Incorporating Feature Labeling into Crowdsourcing for More Accurate Aggregation Labels

Yili Fang, Zhaoqi Pei, Xinyi Ding, Wentao Xu, and Tao Han(✉)

School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China
{fangyl,zhaoqipei,xding,xuwentao6666,hantao}@zjgsu.edu.cn

Abstract. Crowdsourcing is a popular way of collecting crowd wisdom and has been deployed in various senarios. Effective *answer collection* and *answer aggregation* are two important crowdsourcing topics as workers may give incorrect responses. For difficult tasks, workers tend to implicitly use task related information during *answer collection*, and those information could play an important role in aggregating high-quality results. For example, the identification of the size and hair style of one dog in a picture is a simple and necessary prerequisite step for dog breed labeling. However, most existing methods ignore those task related information and fail to achieve high quality data.

In this study, we propose a framework that incorporates the answers of corresponding tasks from workers and their labeling to object features, which we believe are critical task related information for *answer aggregation*. Then, we propose a novel generative probability graph model that can infer the task answers by exploiting label features, as well as worker ability and their responses. We use EM algorithm to estimate model parameters and infer true answers. Experimental results demonstrate that incorporating task related information can greatly improve the accuracy of *answer aggregation*. Compared with state of the art ones that ignore these information, our methods could achieve about 15.9%–36.8% improvement in accuracy.

Keywords: Crowdsourcing · Answer collection · Answer aggregation · Probability graph model

1 Introduction

Crowdsourcing has been successfully applied in solving a large number of practical tasks [4,20,21], such as data annotation [25], text translation [34], sentiment analysis [19], etc. It is a valid and inexpensive way for researchers to collect labels from non-expert crowds in open Internet based marketplaces. But the quality of collected data provided by the workers might be very low due to a variety of factors, including their background, skills, and motivation. A commonly used

H. Gao et al. (Eds.): CollaborateCom 2022, LNICST 461, pp. 306–324, 2022.
https://doi.org/10.1007/978-3-031-24386-8_17

method is to collect multiple labels for the same task from different workers, then apply a label aggregation algorithm to infer the true label for each task. In general, crowdsourcing involves *answer collection* and *answer aggregation*. In *answer collection*, due to the unreliability of workers, requesters usually distribute one task redundantly to multiple workers on a crowdsourcing platform, such as *Amazon's Mechanical Turk*[1]. Workers get paid by processing the assigned tasks and submit their responses. In *answer aggregation*, aggregation algorithms are used to deduce high-quality results from redundant noisy answers submitted by workers. In the past few years, researchers have proposed different models to improve the accuracy of aggregation results.

In crowdsourcing, the task related information plays an important role in the answer collection stage and in answer aggregation. In *answer collection*, workers can better complete the task if given these related information. For example, if we want to annotate a picture of an Alaskan Malamute, some workers without relevant knowledge might label it as a Poodle or other dog breeds with large size. In fact, workers often need to use these related information explicitly or not before he/she can give a label. However, existing methods try to obtain high-quality data by assigning tasks to workers with higher abilities [6,7,23], but often ignore some useful task related information, which is important in answer aggregation. We refer to the collection of these task related information as hidden sub-tasks (HSTs) in crowdsourcing. In fact, workers may have implicitly carried out these HSTs when handling one task and they could play an important role in inferring high-quality results. In the task of dog breed classification, the identification of size and hair style of one dog could be thought of as HSTs. In *answer aggregation*, task related information also plays important roles. In the dog labeling example, methods like MV will ignore the fact that *Alaskan Malamute* tends to be large size while *Poodle* tends to be small. But such features could often help exclude some answers thus improve the overall accuracy. Although a lot of aggregation methods have been proposed to improve the quality of the answers provided by workers, they often ask workers to only focus on the final results and ignore task related information (such as the features of an object) [2,24,27,33]. In short, how to collect and harness the task related information to improve the quality of the overall aggregation results is the main focus of the paper.

In this paper, we focus on using feature labels collected from HSTs to improve the overall aggregation results in labeling tasks. In the *answer collection* phase of crowdsouring, we propose a framework to ask workers to explicitly carry out HSTs and record their responses. This framework allows us to collect object's category as well as feature information from workers. For example, we require workers to give their answers not only to dog breed label but also labels for dog features like size, hair style, etc. In the *answer aggregation* phase, we propose a probabilistic model called the Generative model of Labels, Abilities, and Features (GLAF), in which we factorize the conditional probability of the most accurate answer with respect to current object labels, worker abilities, features, and the relationship between object labels and feature labels. In order to represent the effect of feature

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

labels to the aggregation results, we use a matrix to represent the relationship between object labels and feature labels. Each element in the matrix represents the probability that one class of objects has a certain feature. The inferred results, workers' abilities and other parameters used in the model are estimated using EM (expectation-maximization) algorithm. Our main contributions are summarized as follows:

- We propose a framework, which allows workers to collect as many feature labels as possible based on the fact that workers are usually better at identifying features than categories.
- We propose a novel generative probability model GLAF, which can infer the answers of tasks by exploiting the relationship between object labels and feature labels.
- We conduct extensive experiments and the results show that incorporating feature labels could significantly improve the results. Compared with other state of the art ones without these information, our model is also superior.

The rest of the paper is organized as follows. Section 2 discusses the related works. Section 3 formalizes the problem studied and outlines our framework for *answer aggregation*. Section 4 introduces the details of our approach. The experimental results are shown in Sect. 5 and we conclude the paper in Sect. 6.

2 Related Work

Since the advent of crowdsourcing [9], it has been successfully applied in artificial intelligence that leverages human intelligence to improve machine performance. It has also been applied in many applications such as entity resolution [31], image annotation [25], audio recognition [10], video annoation [30], etc.

In the *answer collection* phase, obtaining answers for the same task from multiple workers is a simple and direct way to improve the data quality [6,7,23]. Task design and rational assignment of tasks to workers on the crowdsourcing platform are also helpful in collecting high-quality data. At the same time, many works [8,17,22] have proved that collecting more comprehensive information about workers and tasks can help improve the quality of *answer aggregation*. Oyama et al. proposed a framework to collect labels and task related information (such as confidence scores) provided by crowdsourcing workers to improve decision accuracy [22]. Li et al. demonstrated an interactive programming tookit that is a unified solution for answering the crowdsourced top-k queries to control the quality of labels [17]. Hoßfeld et al. developed *two-stage QoE crowdtesting design* which leads to more reliable results [8].

In the phase of *answer aggregation*, not only the true answers of tasks are inferred, but also some other potential useful information. This process is also called truth inference. The agnostic label aggregation algorithms that solely use crowdsourced multiple noisy labels have been extensive researched. These works tried to model the complexities of the crowdsourced labeling from different angles, such as confusion matrices [24,29,36,37],reliability [1,3,13,15,33],

intentions [1,14], and difficulties of instances [1,14,32,33], and biases [12,32]of labelers. MV is a simple aggregation method, which takes the value with the highest number of votes as the true label of the task [26]. In addition, Dawid and Skene proposed a classic algorithm DS based on maximum likelihood estimation and EM algorithm. They model the diagnosis results of multiple doctors on the same patient and use a confusion matrix to represent the performance of workers to infer the true label of each task [2]. Inspired by DS, Smyth et al. used a similar method on the Venus image dataset to infer the true value of subjective labels [28]. Raykar et al. proposed a Bayesian method, RY, which adds a specific prior to each category. In this method, crowdsourcing workers have different biases for positive and negative categories and use two parameters to model them [24]. By incorporating worker abilities and object difficulties, Li et al. proposed a family of models to learn the object embeddings from crowdsourced triplet similarity comparisons [16]. A multiple noisy label distribution propagation (MNLDP) method is proposed in [11]. This method at first estimates the multiple noisy label distribution of each instance from its multiple noisy label set. And then it propagates its multiple noisy label distribution to its nearest neighbors. Finally, each instance assimilates a part of the multiple noisy label distributions from its nearest neighbors and at the same time keeps a part of its own original multiple noisy label distribution. All of the above methods [2,16,24,28] only model workers' answers to tasks to infer the true value, while ignoring other task related information such as feature information that may have potential value for inferring the true value of the task. Our research makes full use of feature labels and the relationship between object labels and feature labels.

There are also some existing works that use task related information to help infer the true value. Zhang et al. proposed a cluster-based label aggregation algorithm GTIC [35]. Their method uses a feature vector to represent the task and then use a clustering method to cluster all tasks according to the feature vector of each task. Hang et al. proposed CrowdMKT on the basis of SKT. This model uses knowledge transfer to learn high-level feature vectors of tasks from multiple related data sources and then introduces a probability model to jointly model tasks with high-level features, workers and their annotations [5]. Welinder et al.proposed a method that deals with image formation and annotation process. Each image has different features that are represented in an abstract Euclidean space. Each annotator is modeled as a multidimensional entity with variables representing competence, expertise and bias [32] . PLA (Prediction-based Label Aggregation) is proposed to intelligently aggregate the crowd wisdom and the predicted answers to improve the performance of label aggregation in [18].

Different from these studies, we collect noisy answers and feature labels of the HSTs provided by workers and model worker's abilities and HSTs respectively. After having these data prepared, we focus on incorporating feature labels and its relationship with object labels into *answer aggregation* for more accurate aggregation results.

3 Problem Formulation

In this section, we first define the problem of Turth Inference with feature labels (TIF), then discuss our framework that incorporates feature labeling. The notations used are displayed in Table 1.

Table 1. Table of Notations

Notation	Description
W	Worker set consist of $\|W\|$ workers
T	Task set consist of $\|T\|$ tasks
Ω	Category set of a task
$\|W\|$	Worker numbers
$\|T\|$	Task numbers
w_j	Anchor worker
t_i	Anchor task
E'	Feature set of a task
E	Feature label space of the task
l_{ij}^z	Category answer of task t_i provided by worker w_j
l_{ij}^e	Feature answer of task t_i provided by worker w_j
L=$\{l_{ij}^z, l_{ij}^e\}$	All set of answers

3.1 Definition of TIF

In our study, we ask workers to give not only answers to tasks but also their labels to different features. We use these noisy data to infer the final results. We call our problem the Truth Inference with feature labels problem (TIF) and define it formally as follows.

Definition 1 (Truth Inference with feature labels (TIF) problem). *Let $T = \{t_i | i \in I_T\}$ be the unlabeled task set, $W = \{w_j | j \in I_W\}$ be the workers set, $\Omega = \{c_k | k \in I_\Omega\}$ be the answer domain set. The problem of TIF is to find a function $f : \Omega^{|T|\times|W|} \longrightarrow \Omega^{|T|}$, which generates the most precise answer from all the labels provided by workers for each task.*

Without loss of generality, we consider label aggregation in a classification crowdsourcing task. Let $E' = \{e | e \in I_e\}$ denotes feature labels for a given task. For example, when a worker is working on a task to classify dog breeds, the elements in this set can be traits such as the dog's hair style, body size, etc. Let the vector $E = \{< e_1, e_2, ..., e_m, ..., e_n > | e_m \in \Omega_{e_m}\}$ be the feature label space of the task, and the element in this vector is the specific feature label values of an object. For the dog breeds classification example, the element in this vector can be (long hair, big size). We denote the label set $L = \{L_1, L_2, ..., L_n\}$ where

$L_i \in L$ contains labels that workers give to t_i. Namely, for $\forall t_i \in T$ we get label set $L_i = \{l_{ij}^z, l_{ij}^e | i \in I_T, j \in I_W, e \in I_e\}$ from workers. The problem of TIF is to find a function $f : \Omega^{|T|\times|W|} \times E^{|T|\times|W|} \longrightarrow \Omega^{|T|}$, which generates the most specific label from all the labels provided by workers for each task.

Let $F = \{f | f : \Omega^{|T|\times|W|} \times E^{|T|\times|W|} \longrightarrow \Omega^{|T|}\}$ be the universal set of aggregation algorithms of TIF. Then with the well-defined value function $v : F \longrightarrow \mathbf{R}$ which measures quality of the algorithms, we can formulate the aggregation problem of TIF as to find a function f^*:

$$f^* = \arg\max_{f \in F} v(f). \tag{1}$$

For instance, if there are 100 tasks and 10 workers, one worker has to choose from 4 candidate answers. Each answer has two features and each feature has two possible feature values, then the aggregation algorithm is to find a function with a $100 \times 10 \times 3$ (1 for category, 2 for feature) answer matrix as input and a 100 dimensional answer vector as output.

3.2 Framework of Crowdsourcing with Feature Labeling

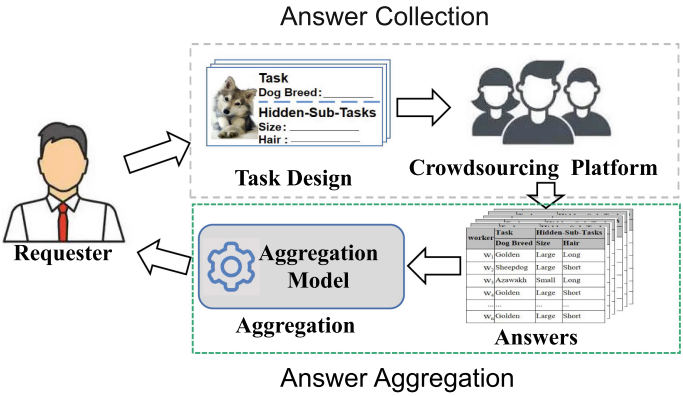


Fig. 1. Framework of crowdsourcing with feature labeling.

To exploit the relationship between object's class and the corresponding feature labels, we propose a novel crowdsourcing framework with feature labeling as shown in Fig. 1. The steps of this workflow are listed as follows.

– **Step 1:** A requester first prepares tasks including the objects that need to be classified and the related information that workers needs to process(such as hair style, size of a dog). Then the requester needs to design the detailed task interface using tools provided by the platform before publishing tasks.

Crowdsourcing platforms like Amazon's Mechanical Turk usually provide flexible tools and APIs that allow requesters to customize the way tasks are presented.

- **Step 2:** Once the requester published the tasks. The platform will assign them to workers based on the requester's requirements and the platform's own policies.
- **Step 3:** For each received task, a worker provides both the answer as well as the feature labels of the corresponding task. One worker will get paid once the platform accepts his/her answers.
- **Step 4:** After collecting all the answers from workers, we run our model with feature labels to infer the aggregated result for each task. Finally, all the aggregated results are returned to the requester.

Using this framework, requesters can easily collect answers to tasks, as well as critical related information that could help improve the aggregation results.

4 GLAF Model

In this section, we introduce a novel probabilistic model that incorporates object label, its relationship with feature labels, as well as worker ability (Sect. 4.1). Then we explain how to estimate the parameters and variables of GLAF model using EM algorithm (Sect. 4.2).

4.1 Probabilistic Modeling

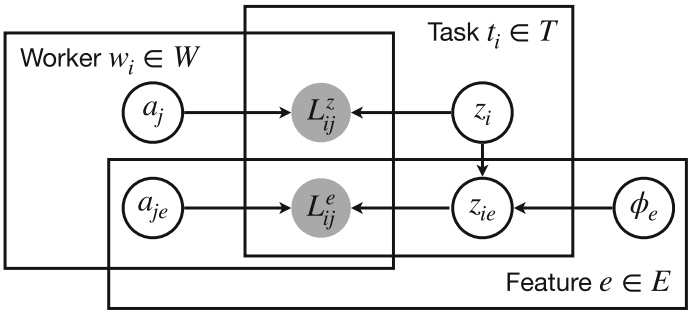


Fig. 2. Probability graph of GLAF model.

Consider a dataset of n tasks, each of which has one of many possible candidate answers. Our goal is to determine the true answer z_i of each task i by querying from workers. The answers depend on several causal factors: (1) the abilities of workers; (2) the relationship between object label and feature labels; and (3) the true answer.

The ability of each worker w_j is modeled by parameter $a = \{a_j, a_{je}\}$. Here $a_j \in (-\infty, +\infty), a_j = +\infty$ means the worker always labels images correctly; $-\infty$ means the worker always labels images incorrectly; $a_j = 0$ means that the worker cannot discriminate the candidate answers. And a_{je} is the same as a_j. The relation between object labels and feature labels is modeled by a parameter matrix $\phi = \{\phi^e | e \in I_E\}$, the value $\phi^e_{kf_{eg}}$ in the matrix represents the probability that the answer k is corresponding with feature label f_{eg}. $\phi^e_{kf_{eg}} = 0$ means that the answer k wouldn't appear together with feature label f_{eg}; $\phi^e_{kf_{eg}} = 1$ means that the answer k would appear with feature label f_{eg} all the time. Let $Z = \{z_i, z_{ie} | i \in I_T, e \in I_E\}$ be the task truth set. Here z_i is the answer truth value of the task t_i and z_{ie} is the truth value of feature e of the task t_i.

The larger a_j is, the more likely one worker could answer the task correctly. So we can present the conditional probability that l^z_{ij} is correct given z_i, a_j with a softmax function as follows:

$$p(l^z_{ij} = z_i | z_i, a_j) = \frac{1}{1 + (K-1)e^{-a_j}}, \tag{2}$$

where K is the number of answers. Note that $p(l^z_{ij} = z_i | z_i, a_j)$ increases with the increase of a_j. When a_j tends to be positive infinity, $p(l^z_{ij} = z_i | z_i, a_j) = 1$ means worker w_j always gives the correct answer. Generally, when ability a_j of worker w_j and category z_i of task t_i are given, we can compute the probability that worker w_j correctly answers category of task t_i by Eq. 2.

The larger a_{je} is, the more likely one worker could answer the task correctly. So the conditional probability that l^e_{ij} is correct given z_i, z_{ie}, a_{je} can be expressed as:

$$p(l^e_{ij} = z_{ie} | z_{ie}, z_i, a_{je}) = \frac{1}{1 + (K_e - 1)e^{-a_{je}}}, \tag{3}$$

where K_e is the number of the alternative of feature e. Again, $p(l^e_{ij} = z_{ie} | z_{ie}, z_i, a_{je})$ increases with the increase of a_{je}. Generally, when ability a_{je} of worker w_j, category z_i of task t_i and feature z_{ie} of task t_i are given, we can compute the probability that worker w_j correctly answers feature e of task t_i by Eq. 3.

We present the relation between object labels and features given ϕ^e, z_i with a number $\phi^e_{kf_{eg}} \in [0, 1]$ as follows:

$$p(z_{ie} = f_{eg} | z_i = c_k, \phi^e) = \phi^e_{kf_{eg}}, \tag{4}$$

If the category z_i of task and the relation matrix ϕ^e between object labels and features are given, we can compute the probability that task t_i has the feature f_{eg} by Eq. 4.

Let $L = \{l^z_{ij}, l^e_{ij} | i \in I_T, j \in I_W, e \in I_E\}$, $\theta = \{a, \phi\}$. We want to find the optimal θ^* that maximizes $P(L|\theta)$. Our objective function is as follows:

$$\theta^* = \arg\max_{\theta} \ln P(L|\theta), \tag{5}$$

where $\ln P(L|\theta)$ is the log likelihood function we want to maximize. In detail, we compute $\ln p(L|\theta)$ as:

$$\begin{aligned}\ln p(L|\theta) = \sum_i \ln \Big[\sum_k p(z_i|\theta) \prod_j p(l_{ij}^z|z_i,\theta) \\ \prod_{e,g} p(z_{ie} = f_{eg}|z_i,\theta) \prod_j p(l_{ij}^e|z_i,z_{ie},\theta)\Big].\end{aligned} \tag{6}$$

We maximize $\ln p(L|\theta)$ to learn the parameters θ and infer the true answers of tasks (Sect. 4.2).

4.2 Inference

We formally introduced the GLAF model as shown in Fig. 2 in Sect. 4.1. In our model, we have l_{ij}^z, l_{ij}^e being the workers answers. The unobserved variables are the true labels z_i, z_{ie}, ability parameters a_j, a_{je} and relation matrix ϕ^e. Our goal is to find the posterior distribution of z_i and select the label z_i with the maximum a posterior estimation as the final answer to task t_i.

For simplicity, the prior distribution of z_i is set to be an uniform discrete distribution over label domain Ω. In addition, we ignore the prior of a_j and a_{je} and the elements in the mapping relationship matrix between object labels and feature labels are initially set as $\frac{1}{K_e}$. Finally, we use EM algorithm to obtain maximum likelihood estimates of the parameters of a_j, a_{je}, ϕ^e.

E Step: Let $l_i = \{l_{ij}^z, l_{ij}^e | j \in I_W, e \in I_E\}$. Then for $\forall i \in I_T$, we compute posterior probability $p(z_i = c_k|L,a,\phi)$ as:

$$\begin{aligned}&p(z_i = c_k|L,a,\phi) \\ \propto &p(z_i = c_k|a,\phi)p(l_i|z_i = c_k,a,\phi) \\ \propto &p(z_i = c_k|a,\phi)\Big[\prod_j p(l_{ij}^z|z_i = c_k,a_j,\phi) \\ &\cdot \prod_e \sum_g p(z_{ie} = f_{eg}|z_i = c_k,\phi^e) \prod_j p(l_{ij}^e|z_i = c_k,a_{je})\Big]\end{aligned} \tag{7}$$

Here we assume features are conditional independent. For $\forall i \in I_T, \forall e \in I_E$, we compute posterior probability
$p(z_{ie}|L, z_i = c_k, a_{je}, \phi^e)$ as:

$$\begin{aligned}&p(z_{ie} = f_{eg}|L, z_i = c_k, a_{je}, \phi^e) \\ \propto &p(z_{ie} = f_{eg}|z_i = c_k, \phi^e)p(l_i^e|z_{ie} = f_{eg}, z_i = c_k, \phi^e) \\ \propto &p(z_{ie} = f_{eg}|z_i, \phi_e) \prod_j p(l_{ij}^e|z_{ie} = f_{eg}, z_i = c_k, a_{je})\end{aligned} \tag{8}$$

If answer set L of workers, category z_i of task, ability a_{je} of worker and the relation matrix ϕ^e between object labels and features are given, we can compute

the conditional probability distribution of feature e of task t_i by Eq. 8 and its category z_i.

M Step: Let $L = \{l_{ij}^z, l_{ij}^e | i \in I_T, j \in I_W, e \in I_E\}, Z = \{z_i, z_{ie} | i \in I_T, e \in I_E\}$. We compute standard auxiliary function Q:

$$\begin{aligned}
&Q(a^{old}, \phi^{old}, a, \phi) \\
=&E[\ln p(L, Z|a, \phi)] \\
=&\sum_i E[\ln p(z_i)] + \sum_i E[\ln p(l_i|z_i, a, \phi)] \\
=&Const + \sum_{i,k} \ln p(l_i^z|z_i, a, \phi)\beta_{ik} + \sum_{i,k} \beta_{ik} \sum_e \ln p(l_i^e|z_i, a, \phi) \\
=&Const + \sum_{i,k,j} \ln p(l_{ij}^z|z_i, a, \phi)\beta_{ik} + \sum_{i,k} \beta_{ik} \sum_{e,g} q_{ikeg} \ln p(l_i^e, z_{ie}|z_i, a, \phi) \\
=&Const + \sum_{i,k,j} \ln p(l_{ij}^z|z_i, a, \phi)\beta_{ik} + \sum_{i,k} \beta_{ik} \sum_{e,g} q_{ikeg} \ln p(z_{ie}|z_i, a, \phi) \\
&+ \sum_{i,k} \beta_{ik} \sum_{e,g,j} q_{ikeg} \ln p(l_{ij}^e|z_{ie}, z_i, a, \phi),
\end{aligned} \tag{9}$$

where

$$\begin{aligned}
\beta_{ik} =& p(z_i = c_k|L, a^{old}, \phi^{old}), \\
q_{ikeg} =& p(z_{ie} = f_{eg}|L, z_i = c_k, a^{old}, \phi^{old}).
\end{aligned}$$

We use the old parameters a^{old} and ϕ^{old} to update posterior probability of E step. Then we use results of E step to update new a and ϕ by

$$(a^*, \phi^*) = \arg\max_{(a,\phi)} Q(a^{old}, \phi^{old}, a, \phi). \tag{10}$$

The problem in Eq. 10 is an optimization problem and we use *Lagrange multiplier* method to solve it with the constraint condition $\sum_g q_{ikeg} = 1$. Then we can get the following results:

$$a_j = \ln(K-1) - \ln\left(\frac{\sum_i \sum_k \beta_{ik} + K * \alpha}{\sum_i \sum_k \beta_{ik} I(l_{ij}^z, c_k) + \alpha} - 1\right) \tag{11}$$

$$a_{je} = \ln(K_e - 1) - \ln\left(\frac{\sum_i \sum_k \beta_{ik} \sum_g q_{ikeg} + K_e * \alpha}{\sum_i \sum_k \beta_{ik} \sum_g q_{ikeg} I(l_{ij}^e, f_{eg}) + \alpha} - 1\right) \tag{12}$$

$$\phi_{kf_{eg}}^e = \frac{\sum_g \sum_i \beta_{ik} q_{ikeg}}{\sum_g \sum_i \beta_{ik}} \tag{13}$$

Algorithm 1: Algorithm 1 EM algorithm for GLAF Model

Input: Label matrix$L = \{l_{ij}^z, l_{ij}^e | i \in I_T, j \in I_W\}$
Output: aggregation labels $Z = \{z_i | i \in I_T\}$

Initialization:
worker j's ability parameter $a_j = 1, a_{je} = 1$
for *n=0 to iterations* **do**
  if *ability errors* $<$ *tolerance* **then**
    break;
  E step:
  compute $p(z_i|L, a, \phi)$, $p(z_{ie}|L, z_i, a, \phi)$
  M step:
  update a, ϕ by $\underset{(a,\phi)}{\operatorname{argmax}} E[\ln p(L, Z|a, \phi)]$
return $z_i = \underset{z_i}{\operatorname{argmax}}\, p(z_i|L, a, \phi)$

where $I(\cdot)$ in Eq. 11 and Eq. 12 is an indicator function and we used *additive smoothing* to solve a_j and a_{je} to prevent a_j or a_{je} become infinity. For simplicity, we set the priori $\alpha = 0.1$.

The derivation detail is omitted due to its complexity and numerous formula derivations. We will provide the details and the code upon the publication of this paper. The EM algorithm is summarized in Algorithm 1.

5 Experiments

In this section, we conduct experiments to evaluate our proposed GLAF model. We first describe three simplified versions of our model and the used baselines. Next, we introduce how to generate simulated data and show the impact of parameters and data scale respectively. We describe our implementation of a real crowdsourcing process in AMT, then discuss the results.

5.1 Experiments Setup

We study the effectiveness of our GLAF model through a series of experiments on one synthetic dataset and one real dataset with features labels. The following nine models are used for experimental comparison:

- **MV** directly uses majority votes to integrate annotations without feature labels or modeling of tasks and workers [26].
- **DS** uses one confusion matrix to model the ability of each worker and infers the true answers of tasks using EM algorithm [2].
- **HDS** simplifies the assumption made by DS to consider a confusion matrix with only a single parameter, in which each worker is assumed to have the same accuracy on each class of task, and have the same error probabilities as well [13,24].

- **FDS** is a simple, yet effective, EM-based algorithm, which can be interpreted as a 'hard' version of DS model that allows much faster convergence while maintaining similar accuracy in aggregation [27].
- **GLAD** is a probabilistic model that simultaneously infers the answer, difficult level of each task, as well as the ability of each worker [33].
- **GLA** uses one parameter to model the ability of each worker and infers the true answers of tasks using EM algorithm, which is the first simplified versions of our method.
- **GLA(f1)** incorporates the feature $f1$ to infer true labels based on the GLA.
- **GLA(f2)** incorporates the feature $f2$ to infer true labels based on the GLA.
- **GLAF** models worker's abilities to label categories and two features ($f1$ and $f2$), and use the relationship between objects and features as parameters to infer truth labels, which is our proposed method in this paper.

The first six methods build models without using feature labels and the relationship between object labels and features. The other three use feature labels and the relationship between object labels and features. For GLAF and its variants, we initially simply let the relationship be uniformly distributed. For the other methods, we set the parameters (if any) following the suggestions of the authors.

5.2 Simulations

Data Preparation. We simulated 100 workers and 1000 tasks. The true answer of each task belongs to the set Z ($Z = \{0, 1, 2, 3\}$) with equal probability. Each task has two features and the value of each feature is 0 or 1. We let the probability of workers answering the task category correctly obey the uniform distribution from $\frac{1}{K}$ to 1 and the probability of workers answering the task feature correctly obey the uniform distribution from $\frac{1}{K_e}$ to 1, where K is the number of categories and K_e is the number of feature e. Then the observed variable l_{ij}^z can be sampled according to Eq. 1 given a worker's ability to label task categories. In the same way, the observed variable l_{ij}^e can be sampled according to Eq. 2 given the ability of a worker to label task features. And the aforementioned mapping relationship between categories and features can be expressed in the probability form of Eq. 3. We ask each task to be answered 10 times. In this way, we obtained 10000×3 labels (1 category and 2 features).

Results. We run each model with different redundancy to compute the consensus annotation and report the accuracy in Fig. 3(a). As expected, the accuracy of all models increase as the redundancy of each task grows. As we can see, our proposed GLAF model that uses both two feature labels achieved the best performance among all models and is also superior to its degenerated versions GLA(f1) and GLA(f2), which use only one feature label. GLA(f1) and GLA(f2) perform better than GLA, which is a commonly used baseline aggregation algorithm without feature labels. DS, HDS, FDS and GLAD come next with regard to accuracy. Among all these models, MV performs the worst, this could due to

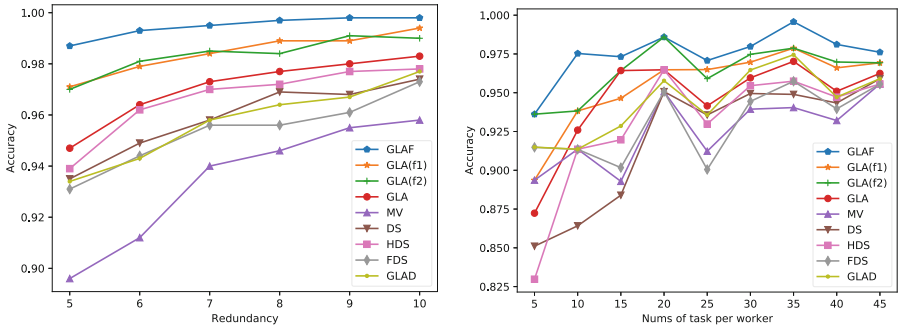


Fig. 3. (a)The accuracies of GLAF model versus other methods for inferring the underlying category labels on simulation data. GLA(s) only uses the features s. (b) The accuracies of GlAF model and other methods vs Numbers of task answered by workers. GLA(s) only uses the features.

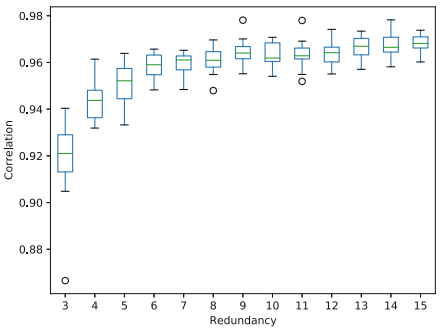


Fig. 4. The ability of GLAF to recover the true a_j parameter on simulation data.

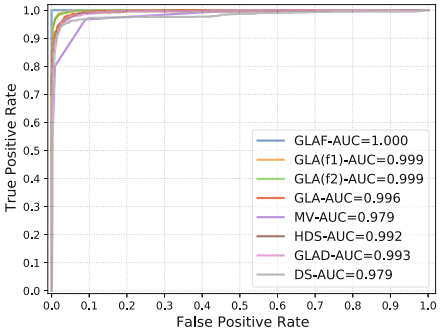


Fig. 5. ROC comparison of GLAF and other models.

the fact that MV directly uses majority vote to integrate annotations without modeling workers ability. Figure 3(b) shows the accuracy when we change the number of tasks performed by worker, again the conclusion is basically the same as in Fig. 3(a), especially when the number of tasks is larger than 30. Overall, the results show that incorporating more features and its relations could significantly improve the model performance.

Correlation. In addition, we change the number of worker from 3 to 15 to see how this impact the learnt parameters. In this simulation, each worker handles 1000 tasks. The truth value of each task belongs to the set Z (Z = 0, 1, 2, 3) with equal probability. The accuracy a_j of each worker was drawn from a uniform distribution. Given these worker abilities, the observed labels l_{ij}^z were sampled

according to Eq. 1 using Z. Finally, the EM inference procedure described above was executed to estimate a_j. This procedure was repeated 20 times to smooth out variability between trials. On each trial we computed the correlation between the parameter estimates $\hat{a_j}$ and the true parameter value a_j. The results (averaged over 20 experimental runs) are shown in Fig. 4. As expected, as the number of workers grows, the parameter estimates converge to the true values.

ROC Curves. We performed experiments on simulation dataset to draw roc curves. We can find similar trends as shown in Fig. 3(a). Figure 5 shows the ROC comparisons and AUC values for GLAF model and baseline models. FDS does not output probability distribution data like other methods, so its ROC curve is not shown here. The experimental results demonstrate that our approaches significantly outperform baseline methods without feature data.

5.3 Real-Data

In this section, we first describe the real dataset used and then report the evaluation results of the proposed GLAF model.

Data Preparation. We crawled images of different dogs from *American Kennel Club*[2] and filtered out the images with dead URLs or images with no dogs in it. We finally obtained 820 unambiguous images for experiments.

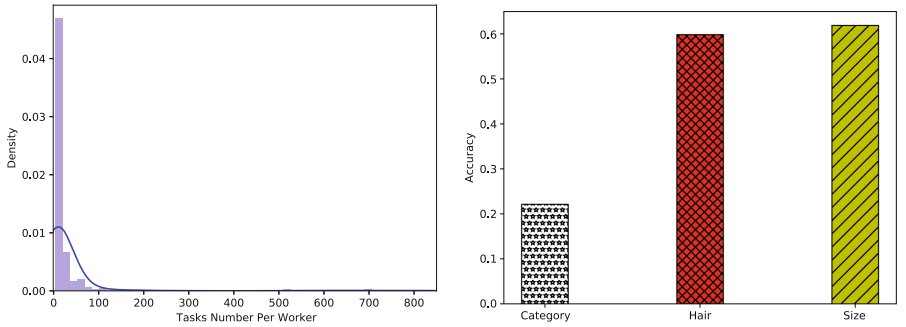
We used *Amazon Mechanical Turk* and followed the workflow shown in Sect. 3.2. We guarantee the quality of labels by employing high-quality workers in the platform. We gave a brief instruction to guide workers to provide as specific labels as possible. Each task is sent to 20 different workers. For each task, workers were asked to choose the label they gave to the image from a drop-down box and complete two multiple choice questions about the features (hair and size). Hair length includes long hair, short hair and no hair and body sizes include large and small. We collected $820 \times 20 \times 3$ labels (1 category label and 2 feature labels).

After removing the invalid labels, there left 13261 labels annotated by 412 workers and the number of unique labels is 59, which is considerably large compared to other crowdsourcing tagging tasks. For evaluation, we use the groundtruth labels provided by *American Kennel Club*.

Results. The probability distribution of the number of tasks completed per worker is shown in Fig. 6(a). As we can see, most workers completed a limited number of tasks and only a small number of workers had completed more than 100 labeling tasks. Such data sparsity is harmful to aggregation methods with unknown parameters. We count the accuracy of category labels and feature labels given by each worker in the original data set as shown in Fig. 6(b). This agrees

[2] https://www.akc.org/.

with our hypothesis that worker is better at labeling obvious features than the entity's category. The two feature labels obtain 180% (hair) and 172% (size) higher accuracy than category on average.



(a) (b)

Fig. 6. (a)Distribution of the number of tasks answered by workers about original data (b) Accuracies of category and features without any methods processing about original data.

Figure 7(a) shows the accuracy of models when we change workers redundancy level. As expected, more workers brought higher accuracy for all models. GLAF, GLA(f1) and GLA(f1) outperform the GLA, MV, DS, HDS, GLAD and FDS in category labeling. The GLAF model performs the best, which implies that we can improve the quality of category labels via incorporating feature labels into *answer aggregation.* Figure 7(b) shows the accuracy of category labels and features labels obtained from MV, DS, HDS, FDS, GLAD and GLAF. The accuracy of category labels obtained by GLAF is higher than that of MV, DS, HDS, FDS and GLAD. The accuracy of two features labels is lower than category labels in GLAF,FDS and GLAD, and the opposite result is observed in MV, DS and HDS. But it is acceptable for us to improve the quality of category labels at the expense of the quality of feature labels.

ROC Curves. We performed experiments on real dataset to draw roc curves and the results are shown in Fig. 8. FDS does not output probability distribution like other models, so its ROC curve is not shown. We can see that our proposed model significantly outperforms the GLA, GLA(f1), GLA(f2), MV, DS, HDS and GLAD. GLAD and MV follow GLAF and its variants. DS is the worst. The best GLAF improves 20% than the worst DS. And GLAF performs significantly better than GLA(f1) and GLA(f2), which proves our conclusion that features information of tasks is very important for inferring the final result.

Runtime. We further investigate the time consumption of GLAF and compare it with other models. All algorithms are implemented in python. We tested them

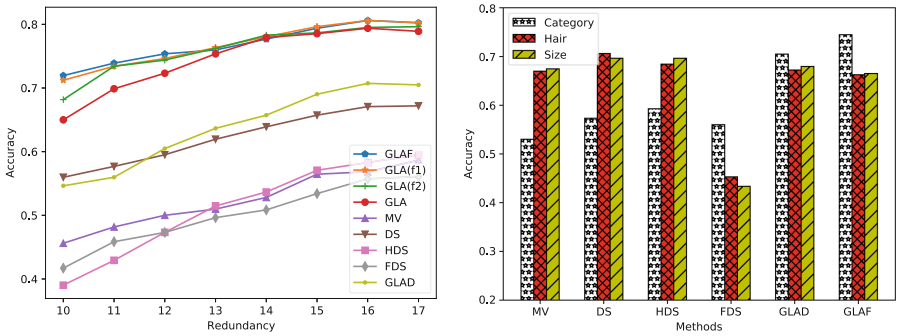


Fig. 7. (a)The accuracies of GLAF model and other methods vs Number of workers for the same task on real data. (b) Accuracy of category and features of GLAF model and other methods on real data.

in a workstation with 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40 GHz ,eight-core CPU, 16 GB RAM and 64 bit Windows 10 OS. The running time of GLAF includes processing category answers, feature answers and parameters estimation by EM algorithm and MLE time. Figure 9 draws their running time in seconds on real datasets.

The runtime results are as follows:MV(0.02 s), DS(1s), HDS(48s), FDS(18s), GLAD(3215 s)s), GLA(310s), GLA(f1)(522s), GLA(f2)(403s), and GLAF(354s). Figure 9 shows that MV undoubtedly has the minimum time cost while GLAD has the maximum time cost. DS, HDS and FDS has the equal time cost level. GLAF,GLA,GLA(f1) and GLA(f2) have the equal time cost level. And DS, HDS and FDS have less time cost than GLAF and its variants. Our proposed model GLAF has more time cost than DS and its variants, but we have already proved that our proposed model has better performance in accuracy and ROC curves. So we think it is worthwhile to spend about 300 s more for higher accuracy.

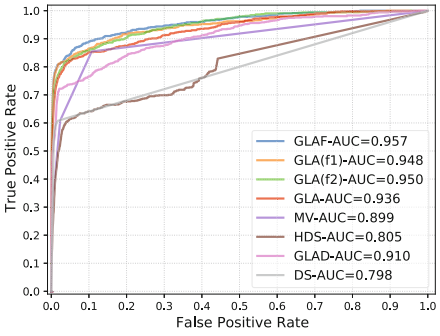


Fig. 8. ROC comparison of GLAF and other models.

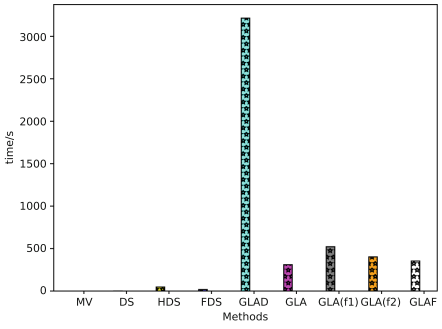


Fig. 9. Runtime comparison of GLAF and other models.

6 Conclusions

In this paper, we propose a novel generative model of labels, abilities and features (GLAF) that can take advantage of feature labels and its relationship with object categories to infer the true answer of one task. In the proposed probabilistic model, it automatically learns the worker ability and the relationship to customize the algorithm to fit the data by using Expectation Maximization (EM) algorithm. Experiment results showed that workers are better at labeling features than labeling one entity's categories and incorporating these feature labels and their relationship could significantly improve the model performance. Our model also gives state of the art results compared with most voting-like methods without features labels. Our future work will investigate techniques which jointly model workers' ability to process category labels and features labels, combined with the relationship between feature labels and category labels. We also want to design a form of shared parameters for inference learning to improve model performance.

Acknowledgments. The authors would like to thank anonymous reviewers for their insightful and constructive comments and suggestions that have helped improve the quality of this paper. This research has been supported by the National Nature Foundation of China under grant 61976187 and the Natural Science Foundation of Zhejiang Province under grant (LZ22F020008, LQ22F020002).

References

1. Bi, W., Wang, L., Kwok, J.T., Tu, Z.: Learning to predict from crowdsourced data. In: UAI, vol. 14, pp. 82–91 (2014)
2. Dawid, A.P., Skene, A.M.: Maximum likelihood estimation of observer error-rates using the EM algorithm. J. Roy. Stat. Soc.: Ser. C (Appl. Stat.) **28**(1), 20–28 (1979)
3. 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, pp. 469–478 (2012)
4. Feng, W., Yan, Z., Zhang, H., Zeng, K., Xiao, Y., Hou, Y.T.: A survey on security, privacy, and trust in mobile crowdsourcing. IEEE Internet Things J. **5**(4), 2971–2992 (2017)
5. Han, G., Tu, J., Yu, G., Wang, J., Domeniconi, C.: Crowdsourcing with multiple-source knowledge transfer. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 2908–2914 (2021)
6. Han, T., Sun, H., Song, Y., Wang, Z., Liu, X.: Budgeted task scheduling for crowdsourced knowledge acquisition. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1059–1068 (2017)
7. Ho, C.J., Vaughan, J.: Online task assignment in crowdsourcing markets. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 26, pp. 45–51 (2012)
8. Hoßfeld, T., et al.: Best practices for QoE crowdtesting: QoE assessment with crowdsourcing. IEEE Trans. Multimedia **16**(2), 541–558 (2013)

9. Howe, J.: The rise of crowdsourcing. Wired Mag. **14**(6), 1–4 (2006)
10. Hwang, K., Lee, S.Y.: Environmental audio scene and activity recognition through mobile-based crowdsourcing. IEEE Trans. Consum. Electron. **58**(2), 700–705 (2012)
11. Jiang, L., Zhang, H., Tao, F., Li, C.: Learning from crowds with multiple noisy label distribution propagation. In: IEEE Transactions on Neural Networks and Learning Systems (2021)
12. 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)
13. Karger, D., Oh, S., Shah, D.: Iterative learning for reliable crowdsourcing systems. In: Advances in Neural Information Processing Systems 24 (2011)
14. Kurve, A., Miller, D.J., Kesidis, G.: Multicategory crowdsourcing accounting for variable task difficulty, worker skill, and worker intention. IEEE Trans. Knowl. Data Eng. **27**(3), 794–809 (2014)
15. Li, H., Yu, B.: Error rate bounds and iterative weighted majority voting for crowdsourcing. arXiv preprint arXiv:1411.4086 (2014)
16. Li, J., Endo, L.R., Kashima, H.: Label aggregation for crowdsourced triplet similarity comparisons. In: Mantoro, T., Lee, M., Ayu, M.A., Wong, K.W., Hidayanto, A.N. (eds.) ICONIP 2021. CCIS, vol. 1517, pp. 176–185. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92310-5_21
17. Li, Y., Kou, N.M., Wang, H., U, L.H., Gong, Z.: A confidence-aware top-k query processing toolkit on crowdsourcing. Proceed. VLDB Endow. **10**(12), 1909–1912 (2017)
18. Liu, J., Tang, F., Chen, L., Zhu, Y.: Exploiting predicted answer in label aggregation to make better use of the crowd wisdom. Inf. Sci. **574**, 66–83 (2021)
19. Liu, X., Lu, M., Ooi, B.C., Shen, Y., Wu, S., Zhang, M.: CDAS: a crowdsourcing data analytics system. arXiv preprint arXiv:1207.0143 (2012)
20. Ma, Y., Sun, Y., Lei, Y., Qin, N., Lu, J.: A survey of blockchain technology on security, privacy, and trust in crowdsourcing services. World Wide Web **23**(1), 393–419 (2020)
21. Mao, K., Capra, L., Harman, M., Jia, Y.: A survey of the use of crowdsourcing in software engineering. J. Syst. Softw. **126**, 57–84 (2017)
22. Oyama, S., Baba, Y., Sakurai, Y., Kashima, H.: Accurate integration of crowdsourced labels using workers' self-reported confidence scores. In: Twenty-Third International Joint Conference on Artificial Intelligence (2013)
23. Rahman, H., Roy, S.B., Thirumuruganathan, S., Amer-Yahia, S., Das, G.: Task assignment optimization in collaborative crowdsourcing. In: 2015 IEEE International Conference on Data Mining, pp. 949–954. IEEE (2015)
24. Raykar, V.C., et al.: Learning from crowds. J. Mach. Learn. Res. **11**(4), 1297–1322 (2010)
25. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: LabelMe: a database and web-based tool for image annotation. Int. J. Comput. Vision **77**(1–3), 157–173 (2008). https://doi.org/10.1007/s11263-007-0090-8
26. 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, pp. 614–622 (2008)
27. Sinha, V.B., Rao, S., Balasubramanian, V.N.: Fast Dawid-Skene: a fast vote aggregation scheme for sentiment classification. arXiv preprint arXiv:1803.02781 (2018)

28. Smyth, P., Fayyad, U., Burl, M., Perona, P., Baldi, P.: Inferring ground truth from subjective labelling of venus images. In: Advances in Neural Information Processing Systems 7 (1996)
29. Venanzi, M., Guiver, J., Kazai, G., Kohli, P., Shokouhi, M.: Community-based bayesian aggregation models for crowdsourcing. In: Proceedings of the 23rd international conference on World wide web, pp. 155–164 (2014)
30. Vondrick, C., Patterson, D., Ramanan, D.: Efficiently scaling up crowdsourced video annotation. Int. J. Comput. Vision **101**(1), 184–204 (2013)
31. Wang, J., Li, G., Kraska, T., Franklin, M.J., Feng, J.: Leveraging transitive relations for crowdsourced joins. In: SIGMOD, pp. 229–240. ACM (2013)
32. Welinder, P., Branson, S., Perona, P., Belongie, S.: The multidimensional wisdom of crowds. In: Advances in Neural Information Processing Systems 23 (2010)
33. Whitehill, J., Wu, T.f., Bergsma, J., Movellan, J., Ruvolo, P.: Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In: Advances in Neural Information Processing Systems 22, pp. 2035–2043 (2009)
34. Zaidan, O., 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, pp. 1220–1229 (2011)
35. Zhang, J., Sheng, V.S., Wu, J., Wu, X.: Multi-class ground truth inference in crowdsourcing with clustering. IEEE Trans. Knowl. Data Eng. **28**(4), 1080–1085 (2015)
36. Zhang, Y., Chen, X., Zhou, D., Jordan, M.I.: Spectral methods meet EM: a provably optimal algorithm for crowdsourcing. In: Advances in Neural Information Processing Systems 27 (2014)
37. Zhou, D., Basu, S., Mao, Y., Platt, J.: Learning from the wisdom of crowds by minimax entropy. In: Advances in Neural Information Processing Systems 25 (2012)