Truth Inference for Crowdsourcing via Bias-Variance Fusion Graph Embedding

Wei Zong

weizong@xidian.edu.cn

Xidian University

Huize Feng

Xidian University

Zongyao Nie

Xidian University

Research Article

Keywords: Crowdsourcing, Truth Inference, Worker reliability, Graph Embedding, Convolutional Neural Network

Posted Date: October 16th, 2025

DOI: https://doi.org/10.21203/rs.3.rs-7637663/v1

Additional Declarations: No competing interests reported.

Truth Inference for Crowdsourcing via Bias-Variance Fusion Graph Embedding

Wei Zong · Huize Feng · Zongyao Nie

Abstract

Crowdsourcing has served as an effective method for data collection and labeling. However, the varying abilities of individual workers lead to inconsistent quality in the collected data. The existing truth inference methods always assign initial reliability values to workers and refine them iteratively. This methodology for estimating worker reliability fails to accurately assess worker reliability and consequently reduces the accuracy of truth inference. In this article, we explore a novel strategy to measure worker reliability by estimating the label quality of each worker for individual instances. Specifically, we introduce two metrics—bias and variance—to quantify, respectively, the discrepancy between a worker's labels and those of other workers on similar instances (bias), and the inconsistency in a worker's own labels across different instances (variance). These components are integrated to compute the label quality of each worker for inference instances, enabling accurate assessment of their actual labeling capability and determining their reliability. Building upon this, we propose Graph Embedding for Truth Inference with Worker Reliability (GETI-WR), a bias-variance fusion graph embedding method for crowdsourced truth inference. GETI-WR leverages the obtained worker reliability to map the "task-worker-label" graph into a low-dimensional continuous vector space. Then we use a graph neural network to transform the crowdsourcing problem into a graph node prediction task to enhance truth discovery efficiency.Experimental results on two real-world datasets demonstrate that compared to six state-of-the-art baseline methods, our method achieves superior performance in both accuracy and F1-score.

Keywords Crowdsourcing · Truth Inference · Worker reliability · Graph Embedding · Convolutional Neural Network

Corresponding author: Wei Zong
weizong@xidian.edu.cn

Huize Fong
23061213007@stu.xidian.edu.cn

Zongyao Nie
N2323985418@Outlook.com

Xidian University, School of Information Management,Xi' an,710126,Shaanxi,China

1 Introduction

In recent years, despite significant advances in computational data processing capabilities, computers still face substantial challenges in domains requiring high-level human cognitive abilities, such as image recognition [2], machine translation [3], and sentiment analysis [4]. Tasks in these fields—including interpreting ambiguous information, grasping complex contexts and background knowledge, handling subjectivity and nuances like sarcasm or cultural metaphors, and performing open-ended creative work—are often intuitive for humans but exceptionally difficult for machines. With the flourishing development of machine learning particularly deep learning, the demand for large-scale, high-quality labeled data has surged dramatically. However, relying on domain experts to obtain ground truth labels for this data is typically time-consuming and costly [5], creating a bottleneck for technological advancement.

To this end, crowdsourcing [1] has emerged as an efficient and economical solution. By distributing tasks to numerous volunteers via internet platforms (e.g., Tencent Souhuobang, JD Crowdsourcing, Zhubajie, Baidu CrowdTest), crowdsourcing enables collaborative, cost-effective data labeling. Owing to its distinct advantages, crowdsourcing has been widely adopted for data collection and annotation. However, crowdsourced labeling is not without limitations, due to the openness of crowdsourcing platforms, workers' abilities, expertise, and concentration levels vary considerably, resulting in inconsistent labeling quality and introducing substantial noisy labels [6]. To address this challenge and enhance final label quality, repeated labeling [7] has become a widely accepted practice, where each data instance is independently labeled by multiple distinct workers.

How to infer the most probable ground truth label for a data instance from these potentially conflicting and erroneous labels. This process is termed truth inference [8], which aims to integrate crowdsourced labels through algorithmic models to produce an "aggregated label" or "predicted truth" for each instance.

Majority Voting (MV) is the simplest truth inference algorithm. As the name suggests, it calculates the vote count for each label in the dataset and selects the most frequently occurring label from the workers' inputs as the predicted truth label (aggregated label) for that instance. While straightforward and effective in some real-world crowdsourcing scenarios, MV suffers from a critical flaw: that all workers are equally reliable, an assumption clearly unreasonable in practical crowdsourcing settings. To overcome this limitation, numerous improved methods have been developed. A representative enhancement is the Weighted Majority Voting (WMV) algorithm [9]. WMV assigns weights to workers, granting higher voting power to high-quality workers and reducing interference from low-quality ones, thereby optimizing label aggregation and improving the accuracy and robustness of truth inference. Another influential approach leverages probabilistic graphical models based on Expectation Maximization (EM). These methods often use the Expectation-Maximization (EM) algorithm to optimize and simultaneously estimate latent worker quality and infer the truth. Recently, neural network approaches like Graph Neural Networks (GNNs) have been introduced to truth inference. By employing graph embedding to model high-order interactions between tasks and workers, GNNs exploit the structural relationships and latent interactions within crowdsourced data to solve the truth inference problem [10], significantly enhancing both accuracy and efficiency.

Existing truth inference methods have made significant contributions to improving inference accuracy. However, these methods commonly exhibit a crucial limitation: a lack of precision in

modeling and quantifying worker reliability—the core metric representing a worker's labeling quality. Many approaches rely on rudimentary statistics or random initialization for reliability parameters, failing to accurately capture the performance differences between workers. This imprecision can introduce bias into the final results, increase the number of iterations needed for convergence, and add unnecessary computational complexity. Therefore, accurately assessing worker reliability is foundational to building efficient and robust truth inference models and is crucial for obtaining high-quality aggregated labels.

To address these limitations, we propose a Graph Embedding-based Truth Inference method with Worker Reliability (GETI-WR), which fuses bias and variance metrics. First, we introduce a novel, multi-dimensional framework for quantifying worker reliability. Specifically: Bias measures the labeling discrepancy between a worker and other workers on identical instances.Variance quantifies the inconsistency in a worker's own labels across similar instances.We argue that high-quality workers should exhibit low bias when labeling identical instances and low variance when labeling similar instances. These two metrics are then integrated to compute the worker's final reliability score. Subsequently, this reliability score is incorporated into a graph-based neural model. We represent workers and tasks as nodes in a graph, using the computed reliabilities to define their feature representations. Utilizing a Graph Autoencoder (GAE) and GNN, we reframe the crowdsourced truth inference problem as a graph node classification task. Node features are iteratively refined through graph convolution. The final inference is obtained by applying max-pooling and an activation function to the optimized task node features.

In summary, the main contributions of this work are as follows:

- Innovative Worker Reliability Calculation: We propose a novel method for computing worker reliability by introducing the statistical metrics of bias and variance to quantify labeling consistency and stability, respectively. These metrics are then weighted and fused to construct a comprehensive reliability score. This approach effectively distinguishes high-quality from low-quality workers, providing credible evidence for subsequent models, while overcoming the limitations of traditional single-metric evaluations and enhancing robustness against noisy labeling.
- Graph Embedding Truth Inference Model: We design a graph embedding-based truth inference model that abstracts workers and tasks as graph nodes. Worker reliability is leveraged to generate node feature vectors, which are jointly optimized using a Graph Autoencoder (GAE) and Graph Neural Network (GNN). The optimized feature vectors are processed through max-pooling and activation functions to derive class probabilities for task nodes, enabling end-to-end truth inference. This framework fully exploits the global interaction properties of graph structures, significantly improving inference accuracy in complex noisy scenarios.
- Comprehensive Experimental Validation: We conducted extensive comparative experiments on two real-world datasets from the CEKA platform, evaluating the proposed GETI-WR method against six state-of-the-art truth inference algorithms. The experimental results demonstrate that GETI-WR outperforms all other contemporary advanced truth inference methods.

2 Related work

In recent years, crowdsourced truth inference has garnered significant attention across data mining, database systems, and computer science due to its substantial advantages in tackling problems

challenging for computers alone. Consequently, improving its accuracy has become an urgent research priority. Extensive studies have been conducted by domestic and international scholars in this field, yielding notable progress.

Research related to crowdsourced truth inference can be traced back to Dawid et al., who employed the Expectation Maximization (EM) [11] algorithm to compute the maximum likelihood estimation of observers' error rates. The EM algorithm enables more effective estimation of errors in observers' answers, thereby enhancing data accuracy and reliability. Currently, research on crowdsourced truth inference methods is primarily categorized into three types [12]: direct computation methods, optimization-based methods, and probabilistic graphical model-based methods. With the advancement of deep learning, deep neural networks have also begun to be integrated into crowdsourcing learning.

Direct computation methods are relatively straightforward. Early approaches typically adopted a redundancy strategy, assigning the same task to multiple workers and then aggregating their labels to determine the ground truth for each task. These methods do not explicitly model the tasks or workers but directly infer the truth from the workers' responses. The most widely used technique is Majority Voting (MV) [13, 14], which assumes the label agreed upon by the majority of workers represents the truth. While simple and often effective, MV suffers from a critical flaw: it implicitly assumes all workers possess equal labeling ability, ignoring individual differences and variations in performance across different task types. This assumption significantly compromises truth inference accuracy in real-world scenarios.

Optimization-based methods center on defining an objective function that captures the relationship between worker reliability and task truth. This function minimizes the total weighted distance between the inferred worker reliabilities and the task truths, typically optimized iteratively. PM (Probabilistic Matrix Factorization) [15] is a classic algorithm in this category, modeling each worker's quality as a single scalar value and inferring both worker quality and task truth by optimizing the objective function. However, PM neglects the impact of the number of tasks answered by a worker on their reliability. To address this, the CRH method [16] employs an optimization framework to minimize the overall deviation between task truths and multiple crowdsourced answers, thereby jointly inferring truths and worker qualities. The CATD model [17] further incorporates worker confidence alongside reliability. It utilizes a chi-square distribution to establish a 95% confidence interval corresponding to the number of tasks answered by a worker; confidence increases with the volume of answered tasks.

Probabilistic graphical model (PGM)-based methods are among the most widely used approaches. These models represent the conditional dependency structure between random variables as a graph to solve truth inference. Gianluca et al. proposed ZenCrowd [18] for large-scale entity linking tasks, developing a probabilistic framework-based system integrated with automatic entity extraction, ranking, and matching techniques. The influential GLAD method [19] introduced by Jacob Whitehill et al. in 2009, extends ZenCrowd by jointly modeling both worker reliability and task difficulty. Yang et al. [20] further modeled the interrelationships between task difficulty, worker reliability, and task truth, iteratively refining them via the EM algorithm to infer the truths. The DASM [21] leverages label similarity, modeling the relationships between task difficulty, worker reliability, candidate label similarity, and the correct label to infer truths. A key limitation of PGM-based methods, however, is their reliance on manually setting initial parameters based on the application context. Poor initialization can trap them in local optima, leading to inefficient algorithms.

Recent advances in deep learning have spurred the integration of novel deep learning techniques with crowdsourced truth inference, yielding promising results. Deep learning's strengths—powerful feature extraction, non-linear modeling capabilities, and the ability to automatically learn complex patterns from large-scale data—offer fresh perspectives on traditional problems. For instance, Wu et al. [22] proposed a label confidence-based clustering algorithm to enhance aggregated label quality in crowdsourced data labeling. Their model utilizes worker reliability to derive label confidence and generate clustering features. Consequently, fusing crowdsourced truth inference with cutting-edge technologies like deep learning represents a highly promising research direction. Zhou et al. [23] introduced a convolutional neural network (CNN)-integrated optimization model for truth inference. This method iteratively updates worker reliability using an optimization model, constructs a graph where tasks and workers are nodes and labels are edges, transforms the derived reliability into node feature vectors via graph embedding, and ultimately reframes truth inference as a node prediction problem solvable by graph neural networks. This demonstrates that integrating deep learning not only overcomes limitations of traditional methods but also leverages its strengths to enhance model performance and practicality.

In summary, while existing methods approach truth inference from diverse angles, most calculate worker reliability either via random initialization or by relying on reliability estimates derived directly from the raw dataset. However, these methods suffer from two major drawbacks: (1) they fail to accurately model workers of differing abilities, and (2) datasets pre-equipped with reliable worker reliability are scarce and lack generalizability.

To address these limitations, this paper pioneers a novel worker reliability measurement strategy. We propose a dual-perspective framework assessing a worker's label proficiency through consistency (bias) and stability (variance). Based on this, we introduce GETI-WR (Graph Embedding for Truth Inference with Worker Reliability), a novel crowdsourced truth inference method that fuses bias-variance metrics with graph embedding and graph neural networks. Section 3 elaborates on the detailed workflow of the proposed GETI-WR method.

3 Preliminary

In traditional crowdsourcing scenarios, a crowdsourced dataset is typically represented as a collection $D=\{(x_n, L_i)\}_{n=1}^{N}$, where N denotes the total number of instances in D. x_n represents the n-th instance in D, and $L_n = \{l_n^r\}_{r=1}^{R}$ represents a set of multiple noisy labels. Here, l_n^r is the label assigned by crowdsourced worker u_r (r=1,2,3...R),which taking a value from a fixed label set $\{c_1, c_2, \dots c_Q, -1\}$. R and Q denote the total number of crowdsourced workers and distinct instance classes, respectively. When $l_n^r = -1$, it indicates that worker u_r did not label instance x_n. The goal of crowdsourced truth inference is to infer the predicted truth $\hat{y}_n$ for each instance x_n, aiming for maximal alignment with the unknown actual ground truth y_n.

Furthermore, via graph embedding, we transform the traditional crowdsourced dataset $D = \{(x_n, L_n)\}_{n=1}^{N}$into a triplet structure $T = \{X, U, L\}$representing a graph. This structure comprises: a Task Node Set$X = \{x_1, x_2, \dots, x_N\}$,where each node corresponds to an instance; a Worker Node Set $U = \{u_1, u_2, \dots, u_R\}$, where each node corresponds to a worker; and a Label Set $L = \{l_n^r | < u_r, x_n, l_n^r >\}$,where an edge $< u_r, x_n, l_n^r >$ connects worker node u_r to task node x_n and is labeled with the value l_n^r assigned by worker u_r to instance x_n. This triplet structure T explicitly models relationships between crowdsourcing tasks and workers: for any task node $x_n \in X$, it can

receive multiple edges with potentially conflicting labels from worker nodes in U, representing diverse labeling actions on the task.

After defining the graph structure T, we obtain feature vectors for each node to construct the feature matrix , perform operations including aggregation, convolution, and pooling on this matrix, and derive final prediction results from the processed representations. Key definitions and corresponding symbols in this article are summarized in Table I.

Table 1 Commonly used symbols

Notation	Description
T	Crowdsourced Dataset, Composed of $T = (X, U, L)$
X, x_n	Task Node Set , n-th task node x_n
U, u_r	Worker Node Set, r-th worker node u_r
L， l_n^r	Label Set: L,Label assigned by worker r to task n
b_{nr}	Discrepancy between worker u_r and other workers on task x_n
v_{nr}	Inconsistency of worker u_r on instances similar to x_n
w_{nr}, w_r	Labeling quality of worker u_r on instance x_n, reliability of worker r
θ_k^r, θ'^n_k,	k-th dimension of feature vector for worker node u_r, k-th dimension of feature vector for task node x_n
α_r, β_n, X_n	Feature vector of worker node u_r, Feature vector of task node x_n, Feature vector of graph G_n
G, V, E, A	Graph set, Node set, Edge set, Adjacency matrix

4 Worker Reliability Computation Based on Bias-Variance Fusion

4.1 Dual-Dimensional Characterization of Worker Reliability

To adapt statistical metrics to the crowdsourcing context, we extend the concept of bias to denote the discrepancy between a worker's label and an instance's ground truth, while variance signifies the fluctuation in the quality of a worker's labels. To clarify these extended definitions, Figure 1 provides an illustrative analogy comparing a worker labeling a specific instance to an athlete shooting at a target, where the entire target represents the instance, the red bullseye corresponds to the actual ground truth, and multiple impact points symbolize the worker's labels.Figure.1(a)–(d) depict a worker's performance across four distinct labeling scenarios for the same instance. The worker's labeling bias decreases as their labels cluster closer to the bullseye, while their labeling variance decreases as the labels exhibit less dispersion. As Figure.1(a) demonstrates, a worker achieves the highest labeling quality when both bias and variance are minimized. Consequently, high-quality workers should exhibit low bias and low variance.

In practical crowdsourcing, however, workers rarely provide multiple labels for identical instances. We posit that similar instances—presumably belonging to the same class—persistently exist in crowdsourced datasets. Thus, for each worker, labels assigned to multiple similar instances can be treated as approximate replicates of labels for a single instance.

Furthermore, since the ground truth (bullseye) remains unknown, we leverage labels from other workers on the same instance as a proxy reference. This approach is justified by the consensus principle: high-quality workers exhibit stronger agreement with peers [24, 25], implying that workers with low ground-truth bias inherently display low discrepancy relative to others. Bias is therefore estimated by comparing a worker's label against those of other workers on identical instances. For variance, reflecting label dispersion in the analogy, we measure variation in a worker's own labels across class-similar instances.

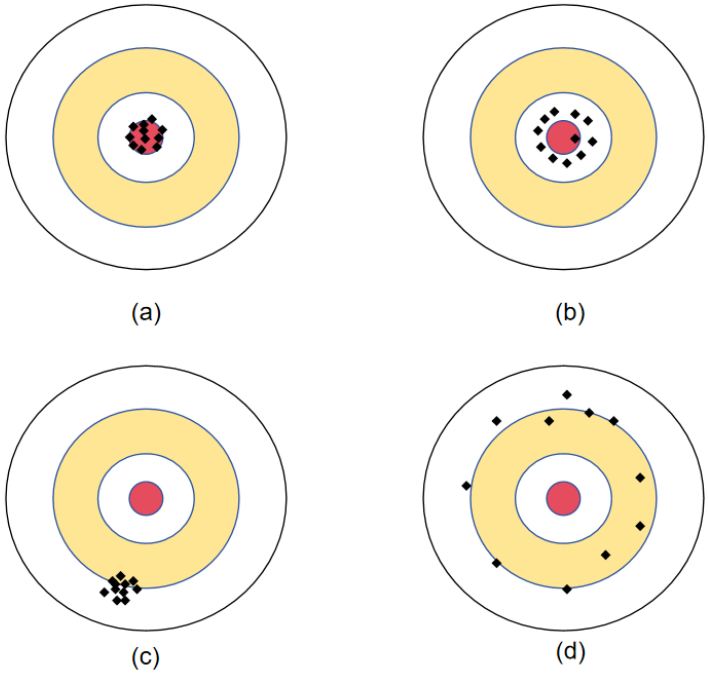

Fig.1 Schematic diagram of worker labeling bias and variance

4.2 K-Nearest Neighbors Acquisition for Instances

As established previously, computing a worker's final reliability requires firstly determining their labeling quality for each instance . For worker u_r, w_{nr} denotes his labeling quality on instance x_n. If u_r did not label x_n, w_{nr} is set to 0. We employ two variables, b_{nr} and v_{nr}, to measure u_r's labeling bias and variance on x_n, respectively. Critically, higher b_{nr} (or v_{nr}) indicates lower bias (or variance), corresponding to higher w_{nr}. Thus, w_{nr} exhibits a positive correlation with both b_{nr} and v_{nr}. However, calculating labeling bias b_{nr} necessitates identifying the K-nearest neighbors of each instance. We first apply the KNN algorithm to acquire these neighbors. Let T_r be the subset of T comprising all instances labeled by u_r, and let x_{nkr} denote the k-th neighbor of x_n within T_r. A special case arises when the number of instances u_r actually labeled is less than the preset K. In this scenario, K is dynamically adjusted to the actual number of labeled instances for the current worker-instance pair. Figure. 2 illustrates the overall workflow for computing b_{nr}, v_{nr}, and w_{nr}.

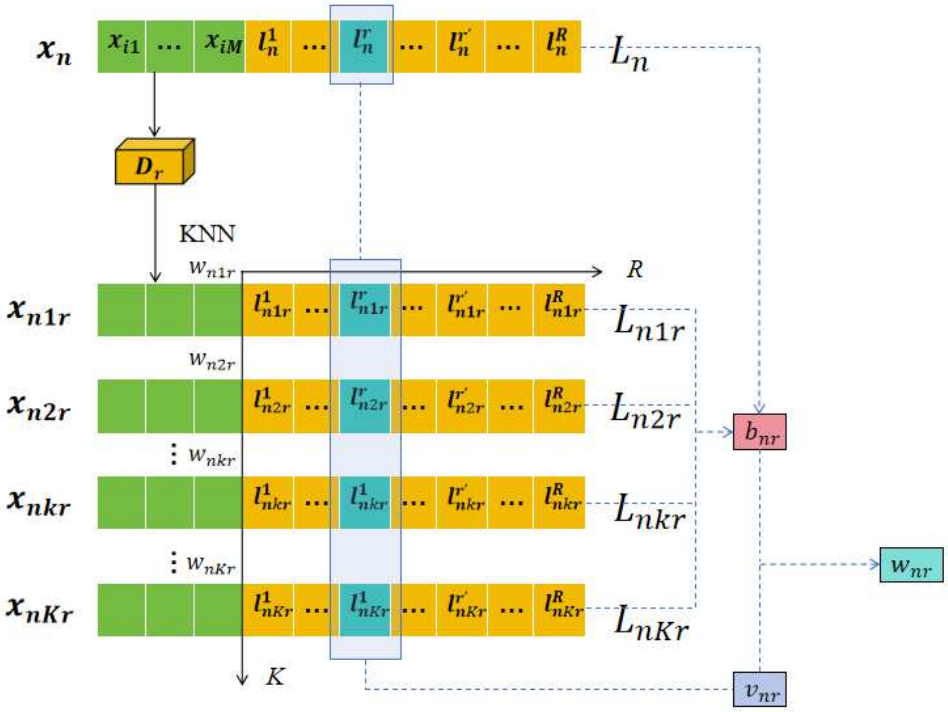

Fig.2 Schematic framework of the worker reliability computation

In the KNN algorithm, distance computation constitutes the most critical step. This paper employs the Heterogeneous Euclidean-Overlap Metric (HEOM) to compute distances between instances, as this metric effectively handles instances with heterogeneous feature spaces. The aggregate distance $d\,(x_n, x_h)$ between two instances x_n and x_h is defined as follows:

$$d(x_n, x_h) = \sqrt{\sum_{m=1}^{M} d_m(x_n, x_h)^2} \tag{1}$$

where M denotes the total number of features across instances, and $d_m(x_n, x_h)$ represents the distancebetween instances x_n and x_h on the m-th feature. The computation of $d_m(x_n, x_h)$is defined as follows:

$$d_m(x_n, x_h) = \begin{cases} 1, & x_{nm} or\ x_{hm}\ is\ unknown \\ Nominal(x_{nm}, x_{hm}), & x_{nm}\ is\ nominal \\ Numerical(x_{nm}, x_{hm}), & otherwise \end{cases} \tag{2}$$

wherex_{nm} denotes the m-th feature value of instance x_n, $Nominal$ represents the distance metric for categorical features, and $Numerical$ signifies the distance metric for continuous-valued features.The functions $Nominal(x_{nm}, x_{hm})$and$Numerical(x_{nm}, x_{hm})$are formally defined as follows:

$$Nominal(x_{nm}, x_{hm}) = \begin{cases} 0, & x_{nm} = x_{hm} \\ 1, & otherwise \end{cases} \quad (3)$$

$$Numerical(x_{nm}, x_{hm}) = \frac{|x_{nm} - x_{hm}|}{max_m - min_m} \quad (4)$$

In Formula (4), max_m and min_m denote the maximum and minimum values, respectively, of the m-th feature within the worker-specific subset T_r.

Utilizing the formulas above, we compute the distance between instance x_n and every instance in the worker-specific subset T_r. After sorting these distances, the K-nearest neighbors $\{x_{nkr}\}_{k=1}^{K}$ of x_n are identified. To mitigate the influence of low-similarity neighbors while amplifying the contribution of high-similarity neighbors, we assign a similarity weight w_{nkr} to each neighbor. The weight for the k-th neighbor is computed as follows:

$$w_{nkr} = 1 - \frac{d(x_n, x_{nkr})}{d(x_n, x_{nKr})} \quad (5)$$

where $d(x_n, x_{nkr})$ denotes the distance between x_n and x_{nkr}, and $d(x_n, x_{nKr})$ represents the distance between x_n and its farthest neighbor x_{iKr} (the K-th nearest neighbor) in the ordered set. Crucially, as a neighbor's distance from x_n increases, its similarity decreases, resulting in a corresponding decrease in weight w_{nkr}. Having identified the K-nearest neighbors of x_n and assigned similarity weights, we now proceed to formally define the computation of labeling bias b_{nr} and labeling variance v_{nr}.

4.3 Bias Computation

As hypothesized earlier, high-quality workers exhibit a greater propensity to reach consensus with peers than low-quality workers [21, 22]. Higher agreement between a worker's label and those of other workers on the same instance correlates strongly with lower bias relative to the ground truth—a relationship empirically validated in prior studies. Therefore, we estimate b_{nr} using the labels provided by u_r and other workers $u_{r'} (r' = 1,2,\ldots R, r' \neq r)$ on identical instance. Specifically, we define b_{nr}^1 as the label agreement of u_r with other workers on instance x_n, computed as follows:

$$b_{nr}^1 = \frac{\sum_{r'=1 \wedge r' \neq r}^{R} I(l_n^r = l_n^{r'})}{\sum_{r'=1 \wedge r' \neq r}^{R} I(l_n^{r'} \neq -1)} \quad (6)$$

where $I(\cdot)$ denotes the indicator function, outputting 1 if the condition in parentheses holds true and 0 otherwise. A practical edge case arises in real-world crowdsourcing when an instance receives labels from only a single worker (i.e., no other workers $u_{r'}$ exist). In this scenario, the label agreement between worker u_r and others on that instance is directly set to 0.

Analogously, for each neighbor within x_n's K-nearest neighbors, we compute worker u_r's label agreement with other workers using the identical method. This paper defines b_{nr}^2 as the weighted mean of u_r's label agreement across these K neighbors, calculated as follows:

$$b_{nr}^{2} = \frac{1}{Z}\sum_{k=1}^{K} \frac{\sum_{r'=1\wedge r'\neq r}^{R} I(l_{nkr}^{r}=l_{nkr}^{r'})}{\sum_{r'=1\wedge r'\neq r}^{R} I(l_{nkr}^{r'}\neq -1)} \cdot w_{nkr} \quad (7)$$

where Z is a normalization constant that maps b_{nr}^{2} to the interval [0, 1], computed as the sum of weights over the K neighbors, i.e.,$Z=\sum_{k=1}^{K} w_{nkr}$. Here, $l_{nkr}^{r}$and $l_{nkr}^{r'}$ denote the labels assigned by workers u_r and $u_{r'}$ respectively to the k-th nearest neighbor x_{nkr}.

This approach holistically integrates worker u_r's label agreement with peers on both instance x_n and its neighbor instances. The final labeling bias b_{nr} is then defined by combining $b_{nr}^{1}$and b_{nr}^{2}:

$$b_{nr} = \frac{b_{nr}^{1}+b_{nr}^{2}}{2} \quad (8)$$

4.4 Variance Computation

The variance v_{nr} quantifies the stability of worker u_r's labeling performance on instances similar to x_n. We posit that workers with stable labeling capabilities should provide consistent labels across similar instances. Consequently, higher consistency between u_r's label on x_n and its neighbors corresponds to higher v_{nr}. The computation of v_{nr} proceeds as follows:

$$v_{nr} = \frac{1}{Z}\sum_{k=1}^{K} I(l_{n}^{r} = l_{nkr}^{r}) \cdot w_{nkr} \quad (9)$$

where w_{nkr} denotes the weight of neighbor x_{nkr}, consistent with its definition in Formula (7). At this stage, we obtain worker u_r's labeling $b_{nr}$and labeling variance v_{nr} on instance x_n. The final labeling quality w_{nr} is then derived by integrating these two metrics using the following formula:

$$w_{nr} = \frac{b_{nr}+v_{nr}}{2} \quad (10)$$

Subsequently, using the computational approach described above, we derive the labeling quality w_{nr} for every worker u_r on each instance they have labeled. We then define the mean value of worker u_r's labeling quality across all instances they labeled as their overall reliability w_r. This reliability metric is formally defined as follows:

$$w_{r} = \sum_{n=1, x_n\in T^r}^{I} \frac{w_{nr}}{I} \quad (11)$$

where I denotes the number of all instances in T^r.

Algorithm1：Worker Reliability Acquisition
Input: Crowdsourced dataset $T=(X,U,L)$

Output: Final reliability w_r for each crowdsourced worker u_r

1: Using KNN algorithm, acquire all neighbors $\{x_{nkr}\}_{k=1}^{K}$ for instance x_n labeled by worker u_r

2: Compute labeling bias of u_r on x_n, $b_{nr}=(b_{nr}^1+b_{nr}^2)/2$

3: Compute labeling variance of u_r on x_n, $v_{nr}=\frac{1}{Z}\sum_{k=1}^{K} I(l_n^r=l_{nkr}^r)\cdot w_{nkr}$

4: repeat

5: for n←1 to N do

6: Compute b_{nr},v_{nr} and w_{nr} for all instances labeled by u_r

7: end for

8: Compute worker u_r's mean labeling quality across all labeled instances: $w_r=\sum\frac{w_{nr}}{I}$

9: repeat

10: for r←1 to R do

11: Compute average labeling quality $\{w_r\}_{r=1}^{R}$ for all workers

12:Obtain final reliability w_r for each crowdsourced worker

5 Graph Embedding Truth Inference Model with Worker Reliability

Having computed the final reliability for each worker through the aforementioned formulas, this section will first construct the "task-worker-label" graph structure, then utilize worker reliability to generate feature vectors for worker nodes and task nodes, and finally feed them into Graph Convolutional Networks (GCN) for truth inference.

5.1 "Task-Worker-Label" Graph Construction

As previously defined, each task node $x_n \in X$ in the crowdsourced dataset $T=\{X,U,L\}$ receives multiple labels from workers in U, where each label represents a pairwise relationship between a task and a worker. Leveraging these relationships, we construct a "task-worker" relational graph for every task node x_n, formally described as $G_n=(V_n,E_n)$, $V_n=\{x_n,u_r|<u_r,x_n,l_n^r>,r\in[1,R]\}$, and $E_n=\{x_n,u_r|<u_r,x_n,l_n^r>,r\in[1,R]\}$. A label l_n^r exists only when task x_n connects to worker u_r, thus it can be represented as an edge label. After transforming all triplets, we obtain a graph set $G=\{G_n=(V_n,E_n)|n=1,2,\dots,N\}$ .Subsequently, we compute the feature vector $\vec{\beta}_n$ for task nodes and feature vector $\vec{\alpha}_r$ for worker nodes. Figure. 3 illustrates the structure of the "task-worker-label" graph.

Task	Worker	Label
t_1	worker1	1
t_1	worker2	2
t_1	worker3	1
t_2	worker2	2
t_2	worker3	2
t_2	worker4	3
t_3	worker4	2
t_3	worker5	3

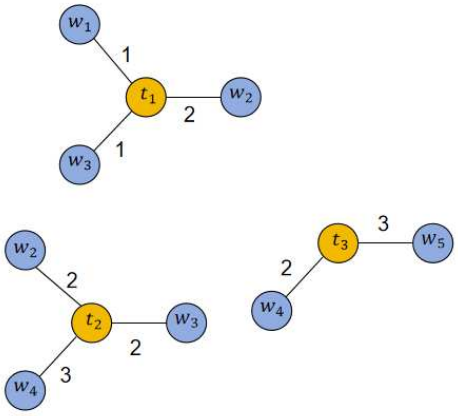

Fig 3 Example structure of the "Task-Worker-Label" graph

5.2 Graph Node Feature Vector Encoding

Upon obtaining the final reliability for each worker, we utilize this metric to generate feature vectors for worker nodes and task nodes. The feature vector for a task node should adhere to the following principle: it should represent the latent probability distribution of the node belonging to each possible class. Specifically, for a crowdsourcing task with K distinct label categories, the corresponding task node's feature vector is a K-dimensional vector where each element denotes the likelihood of the task node belonging to the class associated with that dimension. We let $\Theta_k^{n\prime}$ denotes the k-th dimension feature of the n-th task node, which computed as follows, where a_k represents the set of workers who assigned label k to task x_n:

$$\Theta_k^{n\prime} = \sum_{r \in a_k} w_r \tag{12}$$

Evidently, the result for $\Theta_k^{n\prime}$ is the sum of weights of all workers who assigned label k to instance x_n. In the graph, Θ_k^n represents the likelihood that instance x_n belongs to class k. After computing the feature values for all K dimensions, we obtain the feature vector for the task node:$\overrightarrow{\beta_n}$={$\Theta_1^{n\prime}, \ldots, \Theta_k^{n\prime}, \ldots, \Theta_K^{n\prime}$}.

The feature definition for worker nodes differs from that of task nodes. We posit that this vector should represent all potential label categories a worker may provide, while simultaneously reflecting the reliability levels of such labels. Therefore, we define each feature of worker u_r as follows:

$$\Theta_k^r = \begin{cases} w_r & k \in C_r \\ 0, & others \end{cases} \tag{13}$$

where C_r denotes the set of all classes for which worker u_r has provided labels. For example, consider a 5-class crowdsourcing task where a worker with reliability w_r=0.5 has labeled tasks belonging to classes 1, 2, 4, and 5. This worker's feature vector would be {0.5,0.5,0,0.5,0.5}. Thus, we obtain worker u_r's feature vector: $\overrightarrow{\alpha_r} = \{\Theta_1^r, \ldots, \Theta_k^r, \ldots, \Theta_K^r\}$.

After obtaining the feature vectors for task node x_n and its adjacent worker nodes u_r, we concatenate these vectors into a graph feature matrix for subsequent graph neural network processing. Since both task and worker node feature vectors are row vectors, this paper concatenates them using

the following formula:

$$X_n = concate\left(\overrightarrow{\beta_n}, \overrightarrow{\alpha_r}, \dots, \overrightarrow{\alpha_j}\right), u_r, \dots, u_j \in neighbor(x_n) \quad (14)$$

5.3 Truth Inference via Graph Embedding Model

Building upon the theoretical foundations in Sections 5.1 and 5.2, we optimize each graph $G_n \in G$ using graph embedding techniques. The optimized feature vectors undergo convolution, pooling, and activation operations to infer correct task labels. This paper designs an unsupervised model based on **Graph Autoencoders (GAE)** to refine and re-encode initial feature vectors of worker and task nodes, obtaining optimal feature representations.The final feature vectors are processed through a **Softmax operation** to compute the probability distribution of edge labels between worker and task nodes. These probabilities are propagated into the loss function, with gradients backpropagated for model optimization. The model integrates three core components: graph convolution for neighborhood aggregation, edge prediction for worker-task interaction modeling, and graph pooling for hierarchical abstraction – where graph convolution and edge prediction jointly optimize feature vectors while graph pooling infers the correct task labels.

5.3.1 Graph Convolution

The graph convolution component comprises two fundamental steps: neighborhood aggregation and feature transformation. Neighborhood aggregation synthesizes feature information from the target task node and its adjacent neighbor nodes. By generating intermediate node representations, this operation captures local topological relationships within the graph structure—constituting the core mechanism of graph convolution. Feature transformation then applies nonlinear transformations and dimensional adjustments to the aggregated features. Through multi-layer stacking, this process extracts hierarchical graph patterns, thereby enhancing the model's representational capacity.

Neighborhood Aggregation: We formally define the neighborhood aggregation operation as the sum of the feature vector of a task node and those of all its neighboring nodes. Furthermore, to account for node degree distribution during the aggregation of G_n, we apply normalization to the adjacency matrix. The final aggregation function is formally defined as:

$$aggre(X_n, \tilde{A}_n, \widetilde{D}_n) = \widetilde{D}_n^{-0.5} \tilde{A}_n \widetilde{D}_n^{-0.5} X_n \quad (15)$$

where$\widetilde{A_n}$,$\widetilde{D_n}$, and $X_n$denote the self-looped adjacency matrix, degree matrix, and feature matrix of nodes in G_n, respectively.

Feature Transformation: The aggregated feature matrix X_n' generated by the preceding aggregation function undergoes a high-dimensional mapping via the projection matrix Θ. This operation extracts deep relational information and yields an updated node feature matrix, which serves as input for subsequent processing stages. We formally define this computational procedure as follows:

$$H_n = ReLU(X_n'\Theta) \quad (16)$$

where the Rectified Linear Unit (ReLU) serves as the activation function, and the convolutional kernel Θ functions as the weight matrix. The outcome of this feature transformation is a new feature matrix H_n, obtained by mapping the aggregated feature matrix X_n' to a higher-dimensional space through the projection matrix Θ. Subsequently, the parameters of Θ are optimized during training to minimize the discrepancy between predictions and ground-truth values.

5.3.2 Edge Prediction

Following the acquisition of the updated feature matrix H_n via graph convolution, we perform label prediction for each edge in E_n. Specifically, we concatenate the feature row vectors of task nodes and worker nodes into $H_n^{'}$ and $H_n^{''}$ respectively. The edge prediction result is then computed as $\mathrm{sofmax}(H_n^{'} \cdot H_n^{''})$. This prediction is compared against the worker-provided label set $L=\{l_n^r | < u_r, x_n, l_n^r >\}$ to compute the loss function, which is then minimized by optimizing model parameters through backpropagation.

5.3.3 Graph Pooling

This section processes the optimized node feature vectors from graph convolution. We first map the node feature matrix H_n to a graph-level vector, which serves as the final feature representation of the target object for truth inference. A max graph pooling operation is employed, selecting the maximum value across all node feature vectors in H_n along each feature dimension to form a new feature vector. This vector is subsequently subjected to an activation function, as the prediction outcome for the task instance. The computational procedure for graph pooling is formalized below:

$$\max pool(H_n) = \max_{j=1,2,\ldots,Q} H_n^{(j)} \tag{17}$$

Following graph pooling, the resultant feature vector $(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)$ is obtained. The definitive truth inference outcome for the task instance is formally defined by the equation below:

$$t_n = \arg\max_n \{\varepsilon_n\} \tag{18}$$

Algorithm2：Graph Embedding Truth Inference Algorithm

Input: Graph set $G=\{G_n=(V_n,E_n)|n=1,2,...,N\}$; Graph feature vector set $G=\{G_n=(X_n,A_n)|n=1,2,...,N\}$; Convolution kernel Θ; label set $C=\{c_{mn}/(s_m,o_n,c_{mn})\}$

Output: Predicted truth values t_n for all nodes

repeat
  for n←1 to N do
    $X_n^{'} \leftarrow GAT(X_n,A_n)$
    $H_n \leftarrow conv(X_n^{'},\Theta)$
    Iterate over edge set E_n. Select corresponding task and worker node feature vectors

from H_n, concatenating them into feature matrices $H_n^{'}$ and $H_n^{''}$ respectively

6: Compute edge label predictions via $sofmax(H_n^{'} \cdot H_n^{''})$. Calculate loss according to label set $C=\{c_{mn}/\langle s_m, o_n, c_{mn}\rangle\}$

7: Backpropagate gradients to optimize model parameters

8: end for

9: until convergence

10: Obtain feature matrices $\{H_1, H_2, \ldots, H_n\}$ using optimized parameters

11: Derive task node truth values t_n via max pooling

6 Experiments and results

The proposed algorithm operates as an unsupervised truth inference approach. Experimental trials were conducted on a laptop with 16GB RAM, equipped with an NVIDIA GeForce GTX 1660 Ti GPU and an Intel® Core™ i7-9750H CPU. The GETI-WR implementation was developed in Python using the PyTorch framework (v2.1.4), executed under Windows 11 with Python 3.12.4 and CUDA 12.2. For the graph convolutional component, we set the maximum iterations to 150 and learning rate to 0.001. Owing to differing category counts between datasets, trainable weight matrix dimensions were initialized to D=128 and D=256 respectively.

6.1 Datasets

To validate algorithm performance, we employ two publicly available real-world crowdsourcing datasets—"Income" and "Leaves"—from the CEKA platform [26] for experimental evaluation. These datasets were selected based on a critical requirement: our proposed algorithm relies on task instance attribute features to compute Heterogeneous Euclidean-Overlap Metric (HEOM) [27] distances for K-Nearest Neighbors (KNN) search, and both datasets provide complete instance feature information.

The **"Income"** dataset represents a classical binary classification crowdsourcing benchmark. It comprises 600 instances described by 10 attributes containing both categorical and numerical features, annotated by 64 distinct crowd workers. This dataset contains 6,000 crowdsourced labels in total.

The **"Leaves"** dataset constitutes a classic multi-class crowdsourcing benchmark with six categories. It contains 384 instances characterized by 64 exclusively numerical features, annotated by 83 distinct crowd workers. In total, 3,840 crowdsourced labels were collected for this dataset. Key statistics of both datasets are summarized in Table I:

Table 2: Summary of Real-World Datasets

Dataset	Instance	Feature	Classes	Workers	Labels
Income	600	10	2	64	6000
Leaves	384	64	6	83	3840

Furthermore, both datasets include metrics for the average labeling quality of crowd workers. In the "Income" dataset, the distribution of worker labeling quality is predominantly concentrated between 0.5 and 0.9. Conversely, the "Leaves" dataset exhibits a comparatively lower average label quality of 0.51 – a characteristic attributable to its inherent complexity as a six-class classification task. These average quality metrics additionally serve as benchmark references for evaluating worker reliability calculations in our framework.

6.2 Comparison Algorithms

To rigorously evaluate the effectiveness of our proposed method, we compare GETI-WR against six state-of-the-art truth inference algorithms: Majority Voting (MV), CATD, CRH, ZenCrowd, GLAD, and TiReMGE. Descriptions and parameter configurations follow:

- *Majority Voting (MV)***:** The simplest truth inference approach, which directly assigns the most frequent label as the inferred truth value. We implement MV using existing GitHub repositories .
- *CATD* [17]**:** An optimization-based truth inference method proposed by Li et al. Building upon the PM framework, it incorporates the number of tasks completed by workers as an additional reliability metric. Key parameters include significance level α (set to 0.05) and iteration count (fixed at 100).
- *CRH* [16]**:** A truth discovery algorithm for multi-source heterogeneous data conflicts. It iteratively optimizes reliability-weighted aggregation of source information to estimate truth values.
- *ZenCrowd* [18]**:** A probabilistic graphical model (PGM)-based truth inference algorithm proposed by Gianluca et al. With no prior parameter requirements, we initialize worker weights at 0.8 and limit iterations to 20.
- *GLAD* [19]**:** Another PGM-based approach that jointly models worker reliability and task difficulty (unlike ZenCrowd's uniform difficulty assumption). We initialize task difficulty at 1.0 and worker quality at 0.7.
- *TiReMGE* [28]**:** A multi-view graph neural network method that models tasks, workers, and labels as a heterogeneous graph. It achieves enhanced truth discovery through joint embedding learning and dynamic reliability estimation.

6.3 Comparison Algorithms

- **Accuracy**

Accuracy constitutes a fundamental classification evaluation metric in machine learning, measuring the proportion of correctly predicted instances across the dataset. It quantifies global classification performance by calculating the ratio of correctly classified samples to the total sample size. For example, a binary classifier achieving 90 correct predictions out of 100 instances yields 90% accuracy. This metric offers advantages of computational simplicity and intuitive interpretability, particularly suited to balanced class distributions (e.g., near 1:1 positive-negative ratios). However, accuracy becomes unreliable under severe class imbalance scenarios.

- **F1-Score**

To address this limitation, we additionally employ the F1-score, the harmonic mean of precision and recall ,it can be calculated as:

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recal} \tag{19}$$

The harmonic mean mitigates bias toward either metric, making it particularly effective for imbalanced datasets. For instance, given precision=70% and recall=90%, the F1-score (78.8%) is significantly lower than the arithmetic mean (80%), highlighting the disparity between precision and recall. Consequently, we employ both accuracy and F1-score as final evaluation metrics.

6.4 Discussion

We comprehensively compared our proposed GETI-WR algorithm against various baseline methods on the aforementioned crowdsourcing datasets. Key findings are presented as follows: Figure 4 and Figure 5 demonstrate the accuracy and F1-score of different algorithms on the "Income" dataset respectively, while Figure 6 and Figure 7 present the corresponding metrics for the "Leaves" dataset.

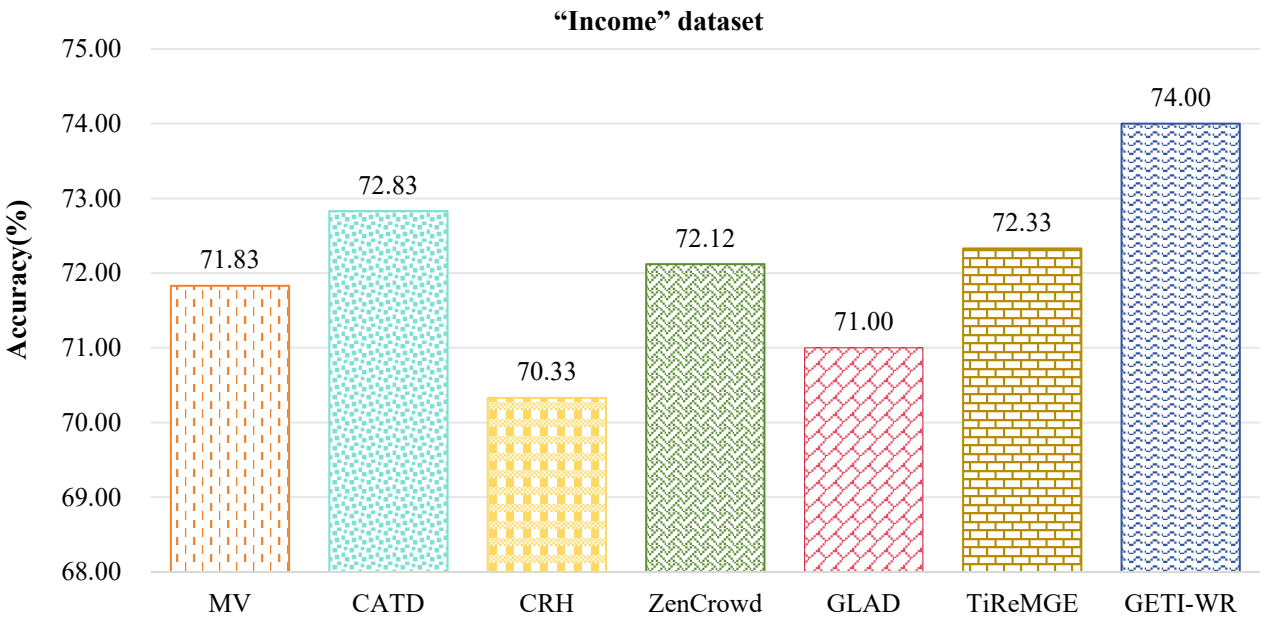

Fig. 4 Accuracy comparison across methods on "Income" dataset

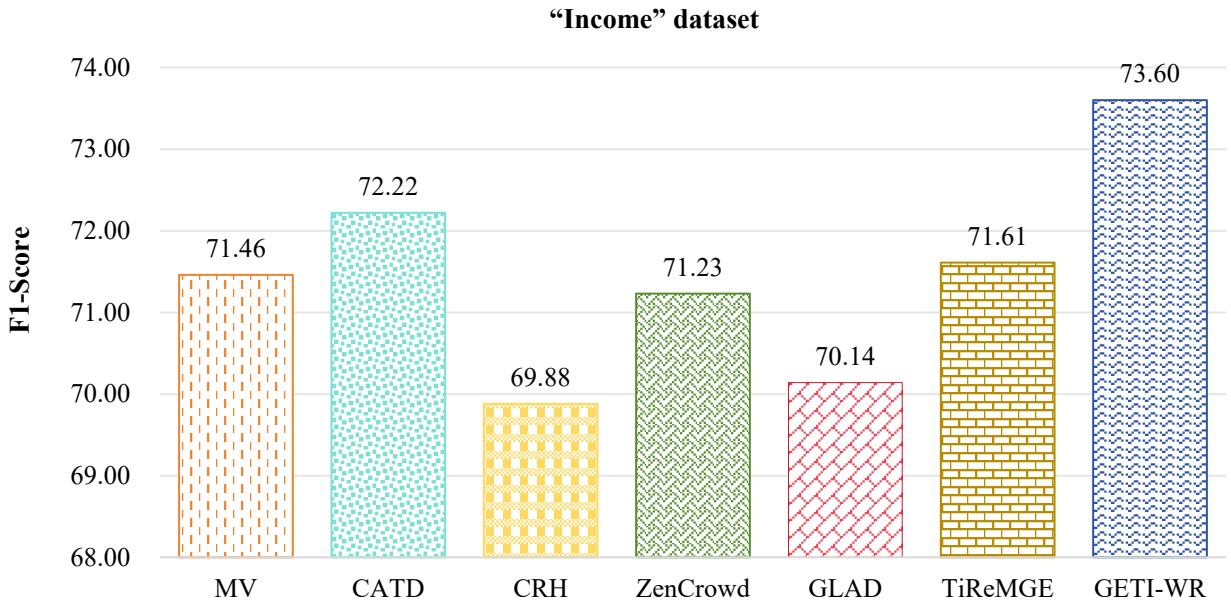

Fig. 5 F1-score comparison across methods on "Income" dataset

Experimental results in Figure 4 demonstrate that our proposed GETI-WR method outperforms all baselines (MV, CATD, CRH, ZenCrowd, GLAD, TiReMGE) on the "Income" dataset. Both accuracy and F1-score metrics confirm the superior truth inference performance of our approach. Specifically, GETI-WR achieves 74.00% accuracy – significantly higher than MV (71.83%), CATD (72.83%), CRH (70.33%), ZenCrowd (72.12%), GLAD (71.00%), and TiReMGE (72.33%). We categorize these methods into four classes: MV represents the simplest voting-based approach; CATD and CRH are optimization-based truth inference methods; ZenCrowd and GLAD employ probabilistic graphical models; TiReMGE and GETI-WR integrate deep neural networks. Notably, while MV demonstrates the simplest implementation, it does not yield the poorest performance.

CATD and CRH achieve average accuracy of 71.58%. This limitation may stem from inherent defects in reliability estimation through label-truth distance optimization and sensitivity to worker task volumes. ZenCrowd and GLAD attain lower average accuracy (70.69%) due to their dependency on prior parameter initialization, which significantly impacts results. The GitHub implementations used may not be optimally configured for our datasets, indicating limited robustness across domains.

TiReMGE and GETI-WR achieve 73.17% average accuracy, validating that deep neural architectures effectively capture latent instance-worker relationships. GETI-WR's peak accuracy (74.00%) arises from its unique incorporation of worker bias and variance metrics during labeling, which quantitatively assess reliability before transforming these features through graph embeddings for graph neural network-based inference.

Results in Figure 5 further confirm GETI-WR's superiority with 73.60% F1-score – 1.38% higher than second-ranked CATD, consistently validating our method's efficacy.

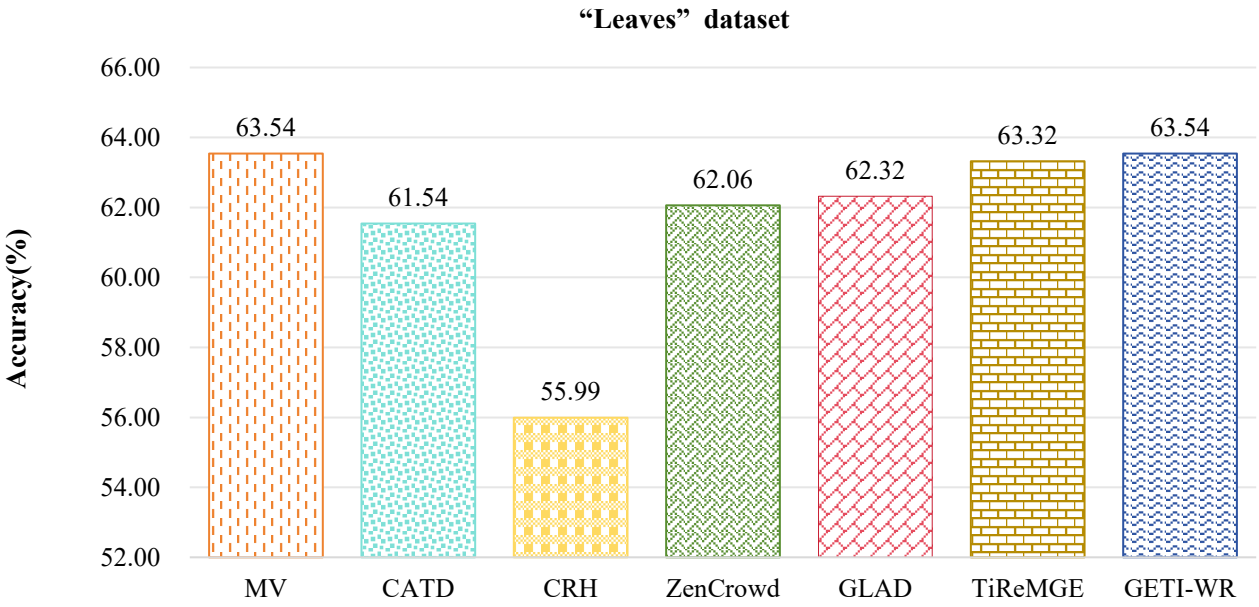

Fig. 6 Accuracy comparison across methods on "Leaves" dataset

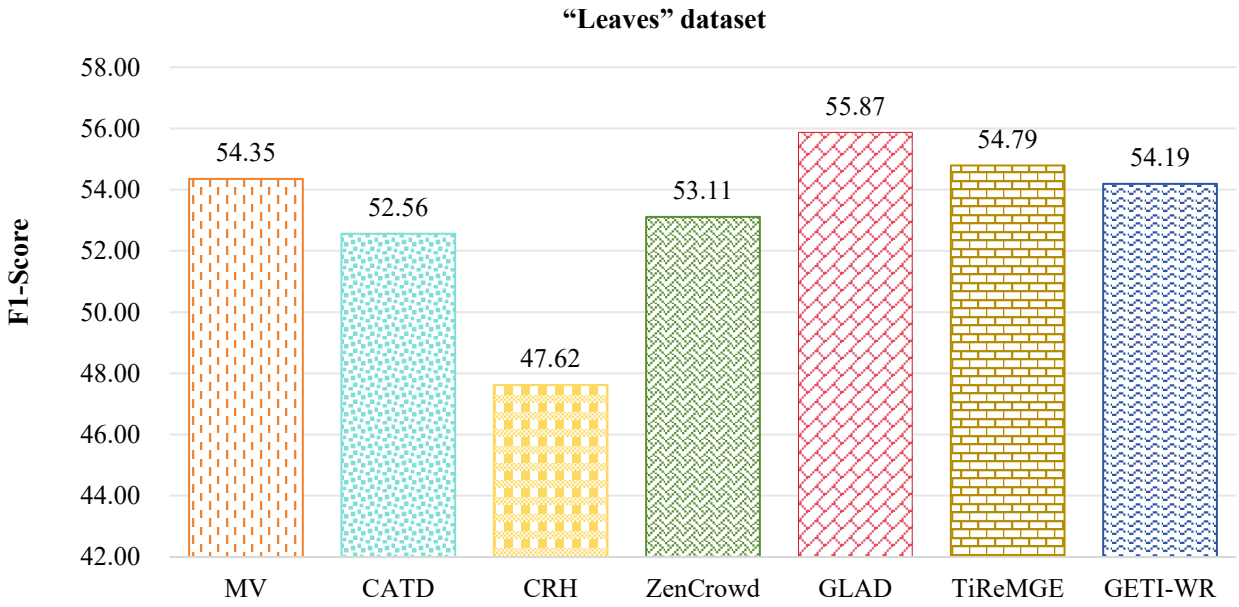

Fig.7 F1-score comparison across methods on "Leaves" dataset

Extending the same analytical approach to Figure 6, we observe that although performance on the multi-class dataset is inferior to that on the binary classification task, our proposed GETI-WR method still surpasses CATD, CRH, ZenCrowd, GLAD, and TiReMGE on the "Leaves" dataset while matching MV's accuracy. Specifically, GETI-WR achieves 63.54% accuracy – higher than CATD (61.54%), CRH (55.99%), ZenCrowd (62.06%), GLAD (62.32%), and TiReMGE (63.32%), and equivalent to MV (63.54%).

Multi-class classification tasks inherently present greater challenges than binary classification, as

crowd workers must distinguish between more categories, increasing the likelihood of confusion and mislabeling. Furthermore, in multi-class settings, workers typically annotate only a small subset of the dataset. As category counts increase, labels per class become sparser, hindering model capacity to learn discriminative class characteristics. Despite these inherent difficulties, GETI-WR demonstrates competitive performance against state-of-the-art baselines, further validating its efficacy and stability.

7 Conclusion

This paper proposes GETI-WR, a graph-embedded truth inference method based on worker reliability for crowdsourcing systems. We introduce bias and variance metrics to comprehensively evaluate worker reliability, demonstrating significant advantages over traditional initialization methods. Subsequently, worker reliability metrics are transformed into graph embeddings through graph embedding techniques and input into a graph neural network , reformulating the crowdsourcing problem as a graph node prediction task. Leveraging the structural perception capability of GCN to mine deep latent relationships between workers and instances, this approach enhances the efficiency and accuracy of truth inference.

Experimental results on two real-world datasets demonstrate that GETI-WR exhibits the following significant advantages: GETI-WR significantly outperforms other methods in accuracy; it also leads in F1-score, fully demonstrating the robustness and generalizability of our algorithm. We believe GETI-WR holds considerable promise in crowdsourcing and truth inference domains. However, due to the scarcity of datasets containing task instance features, this study only employed two real-world datasets to validate GETI-WR's effectiveness. In future work, we will seek more datasets with instance features to further verify GETI-WR's superiority.

Acknowledgements This paper is supported by the National Natural Science Foundation of China(72001164); the Fundamental Research Funds for the Central Universities (JB190609).

Author Contributions Wei Zong and Huize Feng wrote the main manuscript text. Huize Feng prepares the datasets. Huize Feng and ZongYao Nie designed the code and framework. All authors reviewed the manuscript.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Howe J. The rise of crowdsourcing [J]. Wired Magazine, 2006, 14(6):176-183.
2. Peng L, Liu Y, Wei Y, et al. Reliability-aware label distribution learning with attention-rectified for facial expression recognition[J]. Applied Intelligence, 2025, 55(1): 1-13.
3. Moslem Y, Haque R, Kelleher J D, et al. Adaptive machine translation with large language models[J]. ArXiv Preprint arXiv:2301.13294, 2023.
4. Dai W, Kong W, Shang T, et al. Guideline for Novel Fine-Grained Sentiment Annotation and Data Curation: A Case Study[J]. Expert Systems, 2025, 42(4): e70022.
5. Zhang J, Xu S, Sheng V S. Crowdmeta: Crowdsourcing truth inference with meta-Knowledge transfer[J]. Pattern Recognition, 2023, 140: 109525.

6. Yin L, Liu Y, Zhang W, Yu Y. Truth inference with a deep clustering-based aggregation model[J]. IEEE Access, 2020, 8:16662–16675.
7. Han T, Ding X, Fang Y. Multimodal information capture based truth inference network in crowdsourcing[J]. Expert Systems with Applications, 2025, 273(10): 126885.
8. Wang F, Liu H, Bi H, et al. A dataset for the validation of truth inference algorithms suitable for online deployment[J]. ArXiv Preprint arXiv:2403.08826, 2024.
9. Chen Z, Jiang L, Li C. Label augmented and weighted majority voting for crowdsourcing[J]. Information Sciences, 2022, 606: 397-409.
10. Wu H, Ma T, Wu L, et al. Exploiting heterogeneous graph neural networks with latent worker/task correlation information for label aggregation in crowdsourcing[J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2021, 16(2): 1-18.
11. Ma B, Li C, Jiang L. A novel ground truth inference algorithm based on instance similarity for crowdsourcing learning[J]. Applied Intelligence, 2022, 52(15): 17784-17796.
12. Zheng Y, Li G, Li Y, et al. Truth inference in crowdsourcing: Is the problem solved?[J]. Proceedings of the VLDB Endowment, 2017, 10(5): 541-552.
13. Suyal H, Singh A. Comparative Study of Truth Inferences Algorithms In Crowdsourcing[M]//Optimization Methods for Engineering Problems, Apple Academic Press, 2023: 99-111.
14. Li J, Jiang L, Zhang W. Label Consistency-Based Ground Truth Inference for Crowdsourcing[J]. IEEE Transactions on Neural Networks and Learning Systems, 2025, 36(5): 9408-9421.
15. Liu H, Liu J, Tang F, et al. Graph contrastive learning for truth inference[C]//2024 IEEE 40th International Conference on Data Engineering (ICDE), IEEE, 2024: 263-275.
16. Li Q, Li Y, Gao J, et al. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation[C]//Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 2014: 1187-1198.
17. Li Q, Li Y, Gao J, et al. A confidence-aware approach for truth discovery on long-tail data[J].Proceedings of the VLDB Endowment, 2014,8(4): 425-436.
18. Zhang J. Knowledge learning with crowdsourcing: A brief review and systematic perspective[J]. IEEE/CAA Journal of Automatica Sinica, 2022, 9(5): 749-762.
19. Zhang G, Wang N. Multi-Factor Influencing Truth Inference in Crowdsourcing[J]. Journal of Information Science & Engineering, 2021, 37(5): 1231-1246.
20. Yang Y, Bai Q, Liu Q. Modeling random guessing and task difficulty for truth inference in crowdsourcing[C]//Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems, 2019:2288-2290.
21. Fang Y, Sun H, Chen P, et al. Improving the quality of crowdsourced image labeling via label similarity[J]. Journal of Computer Science and Technology, 2017, 32(5): 877-889
22. Wu G, Zhou L, Xia J,et al. Crowdsourcing truth inference based on label confidence clustering[J].ACM Transactions on Knowledge Discovery from Data, 2023, 17(4): 1-20.
23. Zhou L, Zhuo X, Wu G, et al. Research on crowdsourcing truth inference method based on graph embedding[C]//2021 IEEE International Conference on Big Knowledge (ICBK). IEEE, 2021: 206-213.
24. Tao F, Jiang L, Li C. Label similarity-based weighted soft majority voting and pairing for crowdsourcing[J]. Knowledge and Information Systems, 2020, 62(14): 2521-2538.
25. Li J, Baba Y, Kashima H. Incorporating worker similarity for label aggregation in crowdsourcing[C]//International Conference on Artificial Neural Networks. Cham: Springer International Publishing, 2018: 596-606.

26. Zhang J, Sheng V S, Nicholson B A, et al. CEKA: A Tool for Mining the Wisdom of Crowds[J]. Journal of Machine Learning Research, 2015, 16(88): 2853-2858.
27. Krug M, Stockburger J. On stability issues of the HEOM method[J]. The European Physical Journal Special Topics, 2023, 232(20): 3219-3226.
28. Wu G, Zhuo X, Bao X, et al. Crowdsourcing truth inference via reliability-driven multi-view graph embedding[J]. ACM Transactions on Knowledge Discovery from Data, 2023, 17(5): 1-26.