OpenCrowd: A Human-AI Collaborative Approach for Finding Social Influencers via Open-Ended Answers Aggregation

Ines Arous
University of Fribourg
Fribourg, Switzerland
ines.arous@unifr.ch

Jie Yang*
Amazon
Seattle, USA
jie@exascale.info

Mourad Khayati
University of Fribourg
Fribourg, Switzerland
mourad.khayati@unifr.ch

Philippe Cudré-Mauroux
University of Fribourg
Fribourg, Switzerland
pcm@unifr.ch

ABSTRACT

Finding social influencers is a fundamental task in many online applications ranging from brand marketing to opinion mining. Existing methods heavily rely on the availability of expert labels, whose collection is usually a laborious process even for domain experts. Using open-ended questions, crowdsourcing provides a cost-effective way to find a large number of social influencers in a short time. Individual crowd workers, however, only possess fragmented knowledge that is often of low quality.

To tackle those issues, we present OpenCrowd, a unified Bayesian framework that seamlessly incorporates machine learning and crowdsourcing for effectively finding social influencers. To infer a set of influencers, OpenCrowd bootstraps the learning process using a small number of expert labels and then jointly learns a feature-based answer quality model and the reliability of the workers. Model parameters and worker reliability are updated iteratively, allowing their learning processes to benefit from each other until an agreement on the quality of the answers is reached. We derive a *principled* optimization algorithm based on variational inference with efficient updating rules for learning OpenCrowd parameters. Experimental results on finding social influencers in different domains show that our approach substantially improves the state of the art by 11.5% AUC. Moreover, we empirically show that our approach is particularly useful in finding micro-influencers, who are very directly engaged with smaller audiences.

CCS CONCEPTS

• **Information systems → Crowdsourcing**; • **Human-centered computing → Social network analysis**; • **Mathematics of computing → Bayesian computation**; • **Computing methodologies → Neural networks**; **Learning latent representations**.

KEYWORDS

Influencer finding; Human-AI Collaboration; Variational Inference

ACM Reference Format:
Ines Arous, Jie Yang, Mourad Khayati, and Philippe Cudré-Mauroux. 2020. OpenCrowd: A Human-AI Collaborative Approach for Finding Social Influencers via Open-Ended Answers Aggregation. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3366423.3380254

1 INTRODUCTION

Social influence is an important mechanism impacting the dynamics of social networks. Social influencers are users who regularly produce authoritative or novel content on specific topics and who can reach and engage a potentially large group of followers. Finding social influencers has become a fundamental task in many online applications, ranging from brand marketing [34, 43] to opinion mining [20, 28], expert finding for question answering [32], minimizing misinformation propagation [37], or analyzing presidential elections [6].

The task of finding social influencers is challenging due to the complexity in quantifying user engagement, the subjectivity in perceiving social influence, and the need for expert knowledge in determining the authenticity of user-generated content. Existing techniques mainly tackle this problem using supervised machine learning approaches that rely on a training set hand-labeled by domain experts [9, 19, 46]. While models trained in this fashion are effective at finding social influencers who are similar to those in the training data, they are intrinsically limited by the availability of expert labels. These labels are typically very hard to gather. As an example, our collaboration with the largest European fashion retailer[1] reveals that an expert can only recognize no more than 200 fashion influencers on Twitter over a 3-week period of time. Finding social influencers is, therefore, a long and usually laborious process even for domain experts [16].

Compared to an individual expert, online crowds possess *as a whole* a broader knowledge of social influencers in several domains, e.g., fashion, fitness, or information technology. As an example, while it is generally difficult for an expert to come up with a long list of fashion influencers in a short period of time, it is much easier to obtain such a list by asking online workers. Therefore,

*Corresponding author; work performed before joining Amazon.

[1] Zalando SE: https://research.zalando.com/

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380254

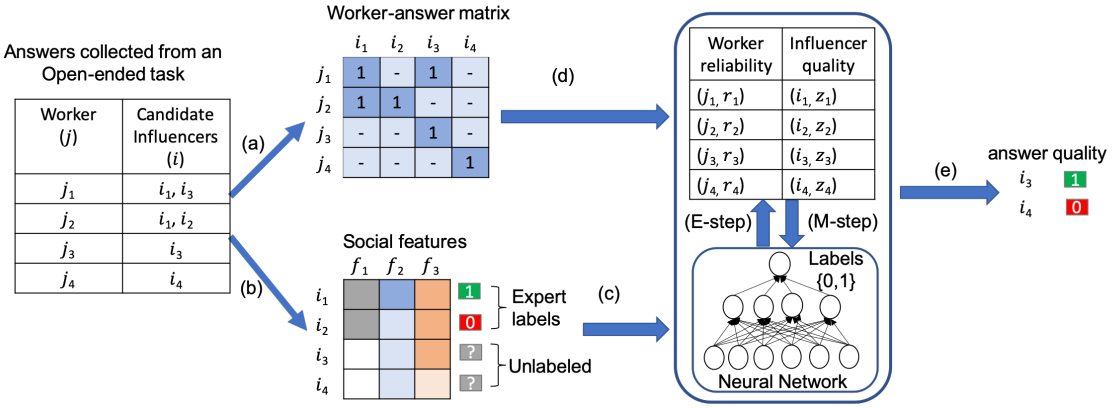


Figure 1: The OpenCrowd framework (a) creates a worker-answer matrix from workers answers; (b) extracts social features for the Candidate Influencers ($i_1, \ldots, i_4$); (c) uses the social features and the expert labels to train an answer quality model and estimate unknown labels; (d) (E-step) uses both the worker-answer matrix and the labels generated by the answer quality model to estimate the worker's reliability and the candidate influencer's labels; (M-step) uses the new candidate influencer's labels (influencer quality) to retrain the answer quality model; and (e) generates labels for the unlabeled candidate influencers.

we advocate a human computation approach that crowdsources the task of finding social influencers in the form of open-ended question-answering, a popular and crucially important, yet severely understudied class of crowdsourcing [29]. Specifically, we consider a task where the crowd is asked to name as many social influencers as possible in a predefined domain. By aggregating the answers from a large number of crowd workers, we can identify the identities (e.g., usernames on Twitter) of a large number of social influencers in an efficient and cost-effective manner.

Despite its obvious benefits, aggregating answers from open-ended crowdsourcing campaigns is challenging: individual crowd workers may only possess fragmented knowledge that is of low-quality. Unlike Boolean crowdsourcing, where crowd workers are asked to classify an existing close pool of data instances into *predefined classes*, open-ended crowdsourcing results in open-ended pools of answers – often of large size – that were *all deemed relevant* by crowd workers. The input data for open-ended answers aggregation is, therefore, a positive-only worker-answer matrix, where each entry indicates the "given by" relationship between an answer and a certain worker, as illustrated in Figure 1. This comes in contrast to the input data for aggregating answers from Boolean crowdsourcing, where each entry indicates a class (e.g., 0 or 1 for the binary case) assigned by a worker to a data instance. As an implication, existing answers aggregation methods [10, 47, 48, 53], which are designed to leverage the disagreement between workers' answers, do not yield good performance for open-ended answers aggregation (cf. Section 5).

To address the problem of open-ended answers aggregation, we introduce a human-AI collaborative approach that integrates both machine learning and crowdsourcing for aggregating open-ended answers. We present OpenCrowd, a Bayesian framework that models the true label of a candidate influencer as dependent on both the features of the candidate and the reliability of the workers who named the candidate. To infer the truth, OpenCrowd leverages a small number of expert labels to bootstrap the inference process. It then jointly learns a feature-based model for the quality of the answers and the reliability of the crowd workers. The model parameters and worker reliability are updated in an iterative manner, allowing their learning processes to benefit from each other until an agreement on answer quality is reached. The overall learning process is illustrated in Figure 1. We formalize such a learning process with a principled optimization algorithm based on variational expectation-maximization. In particular, we derive updating rules that allow both model parameters and worker reliability to be updated incrementally at each new iteration. By doing so, OpenCrowd parameters can be efficiently learned with little extra computational cost compared to the computational cost for training a feature-based answer quality model.

To the best of our knowledge, we are the first to adopt a human-AI collaborative approach for finding social influencers. Our proposed framework is a generic one that can incorporate any machine learning models with crowdsourcing. Moreover, as it solicits contribution directly from crowd workers, the framework is effective in finding a particular type of influencers known as "micro-influencers" [2, 49]. These social influencers are deeply connected to specific niche audiences, thus are able to effectively deliver messages to a highly relevant audience. Unlike macro-influencers who have a huge number of followers (e.g., millions), micro-influencers often have relatively fewer followers, yet they enjoy a more trustworthy reputation (e.g., higher conversion rate in product promotion) and direct relationship with them.

In summary, we make the following key contributions:

- We propose OpenCrowd, a Bayesian framework for finding social influencers through open-ended answers aggregation;
- We derive an efficient learning algorithm based on variational inference with incremental updating rules for OpenCrowd parameter estimation;
- We conduct an extensive evaluation on two domains – fashion and information technology – and show that OpenCrowd substantially improves the state of the art by 11.5% AUC.

2 RELATED WORK

In this section, we first review existing answers aggregation methods applicable for finding social influencers. Then, we discuss metric and feature-based techniques proposed to solve this problem.

2.1 Answers Aggregation

Influence is defined as "the power of producing an effect on the character or behavior of someone" (Oxford Dictionary). This concept is intrinsically difficult to quantify especially in a large scale context. Answers aggregation provides an efficient and cost-effective manner to identify a large number of social influencers. Methods have been mainly developed in Boolean crowdsourcing. Typical methods include majority voting [35] and those based on expectation-maximization (EM), which simultaneously estimate the true labels and parameters related to the annotation process such as worker reliability and task difficulty [53]. Dawid and Skene [10] make a seminal contribution by proposing to model the worker's reliability with a confusion matrix for answers aggregation. Demartini et al. [11] address a similar problem while modeling worker reliability as a scalar parameter, which can be less expressive but more robust for highly sparse worker-answer matrices – as we discuss in our experiments. Whitehill et al. [48] introduce a similar method yet further propose to model task difficulty in addition to worker reliability. Closer to our method is LFC proposed by Raykar et al. [31], which models worker reliability as a latent variable with a prior distribution, thus capable of quantifying the uncertainty of the inference. Unlike these techniques, our proposed framework further incorporates existing labels and social features, thus extending the applicability of answers aggregation to open-ended tasks.

While little work has focused on open-ended answers aggregation [29], some techniques consider features of answers or tasks for answers aggregation. A seminal work by Welinder et al. [47] considers the implicit dimensions of worker expertise and task domains and propose a probabilistic model where such dimensions are modeled as feature vectors of tasks to be learned from the worker-answer matrix. A similar line of work takes advantage of explicit task features to learn such dimensions [12, 22, 52]. Ma et al. [22] propose a joint model that captures the task domain from textual descriptions and worker reliability from the worker-answer matrix. Zheng et al. [52] further consider external knowledge bases to better capture task domains. Fan et al. introduce iCrowd [12], which measures the topical similarity of tasks by employing topic modeling techniques (e.g., LDA [5]), and leverages such similarity for a better estimate of worker reliability. A similar idea is investigated by Lakkaraju et al. [18], which also considers similarity among workers based on their features. All these works, however, rely on unsupervised topic models or ad-hoc modeling of specific features to help estimate worker reliability. Our proposed framework is different in that it incorporates crowdsourcing and supervised models that can consume any features.

Human-AI Collaboration. Our work is related to the emerging field of human-AI collaboration paradigm arising from the intersection between human computation and machine learning [44]. Human computation has been used to enhance machine learning systems by generating the data [50, 53] *before* model training, or providing interpretations for model decisions [33] and debugging the system or the data [26, 51] *after* model training. Typically, human computation and machine learning are treated as disentangled processes. Our approach provides a way to deeply integrate human and machine intelligence in a Bayesian framework, where human characteristics (i.e., reliability) and model parameters are iteratively inferred in a mutually boosting manner until the decisions from aggregated human answers and those from the model agree with each other. Our work can be seen as a development of the "learning-from-crowds" line of research [31, 41, 50], which considers the machine learning problem in the context of noisy labels contributed by the crowd. Unlike existing work, our framework is generic in that it does not assume any type of machine learning models, thus is applicable to a wider range of problems and application domains.

2.2 Social Influencer Finding

Existing methods for social influencer finding can be categorized into two classes: metric and feature-based. Common metrics for identifying social influencers include the number of followers, the number of mentions, and the ratio of the number of comments/likes to the number of followers [15, 17]. These metrics are often insufficient to fully capture the degree of influence because of the difficulty to measure content authenticity and engagement with audience. The latter dimension, i.e., engagement, has become a key consideration along with the shift of focus in industry from finding macro-influencers (including celebrities) to micro-influencers [2, 49].

An alternative approach to finding social influencer is machine learning, which can detect influencers by weighting a large number of social features. Existing work has considered a variety of social features including metadata features such as the number of followers and followees [9, 19], the number of retweets and mentions [8], semantic features such as the topics of a candidate influencer's microposts [32, 46], or features derived from user behavioral data such as the activeness of a candidate influencer in online activities [1, 19]. In addition to those, several pieces of work consider a specific type of feature, namely the structure of the social network among the influencers and other online users, to improve the accuracy in finding social influencers [13, 21, 27, 30, 39, 40]. For instance, Tang et al. [39, 40] propose to find the influencers as the nodes from which the spread of information is maximized. Qiu et al. [30] adopt a deep learning framework where the network embedding and some user-specifc features are fed into a deep neural network for predicting social influencers. Bi et al. [3] introduce a model to incorporate the content of tweets and the followee distribution of microblogs. Similarly, Pal et al. [27] proposed a probabilistic clustering method to produce a ranked list of influencers using node degree, information diffusion, and metrics related to tweets' content.

A less discussed aspect in the machine learning approach is training example creation, which is generally performed by experts through manual screening. The process involves careful examination of content quality, feed consistency, and estimation of the rate of high-quality interactions with the audience. Such a process does not scale for a large number of influencers. Unlike existing methods, our proposed framework only requires a small number of expert labels and shifts the burden of label creation to online workers through crowdsourcing, which is fast, scalable, and cost-effective.

Table 1: Notations.

Notation	Description
$\mathcal{I}$	Set of candidate social influencers
$\mathcal{I}_{\mathcal{L}}$	Set of labeled candidate social influencers
$\mathcal{I}_j$	Set of candidate influencers relevant to a worker j
$\mathcal{J}$	Set of workers
$\mathcal{J}_i$	Set of workers relevant to a candidate influencer i
$\mathcal{W}_I$	Set of parameters for the answer quality model
$\mathbf{A}$	Worker-answer matrix
$\mathbf{x}_i$	Social feature vector of a candidate influencer
z_i	Influencer quality distribution
r_j	Worker reliability distribution
θ_i	Parameter of the influencer quality distribution
A, B	Parameters of the prior distribution of worker reliability
α, β	Variational parameters of the worker reliability distribution

3 PROBLEM FORMULATION

In this section, we first introduce the notations used in the paper and then formally define our problem.

Notations. We use boldface lowercase letters to denote vectors and boldface uppercase letters to denote matrices. For an arbitrary matrix $\mathbf{M}$, we use $\mathbf{M}_{i,j}$ to denote the entry at the i-th row and j-th column. We use capital letters (e.g., $\mathcal{P}$) in calligraphic math font to denote sets. We use $x \propto y$ to denote that the two variables x and y are proportionally related, i.e., $x = ky$, where k is a constant.

Table 1 summarizes the notations used throughout this paper. We denote the set of unique *candidate* social influencers named by the crowd workers as $\mathcal{I}$ and the set of workers as $\mathcal{J}$. We use $\mathbf{A}_{i,j} = 1$ to denote that the candidate influencer i is an answer provided by worker j, and $\mathbf{A}_{i,j} = 0$ otherwise. Due to the fact that an individual worker can only provide a limited number of candidate influencers, $\mathbf{A}_{i,j}$ is a sparse matrix where only a small proportion of the entries are non-zero. For each candidate influencer $i \in \mathcal{I}$, we collect her social features as described in detail in Section 5.1, and denote the resulting feature vector by $\mathbf{x}_i$. The subset of $\mathcal{I}$, denoted as $\mathcal{I}_{\mathcal{L}} \subset \mathcal{I}$, represents candidate influencers who are associated with expert labels y_i (for $i \in \mathcal{I}_{\mathcal{L}}$). Note that we only have a relatively small number of candidate influencers with expert labels, namely, $|\mathcal{I}_{\mathcal{L}}| \ll |\mathcal{I}|$ and that we aim at estimating the true labels of the candidate influencers who are in $\mathcal{I} \setminus \mathcal{I}_{\mathcal{L}}$.

Problem Definition. Let $\mathcal{I}$ be a set of candidate social influencers and $\mathcal{I}_{\mathcal{L}}$ be the subset labeled by experts. Let also $\mathcal{J}$ be the set of workers who collectively nominated $\mathcal{I}$, where each candidate influencer can be named by a different number of workers. We aim at inferring the true labels $z_i \in \{0, 1\}$ for all candidate influencers in $\mathcal{I} \setminus \mathcal{I}_{\mathcal{L}}$.

Note that in an open-ended answers aggregation setting, we do not control the number of answers provided by each worker [29]. Hence, the number of workers relevant to different candidate influencer can vary from one to many, rendering the aggregation task highly challenging. This comes in contrast to the conventional crowdsourcing setting, where the number of workers is usually fixed for every data instance (e.g., five workers per instance), which simplifies answers aggregation that relies on worker disagreement.

4 THE OPENCROWD FRAMEWORK

OpenCrowd is a unified Bayesian framework that incorporates both supervised learning and crowdsourcing for identifying true social influencers via open-ended answers aggregation. In this section, we first describe the model and then present our variational inference algorithm for learning OpenCrowd parameters.

4.1 OpenCrowd as a Generative Model

We represent the generative process of answers as conditioned on both the true labels of the answers and the reliability of the workers. We model the true label of a candidate influencer $z_i \in \{0, 1\}$ with a Bernoulli distribution:

$$z_i \sim Ber(\theta_i), \theta_i = \sigma(f^{\mathcal{W}_I}(\mathbf{x}_i)), \tag{1}$$

where θ_i is the parameter of the distribution predicted by the social features of the candidate influencer through a feature-based answer quality model, denoted by $f(\cdot)$; $\mathcal{W}_I$ is the set of the model parameters; $\sigma(\cdot)$ is a sigmoid function. We denote $f(\cdot)$ as a generic function that can be instantiated with any supervised learning model, be it a linear model or a neural network. Note that $\mathcal{W}_I$ is shared across all candidate influencers [14], which allows us to exploit the similarity among candidate influencers.

We represent worker reliability as $r_j \in [0, 1]$ ($j \in \mathcal{J}$) where $r_j = 1$ indicates that the worker is fully reliable and $r_j = 0$ otherwise. In practice, we would like to have a measure of *confidence* in estimating the reliability of the workers providing different numbers of answers: we should be more confident in estimating the reliability of workers who provide 50 answers than those who provide 5 answers only. To quantify the confidence in our inference, we adopt a Bayesian treatment of r_j by introducing a prior, thus modeling r_j as a latent variable. Given that r_j is a continuous variable in $[0, 1]$, we choose a Beta distribution to model its prior:

$$r_j \sim Beta(A, B), \tag{2}$$

where A and B are the parameters of the distribution. The incorporation of confidence makes our framework more robust to overfitting, as we show later in Section 5.2.

We now define the likelihood of a worker j naming a candidate influencer i as the probability conditioned on the worker's reliability r_j and the true label of the candidate z_i:

$$p(\mathbf{A}_{i,j}|z_i, r_j) = r_j^{\mathbb{1}[z_i=\mathbf{A}_{i,j}]}(1 - r_j)^{\mathbb{1}[z_i\neq\mathbf{A}_{i,j}]}, \tag{3}$$

where $\mathbb{1}[\cdot]$ is an indicator function returning 1 if the statement is True and 0 otherwise. Note that Eq. (3) considers a worker to be reliable if she does not name a candidate influencer who is indeed not a real influencer. It is, however, likely that a worker does not name a candidate influencer i simply because she did not think of i. That means that we can only partly treat the non-named candidate influencers as those the worker considers as non-influencers. It is, therefore, necessary to introduce negative sampling into the inference algorithm.

Negative Sampling. Negative sampling consists in taking a random sample of candidate influencers not nominated by a worker as her answers of non-influencers. Such negative samples are useful to improve worker reliability inference, as we show in our experiment in Section 5.4. For each worker, we consider the candidate

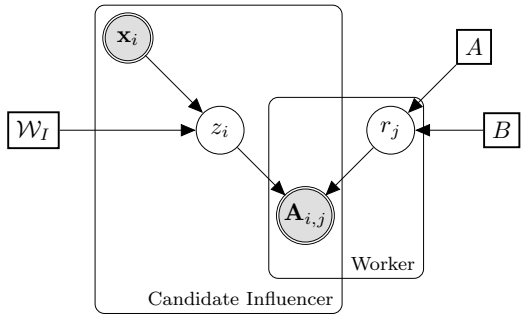


Figure 2: Graphical representation of OpenCrowd. Double (greyed) circles represent observed variables, while single circles represent latent variables. Squares represent model parameters. Edges represent conditional relationships in answer generation.

influencers named by her and the negatively sampled ones as the candidate influencers relevant to her. Similarly, to estimate the quality of a candidate influencer, we consider not only those workers who have nominated the influencer but also those whose negatively sampled answers contain such an influencer.

The overall OpenCrowd framework is depicted in Figure 2. Model learning constitutes of parameter learning for $\mathcal{W}_I$ and posterior inference for the latent variables z_i and r_j.

4.2 Variational Inference for OpenCrowd

Learning the parameters of OpenCrowd resorts to maximizing the following likelihood function:

$$p(\mathbf{A}) = \int p(\mathbf{A}, \mathbf{z}, \mathbf{r}|\mathbf{x}_i; \mathcal{W}_I)\, d\mathbf{z}, \mathbf{r}. \tag{4}$$

where $\mathbf{z}$ and $\mathbf{r}$ are the latent true labels for all candidate influencers and the reliability of all workers, respectively.

Eq. (4) consists of an integral with two latent variables, rendering it computationally unfeasible to optimize [42]. Instead, we consider the log of our likelihood function, i.e.,

$$\log p(\mathbf{A}) = \underbrace{\int q(\mathbf{z}, \mathbf{r}) \log \left(\frac{p(\mathbf{A}, \mathbf{z}, \mathbf{r}|\mathbf{x}_i; \mathcal{W}_I)}{q(\mathbf{z}, \mathbf{r})}\right) d\mathbf{z}, \mathbf{r}}_{\mathcal{L}(\mathcal{W}_I, q)} + \underbrace{\int q(\mathbf{z}, \mathbf{r}) \log \left(\frac{q(\mathbf{z}, \mathbf{r})}{p(\mathbf{z}, \mathbf{r}|\mathbf{A}, \mathbf{x}_i; \mathcal{W}_I)}\right) d\mathbf{z}, \mathbf{r}}_{KL(q||p_{\mathcal{W}_I})} \tag{5}$$

where $q(\mathbf{z}, \mathbf{r})$ is any probability density function and $KL(\cdot)$ is the KL divergence between two distributions. By doing so, the two parts of the objective function can then be optimized iteratively with a variational expectation-maximization method [42]. Specifically, we iterate between two steps: 1) the E-step where we approximate the distribution of latent variables $p(\mathbf{z}, \mathbf{r}|\mathbf{A}, \mathbf{x}_i; \mathcal{W}_I)$ with the variational distribution $q(\mathbf{z}, \mathbf{r})$, by minimizing the KL-divergence and 2) the M-step where we maximize the first term $\mathcal{L}(\mathcal{W}_I, q)$ of Eq. (5) given the newly inferred latent variables.

E-step. We use the mean-field variational inference approach [4] by assuming that $q(\mathbf{z}, \mathbf{r})$ factorizes over the latent variables:

$$q(\mathbf{z}, \mathbf{r}) = \prod_i q(z_i) \prod_j q(r_j). \tag{6}$$

We further assume the following forms for the factor functions:

$$q(z_i) = Ber(\theta_i), q(r_j) = Beta(\alpha_j, \beta_j). \tag{7}$$

where θ_i, α_j and β_j are variational parameters used to perform optimization to minimize the KL-divergence. The latter can then be minimized using coordinate ascent where we update one factor while keeping all others fixed and then iterate until convergence.

In the following, we derive the update rules for the variational distributions $q(z_i)$ and $q(r_j)$. We start by deriving the update rule for $q(z_i)$. Let $p(z_i|x_i; \mathcal{W}_I)$ be the variational distribution of z_i from the last iteration. The KL-divergence in Eq. (5) can be easily simplified [4], by keeping only the terms that depend on z_i, to the following:

$$q(z_i) \propto p(z_i|x_i; \mathcal{W}_I) \prod_{j \in \mathcal{J}_i} \exp\{g_{q(r_j)}(p(\mathbf{A}_{i,j}|z_i, r_j))\}. \tag{8}$$

where $\mathcal{J}_i$ is the set of workers relevant to a candidate influencer i and $g_x(\cdot)$ is the expectation term $\mathbb{E}_x[\log(\cdot)]$ with x being a variational distribution. Based on this equation, we show in the next lemma how to efficiently update $q(z_i)$ using the feature-based answer quality model and the worker reliability parameters from the previous iteration.

Lemma 4.1 (Incremental Answer Quality). *The true label distribution $q(z_i)$ of a candidate influencer i can be incrementally updated from the output of the answer quality model θ_i and the worker's reliability parameters α_j and β_j ($j \in \mathcal{J}_i$) in the previous iteration:*

$$q(z_i = 1) \propto \begin{cases} \theta_i \prod_{j \in \mathcal{J}_i} \exp\{\Psi(\beta_j) - \Psi(\alpha_j + \beta_j)\}, & \text{if } \mathbf{A}_{i,j} = 0, \\ \theta_i \prod_{j \in \mathcal{J}_i} \exp\{\Psi(\alpha_j) - \Psi(\alpha_j + \beta_j)\}, & \text{if } \mathbf{A}_{i,j} = 1. \end{cases} \tag{9}$$

where $\Psi(\cdot)$ is the Digamma function. If $q(z_i = 0)$ then θ_i is replaced with $(1 - \theta_i)$ and, $\Psi(\beta_j)$ and $\Psi(\alpha_j)$ are swapped.

Proof. We show the proof only for $z_i = 1$ since the proof for $z_i = 0$ follows similarly. Using Eq. (1), we have:

$$p(z_i = 1|x_i; \mathcal{W}_I) = \theta_i. \tag{10}$$

We substitute the probabilities $p(z_i|x_i; \mathcal{W}_I)$ and $p(\mathbf{A}_{i,j}|z_i, r_j)$ in Eq. (8) by their respective definitions in Eq. (10) and Eq. (3) and get:

$$q(z_i = 1) \propto \begin{cases} \theta_i \prod_{j \in \mathcal{J}_i} \exp\{g_{q(r_j)}(1 - r_j)\}, & \text{if } \mathbf{A}_{i,j} = 0, \\ \theta_i \prod_{j \in \mathcal{J}_i} \exp\{g_{q(r_j)}(r_j)\}, & \text{if } \mathbf{A}_{i,j} = 1. \end{cases} \tag{11}$$

By computing the geometric mean of the beta distribution [23], we can evaluate the expectations $g_x(.)$ as follows:

$$g_{q(r_j)}(1 - r_j) = \Psi(\beta_j) - \Psi(\alpha_j + \beta_j), \quad g_{q(r_j)}(r_j) = \Psi(\alpha_j) - \Psi(\alpha_j + \beta_j). \tag{12}$$

Putting (12) into (11), the update equation can be simplified as:

$$q(z_i = 1) \propto \begin{cases} \theta_i \prod_{j \in \mathcal{J}_i} \exp\{\Psi(\beta_j) - \Psi(\alpha_j + \beta_j)\}, & \text{if } \mathbf{A}_{i,j} = 0, \\ \theta_i \prod_{j \in \mathcal{J}_i} \exp\{\Psi(\alpha_j) - \Psi(\alpha_j + \beta_j)\}, & \text{if } \mathbf{A}_{i,j} = 1. \end{cases} \tag{13}$$

which concludes the proof. □

Next, we show how to efficiently update the variational distribution $q(r_j)$. Let $p(r_j)$ be the variational distribution of r_j from the last iteration, θ' be the true label distribution from the current iteration and $\mathcal{I}_j$ be the set of all candidate influencers relevant to a worker. The KL-divergence in Eq. (5) can be simplified, similarly to Eq. (8), by keeping only the terms that depend on r_j to get:

$$q(r_j) \propto p(r_j) \prod_{i \in \mathcal{I}_j} \exp\{g_{\theta'}(p(\mathbf{A}_{i,j}|z_i, r_j))\}. \tag{14}$$

The following lemma shows how to solve Eq. (14) using an incremental updating rule.

Lemma 4.2 (Incremental Worker Reliability). *The reliability distribution $q(r_j)$ of worker j can be incrementally updated using the reliability parameters α_j and β_j from the last iteration and the true label distribution from the current iteration, denoted as θ':*

$$q(r_j) \propto \begin{cases} Beta(\alpha_j + \sum_{i \in \mathcal{I}_j} \theta', \beta_j + \sum_{i \in \mathcal{I}_j}(1-\theta')), & if\ \mathbf{A}_{i,j} = 1, \\ Beta(\alpha_j + \sum_{i \in \mathcal{I}_j}(1-\theta'), \beta_j + \sum_{i \in \mathcal{I}_j} \theta'), & if\ \mathbf{A}_{i,j} = 0. \end{cases} \tag{15}$$

Proof. We replace the probablity $p(r_j)$ in Eq. (14) by the Beta distribution with parameters α_j and β_j from the previous iteration:

$$q(r_j) \propto Beta(\alpha_j, \beta_j) \prod_{i \in \mathcal{I}_j} \exp\{g_{\theta'}(p(\mathbf{A}_{i,j}|z_i, r_j))\}. \tag{16}$$

The expectation term in Eq. (16) can be evaluated as follows:

$$\exp\{g_{\theta'}(p(\mathbf{A}_{i,j}|z_i, r_j))\} = \begin{cases} r_j^{\theta'}(1-r_j)^{(1-\theta')}, & \text{if } \mathbf{A}_{i,j} = 1, \\ r_j^{(1-\theta')}(1-r_j)^{\theta'}, & \text{if } \mathbf{A}_{i,j} = 0. \end{cases} \tag{17}$$

In the case when $\mathbf{A}_{i,j} = 1$, we use the expressions from Eq. (17) to replace the second term in Eq. (16) as follows:

$$q(r_j) \propto Beta(\alpha_j, \beta_j) \prod_{i \in \mathcal{I}_j} r_j^{\theta'}(1-r_j)^{(1-\theta')}. \tag{18}$$

The probability density function of r_j's distribution is given by:

$$Beta(\alpha_j, \beta_j) \propto r_j^{(\alpha_j - 1)}(1-r_j)^{(\beta_j - 1)}. \tag{19}$$

Putting (19) into (18), we get:

$$\begin{aligned} q(r_j) &\propto r_j^{(\alpha_j-1)}(1-r_j)^{(\beta_j-1)} \prod_{i \in \mathcal{I}_j} r_j^{\theta'}(1-r_j)^{(1-\theta')} \\ &\propto \prod_{i \in \mathcal{I}_j} r_j^{(\alpha_j-1)}(1-r_j)^{(\beta_j-1)} r_j^{\theta'}(1-r_j)^{(1-\theta')} \\ &\propto \prod_{i \in \mathcal{I}_j} r_j^{(\alpha_j+\theta'-1)}(1-r_j)^{(\beta_j-1+(1-\theta'))} \\ &\propto r_j^{(\alpha_j+\sum_{i \in \mathcal{I}_j}\theta'-1)}(1-r_j)^{(\beta_j+\sum_{i \in \mathcal{I}_j}(1-\theta')-1)}. \end{aligned} \tag{20}$$

Thus, if $\mathbf{A}_{i,j} = 1$ we have:

$$q(r_j) \propto Beta(\alpha_j + \sum_{i \in \mathcal{I}_j} \theta', \beta_j + \sum_{i \in \mathcal{I}_j}(1-\theta')). \tag{21}$$

Algorithm 1: Coordinate Ascent Variational Inference

Input : $\mathbf{A}, \mathbf{x}_i\ (\forall i \in \mathcal{I}), y_i\ (\forall i \in \mathcal{I}_{\mathcal{L}})$
Output : Variational distributions: $q(z_i)$ and $q(r_j)$
Initialize : Variational parameters: θ_i, $\alpha_j = A$, $\beta_j = B$; parameter of the influencer predictor (f): $\mathcal{W}_I$

1 **while** *Eq. (5) has not converged* **do**
2 **for** $i \in \mathcal{I}$ **do**
3 update $q(z_i)$ using Lemma 4.1;
4 **for** $j \in \mathcal{J}$ **do**
5 update $q(r_j)$ using Lemma 4.2;
6 **for** $i \in \mathcal{I}$ **do**
7 Update $\mathcal{W}_I$ via standard gradient descent;

Following the same steps, we similarly obtain the expression of $q(r_j)$ in case $\mathbf{A}_{i,j} = 0$:

$$q(r_j) \propto Beta(\alpha_j + \sum_{i \in \mathcal{I}_j}(1-\theta'), \beta_j + \sum_{i \in \mathcal{I}_j} \theta'). \tag{22}$$

□

M-step. Given the true labels of candidate influencers and the worker reliability inferred by the E-step, the M-step maximizes the first term of Eq. (5) to learn the parameters:

$$\begin{aligned} \mathcal{L}(\mathcal{W}_I, q) &= \int q(z_i, r_j) \log p(\mathbf{A}_{i,j}, z_i, r_j|\mathbf{x}_i; \mathcal{W}_I)\, dz_i, r_j + const. \\ &= \sum_{z_i} \int q(z_i, r_j) \log[p(\mathbf{A}_{i,j}|z_i, r_j) p(z_i|\mathbf{x}_i; \mathcal{W}_I)]\, dr_j + const. \\ &= \underbrace{\sum_{z_i} \int q(z_i, r_j) \log p(\mathbf{A}_{i,j}|z_i, r_j)\, dr_j}_{\mathcal{M}_1} \\ &\quad + \underbrace{\sum_{z_i} q(z_i) \log p(z_i|\mathbf{x}_i; \mathcal{W}_I)}_{\mathcal{M}_2} + const. \end{aligned} \tag{23}$$

where $const. = \mathbb{E}_{q(z_i, r_j)} log(\frac{1}{q(z_i, r_j)})$ is a constant. Only the second part of $\mathcal{L}(\mathcal{W}_I, q)$, i.e., $\mathcal{M}_2$, depends on the model's parameters. $\mathcal{M}_2$ is exactly the inverse of the cross-entropy between $q(z_i)$ and $p(z_i|\mathbf{x}_i; \mathcal{W}_I)$, which is widely used as the loss function for many classifiers. $\mathcal{M}_2$ can, therefore, be optimized using standard methods [25] (e.g., back-propagation in the case of a neural network).

4.3 Algorithm

The overall optimization algorithm is given in Algorithm 1. It iterates over the E-step (rows 2-5) and the M-step (rows 6-7) until our objective function converges. In rows 2-3, we iterate through all candidate influencers where for each candidate influencer i, we update $q(z_i)$ using Lemma 4.1. Similarly in rows 4-5, we iterate through all workers where for each worker j we update $q(r_j)$ using Lemma 4.2. In rows 6-7, $\mathcal{W}_I$ can be incrementally updated starting from the values in the previous iteration. The convergence is reached when answer quality $q(z_i)$ is no longer modified by worker

reliability in the previous iteration (Eq. (8)) and it no longer updates the parameters of the answer quality model (Eq. (23)).

The iterations through the relevant workers for each candidate influencer (rows 2-3) require a time complexity of $O(|\mathbf{A}|)$, where $|\mathbf{A}|$ denotes the number of non-zero entries of $\mathbf{A}$. Similarly, the time complexity for row 4-5 is $O(|\mathbf{A}|)$. The overall complexity of the algorithm is, therefore, $O(\#iter \times |\mathbf{A}| + \mathcal{T}_{\mathbf{W}})$, where $\#iter$ is the number of iterations in the variational inference algorithm and $\mathcal{T}_{\mathbf{W}}$ represents the complexity of learning $\mathcal{W}_I$ in a supervised learning setting.

5 EXPERIMENTS AND RESULTS

This section presents experimental results evaluating the performance of OpenCrowd[2] on two different domains, by comparing it against state-of-the-art Boolean and feature-based aggregation methods. In addition, we investigate the properties of our approach such as the impact of negative sampling on the performance. We start by introducing our experimental setup below before presenting the results of our experiments.

5.1 Experimental Setup

Crowdsourcing Task. We consider the problem of finding social influencers in two domains: fashion and information technology. For both domains, we published question-answering tasks on Figure Eight[3], asking workers to name social influencers they know. To set the context and promote workers to reflect on their experience, we asked workers to assess their domain-specific knowledge (five-point scale), estimate how often do they read social media posts from influencers (never, rarely, sometimes, always), and describe how they got to know the influencers. Workers name candidate influencers by providing their Twitter usernames, from which we retrieve social features (see "Social Feature Extraction").[4] The task took 2 minutes to complete on average. Workers who completed the task received a reward of 30 cents (USD), with an additional bonus: they were paid 10 additional cents (and up to 50 cents) for every social influencer they provided after naming 3 influencers.

Datasets. We collected two datasets of candidate influencers in two domains: *Fashion* and *Infotech*. The size of the collected datasets are comparable to typical datasets for Boolean answer aggregation [36]. Key statistics of these datasets are reported in Table 2. Our manual analysis (see "Expert Assessment") revealed that 30.64% and 43.39% of the crowd answers designate true influencers for Fashion and InfoTech, respectively. The relatively large number of crowd answers collected in a short period of time (<10 hours for both Fashion and InfoTech) confirms our assumption that crowdsourced open-ended question-answering can drastically speed up the data collection for finding social influencers. Moreover, the high sparsity of the answer matrices (Table 2) and the fact that the majority of the answers are incorrect substantiates the necessity of open-ended answers aggregation that takes into account the workers' reliability.

[2]The implementation is available here: https://github.com/eXascaleInfolab/OpenCrowd.
[3]https://www.figure-eight.com
[4]Twitter usernames are first verified automatically through Twitter API.

Table 2: Description of the datasets.

Dataset	#Cand. Infl.	#Workers	#Answers	Sparsity
Fashion	890	250	1416	99.36%
InfoTech	1057	200	1643	99.22%

Expert Assessment. We conducted a series of interviews with experts from three leading companies[5] that connect brands to social influencers. We distilled four main characteristics of influencer assessments: authenticity, dedication, branding, and communication. Following their guidelines and examples, three of the authors randomly selected 40% of the candidate influencers and labeled them by manually examining their profile and content on Twitter. In more detail, a candidate was considered as a real influencer whenever she: 1) tweets about a specific topic; 2) posts new content regularly; 3) keeps a consistent and unique style in her posts; and 4) communicates with her followers through comments (mostly for micro-influencers). The authors reached an initial agreement of over 80%. In case of disagreement, they discussed it until reaching a decision.

Social Feature Extraction. The features used in our framework are extracted from the Twitter account of the named candidate influencers. These features include metadata features such as the number of followers, number of followees and number of tweets, and semantic features such as the topics of a candidate influencer's tweets. In order to extract the topics from the tweets, we first represent all tweets as a bag of words. Then, we apply a grid search in {5, 10, 20, 50, 100} to set a threshold on the word's frequency. For our experiments, we keep only the words that appear more than 20 times. We finally compute the TF-IDF scores of the constructed bag of words and use the scores together with the other features to train our answer quality model.

Comparison Methods. Due to the lack of existing open-ended answers aggregation methods (cf. Section 2), we first compare against the following state-of-the-art closed-pool (Boolean) aggregation methods: 1) ZenCrowd [11], an expectation-maximization (EM) method that estimates worker reliability as a model parameter; 2) Dawid-Skene (DS) [10], an EM method that learns worker reliability as a confusion matrix; 3) GLAD [48], an EM method that simultaneously learns worker reliability and task difficulty; and 4) LFC [31], an EM method that incorporates priors in modeling worker reliability. Then, we compare against existing techniques that take into account task's features for answer aggregation: 1) LFC_SoT [41], a statistical model that estimates both worker reliability and task clarity by clustering workers into groups; 2) CUBAM [47], a Bayesian probabilistic model that learns worker reliability and task domains as a feature vector from the worker-answer matrix; and 3) iCrowd [12], a crowdsourcing framework that considers the topical similarity of tasks based on their textual description for worker reliability inference.

We compare all the EM-based methods in a semi-supervised setting by fixing the known labels in the EM algorithm [36] (See [38, 45] for the case of semi-supervised DS). Then, in order to apply

[5]Collabary (www.collabary.com), Influencer Check (www.influencer-check.ch), and Reachbird (www.reachbird.io)

Table 3: Performance (accuracy and AUC) comparison of aggregation techniques on two datasets with supervision degree s_deg from 50% to 90%. The best performance is highlighted in bold; the second best performance is marked by '*' for accuracy and by '+' for AUC.

Method	Metric	Fashion					InfoTech				
		50%	60%	70%	80%	90%	50%	60%	70%	80%	90%
DS	Accuracy	0.689	0.716*	0.703	0.688	0.711	0.662	0.660	0.626	0.641	0.536
	AUC	0.191	0.169	0.242+	0.244	0.263	0.174	0.203+	0.222+	0.255	0.272
GLAD	Accuracy	0.697	0.716*	0.724	0.700	0.688	0.669*	0.667	0.637	0.672	0.595
	AUC	0.183	0.189	0.229	0.224	0.263	0.150	0.186	0.138	0.219	0.307+
ZenCrowd	Accuracy	0.701	0.686	0.733*	0.702*	0.688	0.651	0.674*	0.664*	0.683*	0.627
	AUC	0.157	0.175	0.203	0.239	0.287+	0.146	0.198	0.212	0.246	0.234
LFC	Accuracy	**0.721**	0.694	0.718	0.691	0.755*	0.653	0.627	0.643	0.616	0.636*
	AUC	0.203+	0.203+	0.225	0.264+	0.277	0.189+	0.192	0.215	0.276+	0.307+
OpenCrowd	Accuracy	0.708*	**0.740**	**0.751**	**0.769**	**0.889**	**0.734**	**0.782**	**0.790**	**0.797**	**0.804**
	AUC	**0.304**	**0.350**	**0.350**	**0.452**	**0.495**	**0.270**	**0.279**	**0.353**	**0.326**	**0.339**

these methods to our problem, we use negative sampling to simulate a worker's answers of non-influencers by sampling the candidate influencers she does not name. We empirically determine the optimal sampling rates for each comparison method. Furthermore, for the techniques that model a task based on its textual description, we use the textual social features as input to model a candidate influencer.

To further investigate the benefits of taking into account the worker-answer matrix, we compare OpenCrowd against some feature-based methods: logistic regression (LR) and a multi-layer perceptron (MLP). We define two variants of our framework: 1) OpenCrowd-EM: OpenCrowd that aggregates workers' answers but models worker reliability as a fixed parameter; and 2) OpenCrowd, our framework that models worker reliability as a latent variable.

Parameter Settings. The parameters of our framework and those for model training are empirically set. We search for the best model architecture for MLP, and the predictor f in OpenCrowd-EM and OpenCrowd, with 0, 1, and 2 hidden layers, and apply a grid search in {64, 128, 256, 512, 1024} for the dimension of the hidden layers. In model training, we select learning rates from {0.0001, 0.001, 0.01, 0.1, 1} for the learning of $\mathcal{W}_I$ in all variants of our framework, as well as for the learning of r_j in OpenCrowd-EM. To investigate the impact of negative sampling, we experiment with sampling rates (s_rate) in {0, 0.1, 1, 10, 100} where $s\_rate = 10$ indicates for example that for each worker, the negative samples are ten times the size of the candidate influencers named (i.e., deemed as positive) by each worker. For OpenCrowd, we set the priors A and B by sampling from a uniform distribution $\sim [0, 10]$ and update them in the E-step according to Lemma 4.2.

Evaluation Protocols. We split the labeled subset of candidate influencers into training, validation, and test sets. OpenCrowd is trained on the answers in the training set, tuned on the validation set and evaluated on the test set. To investigate the impact of the degree of supervision (s_deg) on OpenCrowd performance, we split the labeled subset by $s\_deg \in \{50\%, 60\%, 70\%, 80\%, 90\%\}$, where $s\_deg = 60\%$ means that 60% of the labeled subset is used for training, and the rest for validation and test with equal split. We use accuracy and area under the precision-recall curve (AUC) to measure the performance. Higher values of accuracy and AUC indicate better performance. Note that given the imbalanced classes in our datasets, accuracy is dominated by the results on the non-influencers; similarly, the metric area under the ROC curve would also be biased to the non-influencers. In contrast, the AUC we use – area under the precision-recall curve – is *more indicative* of the performance in our context, as we are more interested in detecting real influencers from the workers' answers [7].

5.2 Comparison to Boolean Aggregation

Table 3 summarizes the performance of boolean answers aggregation methods on our two datasets with different supervision degrees. We make several observations.

First, we observe that ZenCrowd outperforms DS and GLAD in terms of accuracy and has a comparable performance in terms of AUC. Recall that ZenCrowd is less expressive compared to DS and GLAD, as it only models worker reliability as a parameter. In comparison, DS models worker reliability as a confusion matrix, and GLAD further models the task difficulty (in our context the ambiguity of a candidate influencer being the true influencer). The comparison result indicates that in our context, more expressive models do not necessarily lead to higher performance. This is likely due to the high sparsity of the worker-answer matrices that can easily lead to overfitting. Second, we observe that methods that model worker reliability as a latent variable with a prior distribution, namely LFC and our framework OpenCrowd, outperform the other methods. Such a result confirms the necessity of modeling worker reliability as a latent variable, as it helps to account for the confidence in estimating model parameters. This is particularly important to improve model robustness for sparse datasets similar to our case. We provide more results about this point in Section 5.4.

Most importantly, OpenCrowd achieves the best performance among all answers aggregation methods under comparison: it improves the state of the art by 6.94% accuracy and 62.06% AUC on Fashion, and by 17.56% accuracy and 33.54% AUC on InfoTech. This significant improvement clearly demonstrates the effectiveness of our framework in open-ended answers aggregation.

Table 4: Performance (accuracy and AUC) comparison of feature based aggregation techniques on the two datasets.

Dataset	Method	Acuracy	AUC
Fashion	CUBAM	0.718	0.290
	iCrowd	0.712	0.301+
	LFC_SoT	0.724*	0.253
	OpenCrowd	**0.751**	**0.350**
InfoTech	CUBAM	0.661	0.326
	iCrowd	0.630	0.372+
	LFC_SoT	0.698*	0.252
	OpenCrowd	**0.790**	**0.397**

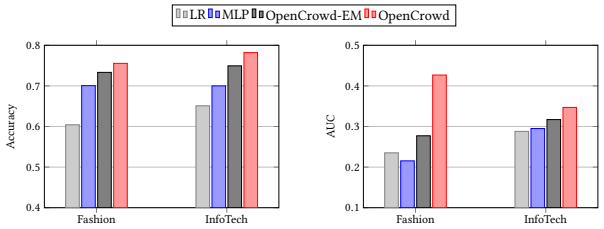
Impact of Supervision Degree. The supervision degree *s_deg* controls the number of observed labels in model training. We observe that the performance of our framework increases along with the increase of *s_deg*, as measured by both accuracy and AUC. This is natural as using more labeled data provides more information in discriminating influencers from non influencers. Such a pattern, however, is not observed for the other methods that we compare to. This is likely due to the fact that the other methods do not take advantage of the social features, which are useful as they serve as a means to propagate the labels to non-labeled candidate influencers. These results show that OpenCrowd is better at utilizing existing labels for answers aggregation.

Robustness. The learning of OpenCrowd involves two types of random processes, i.e., the random initialization of the parameters (e.g., $\mathcal{W}_I$) and negative sampling. To investigate their impacts on OpenCrowd performance, we measure the standard deviation of OpenCrowd performance over 10 runs. Results show that the standard deviation in terms of accuracy is 0.017 and 0.018 on Fashion and InfoTech, respectively; and in terms of AUC, the standard deviation is 0.023 and 0.028 on Fashion and InfoTech, respectively. The standard deviations are small compared to the absolute accuracy and AUC. Such a result is consistent across different supervision degrees. These results signify the robustness of OpenCrowd across different runs.

5.3 Comparison to Feature-Based Aggregation

We now compare the performance of our method against feature-based aggregation techniques. Table 4 shows the results of our comparison against these methods in terms of accuracy and AUC on both the Fashion and InfoTech datasets (with a supervision degree of 60%). From these results, we make the following observations.

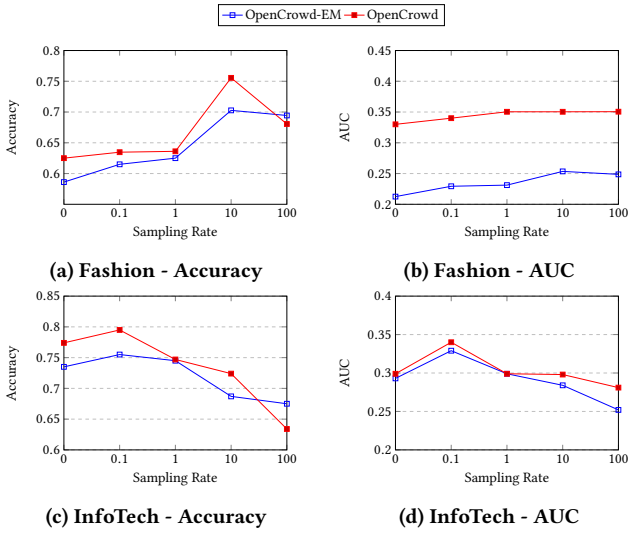
Among the baselines, LFC_SoT achieves the best accuracy yet the lowest AUC. In fact, LFC_SoT cannot handle the case where some workers do not give an answer to some tasks and hence cannot properly support negative sampling. Since the worker-answer matrix is very sparse in our setting, LFC_SoT labels most candidate influencers as negative (more than 75%). Therefore, LFC_SoT infers most true influencers to be non influencers and hence the results. In contrast, iCrowd achieves better performance in terms of AUC than CUBAM and LFC_SoT. Recall that iCrowd takes into account the social features of candidate influencers and combines them with worker reliability to infer the truth. In comparison, CUBAM models the task difficulty as a vector but relies solely on the worker-answer matrix. This result confirms the necessity of taking into account the social features to identify real influencers in the set of candidates.



(a) Accuracy **(b) AUC**

Figure 3: Comparison between feature-based methods and OpenCrowd variants measured by (a) Accuracy and (b) AUC.

Overall, OpenCrowd achieves the best performance among all feature-based aggregation methods: it outperforms the second best method by 3.7% accuracy and 16.27% AUC on Fashion and by 13.18% accuracy and 6.7% AUC on InfoTech (on average: 8.44% accuracy and 11.5% AUC). Unlike the baseline methods that do not use social features (e.g., CUBAM) or rely only on textual features (e.g., iCrowd), OpenCrowd is able to leverage any type of social features, including non-textual ones. More importantly, unlike the unsupervised topic modeling used by iCrowd, the supervised answer quality model in OpenCrowd learns from the labeled data the "weights" of social features, thereby making it better at influencer identification.



(a) Fashion - Accuracy **(b) Fashion - AUC**

(c) InfoTech - Accuracy **(d) InfoTech - AUC**

Figure 4: Performance of OpenCrowd with varying *s_rate*.

5.4 Properties of OpenCrowd

The comparison between OpenCrowd variants against feature-based methods (see "Comparison Methods") is shown in Figures 3(a,b). OpenCrowd-EM outperforms both LR and MLP by 18.25% and 5.82% accuracy and by 13.69% and 18.05% AUC, respectively. These results show the importance of considering worker reliability in aggregating workers' answers. Among the two variants, OpenCrowd outperforms OpenCrowd-EM by 7.37% accuracy and 31.62%

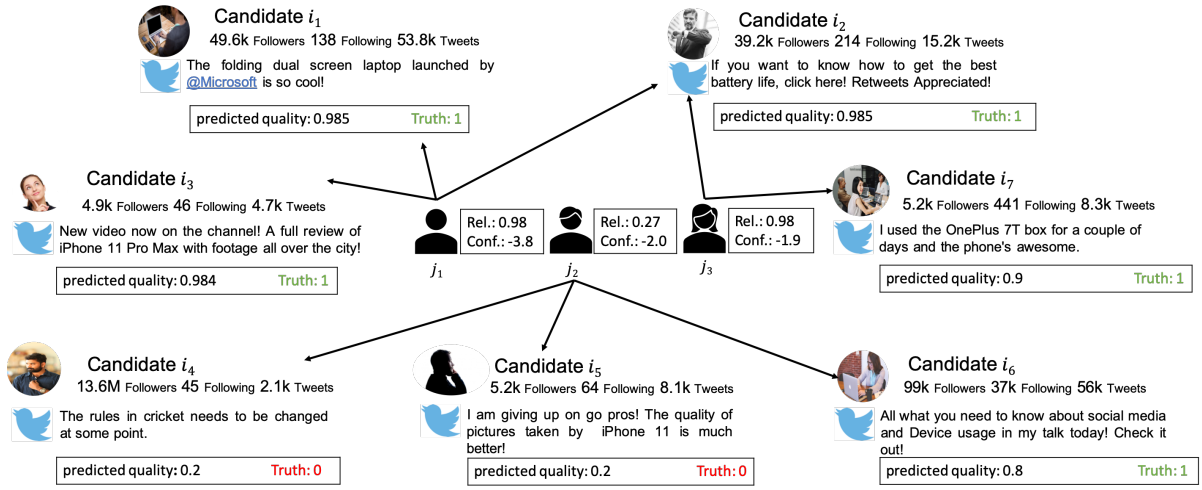


Figure 5: Examples of workers and their nominated candidate influencers in the InfoTech domain. We show for workers the inferred reliability (Rel.) and inference confidence (Conf.), and for candidate influencers the predicted quality as well as the ground-truth labels. Profile pictures are from public data sources and are randomly assigned to anonymize user identities.

AUC. This result indicates that modeling the worker reliability as a latent variable with a prior distribution not only makes the model more robust, but also improves the aggregation performance.

Impacts of Sampling Rate. The sampling rate s_rate controls the size of randomly sampled candidate influencers in estimating the workers' reliability. The results are shown in Figure 4. We observe that, as the sampling rate increases from 0 to 100, the performance first increases then decreases. Such a result is consistent on both datasets, measured by both accuracy and AUC. The optimal performance is reached for $s\_rate = 10$ for Fashion and $s\_rate = 0.1$ for InfoTech, indicating that workers' evaluation on candidate influencers they do not name is more negative on Fashion. Overall, the variation of the performance with different s_rate indicates the importance of selecting the optimal sampling rate. The similarity in performance variation across the two datasets again demonstrates the robustness of OpenCrowd.

Interpretation of Learning Results. Results of OpenCrowd can be explained in terms of the social features of candidate influencers and of the correlation between worker answers. We show in Figure 5 the learning results of real-world examples for three workers and seven candidate influencers from the InfoTech dataset. We also show the mean and confidence (differential entropy [24]) of worker reliability distribution (r_j), the predicted quality (θ_i) and the ground-truth labels of candidate influencers. We observe that workers who name real influencers have a high reliability as inferred by OpenCrowd, and otherwise have a low reliability. For example, the three influencers named by worker j_1, who has the highest reliability, are all real influencers. Among them, candidates i_1 and i_2 clearly exhibit influencer characteristics, e.g., they have a large number of followers and tweets dedicated to InfoTech. These results indicate that our approach is able to correctly infer the reliability of workers by leveraging the social features of the candidate influencers they named. Thanks to the worker's reliability, micro-influencers with a smaller number of followers, such as candidate i_3, can also be successfully detected by our approach. We also observe that worker j_3 has the same high reliability as j_1, despite the fact that only one of the candidates (i_2) she named exhibits influencer characteristics. This is because OpenCrowd leverages the correlation between worker answers in reliability inference: i_2 is named by both workers j_1 and j_3. The difference between the number of answers provided by j_1 and j_3 is captured through the confidence measure: i_1 has a higher confidence than i_3. Most importantly, we observe that the high reliability inferred for j_3 helps to detect an additional micro-influencer that she named, i.e., i_7. These results demonstrate that OpenCrowd can find micro-influencers through reliable workers, whose reliability can be inferred either through further named candidate influencers or through similar workers.

6 CONCLUSION

In this paper, we have presented OpenCrowd, a unified Bayesian framework that seamlessly incorporates machine learning and crowdsourcing for social influencer identification. Our framework aggregates open-ended answers while modeling both the quality of the workers' answers and their reliability. We derived a principled optimization algorithm based on variational inference with efficient incremental update rules for learning OpenCrowd parameters. Extensive validation on two real-world datasets shows that OpenCrowd is an effective and robust framework that substantially outperforms state-of-the-art answers aggregation methods. Results further show that our framework is particularly useful in finding micro-influencers by exploiting the social features and the correlation between worker answers.

ACKNOWLEDGMENTS

This work is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 732328 (FashionBrain) and the European Research Council (ERC) under grant agreement No 683253 (GraphInt).

REFERENCES

[1] Nitin Agarwal, Huan Liu, Lei Tang, and Philip S. Yu. 2008. Identifying the Influential Bloggers in a Community. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM)*. ACM, Palo Alto, California, USA, 207–218.

[2] Shane Barker. 2019. The Ultimate Guide to Micro-Influencers. https://shanebarker.com/blog/micro-influencers-guide/. Accessed: 2019-10-11.

[3] Bin Bi, Yuanyuan Tian, Yannis Sismanis, Andrey Balmin, and Junghoo Cho. 2014. Scalable topic-specific influence analysis on microblogs. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, New York, NY, USA, 513–522.

[4] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[6] Robert M Bond, Christopher J Fariss, Jason J Jones, Adam DI Kramer, Cameron Marlow, Jaime E Settle, and James H Fowler. 2012. A 61-million-person experiment in social influence and political mobilization. *Nature* 489, 7415 (2012), 295.

[7] Kendrick Boyd, Kevin H Eng, and C David Page. 2013. Area under the precision-recall curve: point estimates and confidence intervals. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*. Springer, Prague, Czech Republic, 451–466.

[8] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and Krishna P Gummadi. 2010. Measuring user influence in twitter: The million follower fallacy. In *Fourth International AAAI Conference on Weblogs and Social Media (ICWSM)*. The AAAI Press, Washington, DC, USA, 10–17.

[9] Zhiyuan Cheng, James Caverlee, Himanshu Barthwal, and Vandana Bachani. 2014. Who is the Barbecue King of Texas?: A Geo-spatial Approach to Finding Local Experts on Twitter. In *Proceedings of the 37th ACM SIGIR International Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, Gold Coast, Queensland, Australia, 335–344.

[10] P. Dawid, A. M. Skene, A. P. Dawidt, and A. M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28 (1979), 20–28.

[11] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2012. ZenCrowd: Leveraging Probabilistic Reasoning and Crowdsourcing Techniques for Large-scale Entity Linking. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. ACM, Lyon, France, 469–478.

[12] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. 2015. icrowd: An adaptive crowdsourcing framework. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, Melbourne, Victoria, Australia, 1015–1030.

[13] Ju Fan, Jiarong Qiu, Yuchen Li, Qingfei Meng, Dongxiang Zhang, Guoliang Li, Kian-Lee Tan, and Xiaoyong Du. 2018. Octopus: An online topic-aware influence analysis system for social networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE Computer Society, Paris, France, 1569–1572.

[14] Samuel Gershman and Noah Goodman. 2014. Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society (CogSci)*, Vol. 36. cognitivesciencesociety.org, Quebec City, Canada.

[15] Behnam Hajian and Tony White. 2011. Modelling influence in a social network: Metrics and evaluation. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom)*. IEEE Computer Society, Boston,MA, USA, 497–500.

[16] Leading Global Influencer Marketing Agency Relatable in collaboration with 350 Brands and Agencies. 2019. The 2019 state of influencer marketing report. https://www.relatable.me/the-state-of-influencer-marketing-2019. Accessed: 2019-05-22.

[17] Alexy Khrabrov and George Cybenko. 2010. Discovering influence in communication networks using dynamic graph analysis. In *2010 IEEE Second International Conference on Social Computing (SocialCom)*. IEEE Computer Society, Minneapolis, Minnesota, USA, 288–294.

[18] Himabindu Lakkaraju, Jure Leskovec, Jon Kleinberg, and Sendhil Mullainathan. 2015. A bayesian framework for modeling human evaluations. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*. SIAM, Vancouver, BC, Canada, 181–189.

[19] Janette Lehmann, Carlos Castillo, Mounia Lalmas, and Ethan Zuckerman. 2013. Finding News Curators in Twitter. In *Companion Proceedings of the 22nd International Conference on World Wide Web (WWW)*. ACM, Rio de Janeiro, Brazil, 863–870.

[20] Daifeng Li, Xin Shuai, Guozheng Sun, Jie Tang, Ying Ding, and Zhipeng Luo. 2012. Mining Topic-level Opinion Influence in Microblog. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, Maui, Hawaii, USA, 1562–1566.

[21] Y. Li, J. Fan, Y. Wang, and K. Tan. 2018. Influence Maximization on Social Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 30, 10 (2018), 1852–1872.

[22] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. 2015. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, Sydney, NSW, Australia, 745–754.

[23] Zhanyu Ma and Arne Leijon. 2011. Bayesian estimation of beta mixture models with variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33 (2011), 2160–2173.

[24] Joseph Victor Michalowicz, Jonathan M Nichols, and Frank Bucholtz. 2013. *Handbook of differential entropy*. Chapman and Hall/CRC.

[25] Michael A Nielsen. 2015. *Neural networks and deep learning*. Vol. 25. Determination press San Francisco, CA, USA:.

[26] Besmira Nushi, Ece Kamar, Eric Horvitz, and Donald Kossmann. 2017. On human intellect and machine failures: troubleshooting integrative machine learning systems. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, San Francisco, California, USA, 1017–1025.

[27] Aditya Pal and Scott Counts. 2011. Identifying topical authorities in microblogs. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, Hong Kong, China, 45–54.

[28] Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2, 1–2 (2008), 1–135.

[29] Aditya Parameswaran, Akash Das Sarma, and Vipul Venkataraman. 2016. Optimizing Open-Ended Crowdsourcing: The Next Frontier in Crowdsourced Data Management. *Bulletin of the Technical Committee on Data Engineering* 39, 4 (2016), 26.

[30] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2110–2119.

[31] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11, Apr (2010), 1297–1322.

[32] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. 2012. Finding Expert Users in Community Question Answering. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. ACM, Lyon, France, 791–798.

[33] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, San Francisco, CA, USA, 1135–1144.

[34] Matthew Richardson and Pedro Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, Edmonton, Alberta, Canada, 61–70.

[35] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. 2008. 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 (KDD)*. ACM, Las Vegas, Nevada, USA, 614–622.

[36] Aashish Sheshadri and Matthew Lease. 2013. Square: A benchmark for research on computing crowd consensus. In *First AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*. AAAI, Palm Springs, CA,USA, 156–164.

[37] Chonggang Song, Wynne Hsu, and Mong-Li Lee. 2017. Temporal Influence Blocking: Minimizing the Effect of Misinformation in Social Networks. In *33rd IEEE International Conference on Data Engineering (TKDE)*. IEEE Computer Society, San Diego, CA, USA, 847–858.

[38] Wei Tang and Matthew Lease. 2011. Semi-supervised consensus labeling for crowdsourcing. In *SIGIR 2011 workshop on crowdsourcing for information retrieval (CIR)*. 1–6.

[39] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence Maximization in Near-Linear Time: A Martingale Approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, Melbourne, Victoria, Australia, 1539–1554.

[40] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence Maximization: Near-optimal Time Complexity Meets Practical Efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, Snowbird, Utah, USA, 75–86.

[41] Yuandong Tian and Jun Zhu. 2012. Learning from Crowds in the Presence of Schools of Thought. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, Beijing, China, 226–234.

[42] Dimitris G Tzikas, Aristidis C Likas, and Nikolaos P Galatsanos. 2008. The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine* 25, 6 (2008), 131–146.

[43] Christophe Van den Bulte and Yogesh V Joshi. 2007. New product diffusion with influentials and imitators. *Marketing Science* 26, 3 (2007), 400–421.

[44] Jennifer Wortman Vaughan. 2018. Making Better Use of the Crowd: How Crowdsourcing Can Advance Machine Learning Research. *Journal of Machine Learning*

Research 18 (2018), 193–1.

[45] Jing Wang, Panagiotis G. Ipeirotis, and Foster Provost. 2011. Managing crowdsourcing workers. In *In The 2011 Winter Conference on Business Intelligence*. 10–12.

[46] Wei Wei, Gao Cong, Chunyan Miao, Feida Zhu, and Guohui Li. 2016. Learning to find topic experts in Twitter via different relations. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 28, 7 (2016), 1764–1778.

[47] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. 2010. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Vancouver, British Columbia, Canada, 2424–2432.

[48] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. 2009. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Vancouver, British Columbia, Canada, 2035–2043.

[49] Barrett Wissman. 2019. Micro-Influencers: The Marketing Force Of The Future? https://www.forbes.com/sites/barrettwissman/2018/03/02/micro-influencers-the-marketing-force-of-the-future. Accessed: 2019-10-11.

[50] Jie Yang, Thomas Drake, Andreas Damianou, and Yoelle Maarek. 2018. Leveraging crowdsourcing data for deep active learning an application: Learning intents in alexa. In *Proceedings of the 2018 World Wide Web Conference (WWW)*. ACM, Lyon, France, 23–32.

[51] Jie Yang, Alisa Smirnova, Dingqi Yang, Gianluca Demartini, Yuan Lu, and Philippe Cudré-Mauroux. 2019. Scalpel-cd: leveraging crowdsourcing and deep probabilistic modeling for debugging noisy training data. In *Proceedings of the 2019 World Wide Web Conference (WWW)*. ACM, San Francisco, CA, USA, 2158–2168.

[52] Yudian Zheng, Guoliang Li, and Reynold Cheng. 2016. DOCS: a domain-aware crowdsourcing system using knowledge bases. *Proceedings of the VLDB Endowment* 10, 4 (2016), 361–372.

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