn from the synthetic dataset: when the proportion of biased sloppy workers ≤ 0.5, no significant improvement could be obtained.

We next give an insight of the proposed ITSC-TD framework performance on the dataset where both spam and sloppy workers exist. As mentioned earlier, the AC2 dataset is utilized for the analysis. To detect the spam workers, a mean squared error based function, which is explained in Sect. 3.3, is applied. The results are presented in Table 10. The last two rows of the table give the number of spam workers and biased sloppy workers recognized in ITSC-TD framework. Different from the calculated metrics indicated in Table 9 for AC2, in which EM-DS showed superior performance, the proposed ITSC-TD algorithm offered the most precise estimated truths here. It is also clear that the overall quality of the estimated labels is lower in Table 10 than Table 9. The reason for the differences is that the spam workers are included in the experiments here for Table 10. Due to the fact that ITSC-TD approach deals with both spam and biased workers, we could obtain better results by utilizing the proposed method than the baseline models.

5 Conclusion and discussion

In this paper, we introduced a hierarchical categorization of the crowd workers. Instead of investigating binary labeling tasks, our work focuses on more complicated scoring tasks with the scale of multiple and ordinal labels. We investigated the influence of sloppy workers, especially the biased sloppy workers, on the performance of estimating the worker reliability and truth discoveries. An iterative self-correcting algorithm combined with truth discovery (ITSC-TD) approach was proposed to deal with the highly biased crowd workers and infer the truths from observed labels. Both synthetic and real-world datasets were applied to present the effectiveness of the proposed methodology. Finally, the experimental results are compared with several prevalent baseline models, which include MV, TD, GLAD, and EM-DS methods. The comparisons showed that our proposed ITSC-TD outperforms other baseline approaches while the ratio of biased sloppy workers > 0.5, and a significant improvement could be obtained in the simulated datasets. As a result, around 10% to 16% improvement for the accuracy was presented in this case. For real-world data, ITSC-TD is able to achieve better results regarding the selected measurement metrics while proportion of highly biased workers > 0.5, comparing with multiple baseline methods. For example, comparing with baseline methods with TREC dataset, 2 and 8% improvement for accuracy, and 4–9% improvement for F measure can be obtained by using proposed approach.

In the proposed approach, we assume there is no gold truth available while recognizing biased sloppy workers and inferring truths and worker reliability. It would be interesting to investigate the performance of the ITSC-TD method if a subset of the true labels is known. By utilizing these known truths, it might be possible to adjust the proposed method, such as the settings of prior distributions, for better inferencing of the unknown truths. Furthermore, efforts could be made to explore the possibility of utilizing the data from sloppy workers when there is mixture of ordinal and continuous scale labels. The optimization-based truth discovery framework utilized in our work makes it easier to extend the proposed ITSC-TD approach to mixed ordinal and continuous data types.

References

1. Aydin, B.I., Yilmaz, Y.S., Li, Y., Li, Q., Gao, J., Demirbas, M. Crowdsourcing for multiple-choice question answering. In: AAAI, pp. 2946–2953 (2014)
2. Bertsekas, D.P. Non-linear programming. In: Athena scientific (1999)
3. Buckley, C., Lease, M., Smucker, M.D., Jung, H.J., Grady, C. Overview of the TREC 2010 relevance feedback track (notebook).

In: The Nineteenth Text Retrieval Conference (TREC) Notebook (2010)
4. Dawid, A.P., Skene, A.M.: Maximum likelihood estimation of observer error-rates using the EM algorithm. Appl. Stat. **28**, 20–28 (1979)
5. De Alfaro, L., Shavlovsky, M. Crowdgrader: a tool for crowdsourcing the evaluation of homework assignments. In: Proceedings of the 45th ACM Technical Symposium on Computer Science Education. ACM, pp. 415–420 (2014)
6. Dekel, O., Shamir, O.: Vox populi: collecting high-quality labels from a crowd (2009)
7. Demartini, G., Difallah, D.E., Cudré-Mauroux, P.: Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In: Proceedings of the 21st International Conference on World Wide Web. ACM, pp. 469–478 (2012)
8. Dong, X.L., Berti-Equille, L., Srivastava, D.: Integrating conflicting data: the role of source dependence. Proc. VLDB Endow. **2**(1), 550–561 (2009)
9. Ertekin, S., Hirsh, H., Rudin, C.: Learning to predict the wisdom of crowds (2012). Preprint. arXiv:1204.3611
10. Galland, A., Abiteboul, S., Marian, A., Senellart, P.: Corroborating information from disagreeing views. In: Proceedings of the third ACM International Conference on Web Search and Data Mining. ACM, pp. 131–140 (2010)
11. Gao, J., Li, Q., Zhao, B., Fan, W., Han, J.: Truth discovery and crowdsourcing aggregation: a unified perspective. Proc. VLDB Endow. **8**(12), 2048–2049 (2015)
12. Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B.: Bayesian Data Analysis, vol. 2. CRC Press, Boca Raton (2014)
13. Gneezy, U., Rustichini, A.: Pay enough or don't pay at all. Q. J. Econ. **115**(3), 791–810 (2000)
14. Hama, A.: Predictably irrational: the hidden forces that shape our decisions. Mank. Q. **50**(3), 257 (2010)
15. Ipeirotis, P.G., Gabrilovich, E.: Quizz: targeted crowdsourcing with a billion (potential) users. In: Proceedings of the 23rd International Conference on World Wide Web. ACM, pp. 143–154 (2014)
16. Ipeirotis, P.G., Provost, F., Wang, J.: Quality management on amazon mechanical turk. In: Proceedings of the ACM SIGKDD Workshop on Human Computation. ACM, pp. 64–67 (2010)
17. Kamar, E., Kapoor, A., Horvitz, E.: Identifying and accounting for task-dependent bias in crowdsourcing. In: Third AAAI Conference on Human Computation and Crowdsourcing (2015)
18. Karger, D.R., Oh, S., Shah, D.: Iterative learning for reliable crowdsourcing systems. In: Advances in Neural Information Proceeding Systems, pp. 1953–1961 (2011)
19. Kazai, G., Kamps, J., Milic-Frayling, N.: Worker types and personality traits in crowdsourcing relevance labels. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management. ACM, pp. 1941–1944 (2011)
20. Li, Q., Li, Y., Gao, J., Zhao, B., Fan, W., Han, J.: Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. ACM, pp. 1187–1198 (2014)
21. Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., Han, J.: A survey on truth discovery. ACM Sigkdd Explor. Newsl. **17**(2), 1–16 (2016)
22. Meng, C., Jiang, W., Li, Y., Gao, J., Su, L., Ding, H., Cheng, Y.: Truth discovery on crowd sensing of correlated entities. In: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems. ACM, pp. 169–182 (2015)
23. Passonneau, R.J., Carpenter, B.: The benefits of a model of annotation. Trans. Assoc. Comput. Linguist. **2**, 311–326 (2014)
24. Pasternack, J., Roth, D.: Knowing what to believe (when you already know something). In: Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics, pp. 877–885 (2010)
25. Raykar, V.C., Yu, S.: Eliminating spammers and ranking annotators for crowdsourced labeling tasks. J. Mach. Learn. Res. **13**, 491–518 (2012)
26. Raykar, V.C., Yu, S., Zhao, L.H., Valadez, G.H., Florin, C., Bogoni, L., Moy, L.: Learning from crowds. J. Mach. Learn. Res. **11**, 1297–1322 (2010)
27. Sheng, V.S., Provost, F., Ipeirotis, P.G.: Get another label? Improving data quality and data mining using multiple, noisy labelers. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 614–622 (2008)
28. Snow, R., O'Connor, B., Jurafsky, D., Ng, A.Y.: Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 254–263 (2008)
29. Von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, pp. 319–326 (2004)
30. Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., Blum, M.: Recaptcha: human-based character recognition via web security measures. Science **321**(5895), 1465–1468 (2008)
31. Vondrick, C., Patterson, D., Ramanan, D.: Efficiently scaling up crowdsourced video annotation. Int. J. Comput. Vis. **101**(1), 184–204 (2013)
32. Vuurens, J., de Vries, A.P., Eickhoff, C.: How much spam can you take? An analysis of crowdsourcing results to increase accuracy. In: Proceedings of hte ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR11), pp. 21–26 (2011)
33. Wauthier, F.L., Jordan, M.I.: Bayesian bias mitigation for crowdsourcing. In: Advances in Neural Information Processing Systems, pp. 1800–1808 (2011)
34. Welinder, P., Branson, S., Perona, P., Belongie, S.J.: The multidimensional wisdom of crowds. In: Advances in Neural Information Processing Systems, pp. 2424–2432 (2010)
35. Whitehill, J., Wu, T.F., Bergsma, J., Movellan, J.R., Ruvolo, P.L.: Whose vote should count more: optimal integration of labels from labelers of unknown expertise. In: Advances in Neural Information Processing Systems, pp. 2035–2043 (2009)
36. Yan, Y., Rosales, R., Fung, G., Dy, J.G.: Active learning from crowds. ICML **11**, 1161–1168 (2011)
37. Yin, X., Han, J., Philip, S.Y.: Truth discovery with multiple conflicting information providers on the web. IEEE Trans. Knowl. Data Eng. **20**(6), 796–808 (2008)
38. Zaidan, O.F., Callison-Burch, C.: Crowdsourcing translation: professional quality from non-professionals. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, Association for Computational Linguistics, pp. 1220–1229 (2011)
39. Zhang, J., Sheng, V.S., Li, Q., Wu, J., Wu, X.: Consensus algorithms for biased labeling in crowdsourcing. Inf. Sci. **382**, 254–273 (2017)
40. Zhou, D., Basu, S., Mao, Y., Platt, J.C.: Learning from the wisdom of crowds by minimax entropy. In: Advances in Neural Information Processing Systems, pp. 2195–2203 (2012)