SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

October 2024, Vol. 67, Iss. 10, 202102:1–202102:17
https://doi.org/10.1007/s11432-023-3854-7

FNNWV: farthest-nearest neighbor-based weighted voting for class-imbalanced crowdsourcing

Wenjun ZHANG[1], Liangxiao JIANG[1*], Ziqi CHEN[1] & Chaoqun LI[2,3]

[1]*School of Computer Science, China University of Geosciences, Wuhan* 430074*, China;*
[2]*Key Laboratory of Artificial Intelligence, Ministry of Education, Shanghai* 200240*, China;*
[3]*School of Mathematics and Physics, China University of Geosciences, Wuhan* 430074*, China*

Received 12 January 2023/Revised 30 March 2023/Accepted 29 June 2023/Published online 24 September 2024

Abstract In crowdsourcing scenarios, we can hire crowd workers to label crowdsourced tasks and then use label integration algorithms to infer the integrated label for each instance in the tasks. As more and more label integration algorithms are proposed, the performance of inference based only on the information of the inferred instance gradually converges. Recent algorithms attempt to exploit the information of the inferred instance's nearest neighbors to infer and achieve good performance. However, when crowdsourced tasks are class-imbalanced, negative instances are more easily to occur in the nearest neighbors because negative instances are the majority, and thus recent algorithms are more easily biased toward the negative class. To this end, in this paper, we propose a novel label integration algorithm called farthest-nearest neighbor-based weighted voting (FNNWV) for class-imbalanced crowdsourcing. Specifically, FNNWV considers the nearest neighbors to be more similar to the inferred instance and thus uses them to vote ayes in weighted voting. Yet at the same time, FNNWV considers the farthest neighbors to be more different from the inferred instance and thus uses them to vote nays in weighted voting. Since negative instances are easier to occur in both the nearest neighbors and the farthest neighbors, FNNWV weakens the effect of negative instances by voting ayes and nays. The experimental results on 22 simulated and one real-world crowdsourced datasets show that FNNWV significantly outperforms all the other state-of-the-art competitors.

Keywords crowdsourcing, label integration, nearest neighbor, farthest neighbor, weighted voting

1 Introduction

The classification task [1] is one of the most common tasks in supervised learning and is widely distributed in reality. To handle classification tasks, we usually train classifiers based on the labeled data, and then use classifiers to predict unknown labels for instances to be classified [2]. Therefore, the performance of classifiers is influenced by the quality of labeled data [3,4]. At the same time, with the development of deep learning, classifiers' demand for the scale and quality of labeled data has increased accordingly. However, it is hard to obtain high-quality labeled data on a large scale in reality [5–7]. At the root, traditional approaches usually rely on domain experts or well-trained workers to label classification tasks, which is expensive and time-consuming [8, 9].

Fortunately, the emergence of crowdsourcing has turned this trouble for the better [10, 11]. In crowdsourcing scenarios, through crowdsourcing platforms such as Crowdflower, Clickworker, and Amazon Mechanical Turk (AMT), employers can publish unlabeled classification tasks and hire crowd workers to label these tasks [12, 13]. Because crowd workers are easy to hire and inexpensive, crowdsourcing is more cost-effective and efficient than traditional approaches. However, since the expertise of crowd workers tends to be poorer than that of domain experts, labels assigned by crowd workers are usually noisy [14, 15]. To reduce the impact of noisy labels on the quality of trained classifiers, employers usually hire multiple crowd workers to label each instance in classification tasks, which is also known as repeated labeling [16]. After repeated labeling, employers can get a multiple noisy label set for each instance, and then they can infer an integrated label for each instance from its multiple noisy label set by label integration [17–19].

* Corresponding author (email: ljiang@cug.edu.cn)

Nowadays, a large number of label integration algorithms have been proposed. Among them, the simplest but usually effective algorithm is majority voting (MV) [16]. At first, similar to MV, label integration algorithms make inferences based only on the information of the inferred instance [20, 21]. However, due to the inferred instance itself only containing little information, the performance of inference gradually converges as more and more label integration algorithms are proposed. Recently, inspired by K-nearest neighbor (KNN), scholars have proposed some new label integration algorithms [22,23]. These algorithms exploit the information of the inferred instance's nearest neighbors to infer and achieve good performance. However, this leads to an unattended problem: when crowdsourced tasks are class-imbalanced, negative instances are more easily to occur in the nearest neighbors because negative instances are the majority, and thus recent algorithms are more easily biased toward the negative class.

To this end, we propose a novel label integration algorithm called farthest-nearest neighbor-based weighted voting (FNNWV) for class-imbalanced crowdsourcing. Specifically, FNNWV considers the nearest neighbors to be more similar to the inferred instance and thus uses the nearest neighbors to vote ayes in weighted voting. Yet at the same time, FNNWV considers the farthest neighbors to be more different from the inferred instance and thus uses the farthest neighbors to vote nays in weighted voting. Since negative instances are easier to occur in both the nearest neighbors and the farthest neighbors, FNNWV weakens the effect of negative instances by voting ayes and nays. To sum up, the main contributions of this paper include:

(1) We theoretically demonstrate the shortcomings of recent label integration algorithms in the face of class-imbalanced crowdsourcing and analyze the reasons for these shortcomings.

(2) We propose a novel neighbor query strategy called farthest-nearest neighbor (FNN) for class-imbalanced crowdsourcing. Different from KNN, FNN queries both the nearest neighbors and the farthest neighbors for each instance.

(3) We combine FNN and weighted voting to propose a novel label integration algorithm called FNNWV. FNNWV achieves good performance in class-imbalanced crowdsourcing by exploiting the information from both the nearest neighbors and the farthest neighbors when inferring the integrated label for each instance.

The rest of this paper is organized as follows. Section 2 introduces the related work on label integration. Section 3 demonstrates the shortcomings of recent label integration algorithms. Section 4 describes the proposed FNNWV in detail. Section 5 reports the experiments and results on simulated and real-world datasets. Section 6 summarizes this paper and outlines the research directions for future work.

2 Related work

In this section, we introduce existing state-of-the-art label integration algorithms and divide them into two categories based on whether they exploit the information of the inferred instance's nearest neighbors.

The first category of label integration algorithms does not exploit the information of the inferred instance's nearest neighbors. Li et al. [20] proposed iterative weighted majority voting (IWMV), which introduces workers' label quality for MV and iteratively estimates integrated labels and the label quality of workers. Karger et al. [24] proposed KOS (Karger, Oh, and Shah), which estimates integrated labels and the label quality of workers by belief propagation and low-rank matrix approximation. Zhang et al. [21] proposed a positive label frequency threshold (PLAT), which be used to handle the imbalanced labeling by modeling the decision thresholds. Yin et al. [25] proposed a label-aware autoencoder (LAA), which learns the relationship between multiple noisy label sets and integrated labels by minimizing the objective function of autoencoders. Tian et al. [26] proposed max-margin MV (M^3V), which is inspired by a support vector machine (SVM) and maximizes the margin directly to infer the most likely class. Li et al. [27] proposed a Bayesian algorithm called enhanced Bayesian classifier combination (EBCC), which learns the correlation between different workers by Bayesian inference. Tao et al. [28] proposed a labeling function weighted majority voting (LFWMV), which treats labels assigned by all workers as the target domain and labels assigned by each worker separately as the source domain and then learns the label quality of workers by multiple-domain adaptation. Sheng et al. [29] proposed four soft MV algorithms, which improve the model quality of MV using the label distribution generated by MV. Yang et al. [30] proposed decision tree-based weighted majority voting (DTWMV), which uses workers' labels to grow trees and then uses the depth of workers in trees to estimate the label quality of workers.

As more and more label integration algorithms are proposed, the performance of inference based only

on the information of the inferred instance gradually converges. Scholars have gradually realized that the inferred instance itself only contains little information. Inspired by KNN, they proposed the second category of label integration algorithms, which exploit the information of the inferred instance's nearest neighbors to infer and achieve good performance. Recently, Jiang et al. [23] proposed multiple noisy label distribution propagation (MNLDP). MNLDP first calculates the label distribution of each instance. Then, MNLDP finds the nearest neighbors for the inferred instance in the attribute space and optimizes the weights of the nearest neighbors of the inferred instance by the locally linear embedding. Finally, MNLDP uses label propagation to combine the label distributions of the inferred instance and its nearest neighbors and infers the integrated label for the inferred instance accordingly. Chen et al. [22] proposed label augmented and weighted majority voting (LAWMV). LAWMV first finds the nearest neighbors for the inferred instance in the attribute space and then calculates the weights of the nearest neighbors based on the distance and label similarity between the inferred instance and its nearest neighbors. Finally, LAWMV augments the noisy labels of the nearest neighbors to the multiple noisy label set of the inferred instance based on the weights and infers the integrated label by weighted voting.

The second category of algorithms significantly improves the performance of label integration compared to the first category of algorithms. However, the second category of algorithms still has some shortcomings. In Section 3, we will prove that these label integration algorithms that exploit the information of the inferred instance's nearest neighbors are more easily biased toward the negative class in class-imbalanced crowdsourcing.

3 Problem analysis

In this section, we first introduce some notations to define the class-imbalanced crowdsourcing. Then, we theoretically analyze the shortcomings of recent label integration algorithms when facing class-imbalanced crowdsourcing. Finally, we indicate the reasons why traditional class-imbalanced algorithms cannot be directly applied in class-imbalanced crowdsourcing.

3.1 Class-imbalanced crowdsourcing

In this paper, we only discuss the class imbalance of binary classification. Given a crowdsourced task consisting of a set of unlabeled instances $\{\boldsymbol{x}_i\}_{i=1}^N$, where each instance $\boldsymbol{x}_i$ has an unknown true label y_i and $y_i \in \{-1, +1\}$, N is the number of instances. For convenience, we let "-1" denote the negative class and "$+1$" denote the positive class. We can define the imbalance ratio τ as follows:

$$\tau = \frac{\sum_{i=1}^N \delta(y_i, -1)}{\sum_{i=1}^N \delta(y_i, +1)}, \tag{1}$$

where $\delta(\cdot)$ is an indicator function that outputs 1 if its two parameters are identical, and 0 otherwise. In class-imbalanced crowdsourcing, negative instances are usually the majority and positive instances are usually the minority. Therefore, when τ is significantly larger than 1, the crowdsourced task is class-imbalanced. Although τ is unknown due to y_i being unknown, class-imbalanced crowdsourced tasks naturally exist.

Further, a set of crowd workers $\{u_r\}_{r=1}^R$ can be hired to label the class-imbalanced crowdsourced task, where u_r is the r-th worker, and R is the number of workers. After repeated labeling, we can obtain a multiple noisy label set $\boldsymbol{L}_i = \{l_{ir}\}_{r=1}^R$ for each instance $\boldsymbol{x}_i$, where l_{ir} is the label of $\boldsymbol{x}_i$ assigned by the worker u_r and l_{ir} takes a value from the set $\{-1, 0, +1\}$. Here, $l_{ir} = 0$ denotes that u_r does not label $\boldsymbol{x}_i$. Finally, we can obtain a class-imbalanced crowdsourced dataset $D = \{(\boldsymbol{x}_i, \boldsymbol{L}_i)\}_{i=1}^N$. According to D, we can then infer an integrated label $\hat{y}_i$ for the instance $\boldsymbol{x}_i$ in the crowdsourced task by label integration algorithms, and $\hat{y}_i$ is expected to be identical to the unknown true label y_i.

3.2 Shortcomings of recent label integration algorithms

Now, we can analyze the prior probability that $\boldsymbol{x}_i$ belongs to each class. Given $\boldsymbol{x}_i$, we denote the prior probability that $\boldsymbol{x}_i$ belongs to the positive class by $p(+1|\boldsymbol{x}_i)$ and the prior probability that $\boldsymbol{x}_i$ belongs to the negative class by $p(-1|\boldsymbol{x}_i)$. According to the definition of τ, $p(+1|\boldsymbol{x}_i)$ and $p(-1|\boldsymbol{x}_i)$ should satisfy

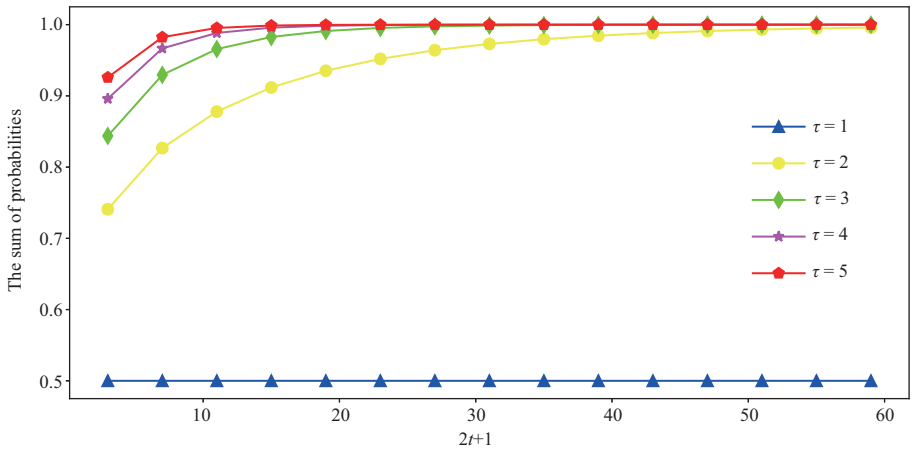

Figure 1 (Color online) Relationship between the sum of probabilities $\tilde{p}$, the ratio τ and the number of instances $2t+1$. When τ is fixed (larger than 1), the larger t is, the closer $\tilde{p}$ is to 1. When t is fixed, the larger τ is, the closer $\tilde{p}$ is to 1.

the following equations:

$$\begin{cases} p(+1|\boldsymbol{x}_i) + p(-1|\boldsymbol{x}_i) = 1, \\ \dfrac{p(-1|\boldsymbol{x}_i)}{p(+1|\boldsymbol{x}_i)} = \tau. \end{cases} \tag{2}$$

Ultimately, we can get

$$\begin{cases} p(+1|\boldsymbol{x}_i) = \dfrac{1}{1+\tau}, \\ p(-1|\boldsymbol{x}_i) = \dfrac{\tau}{1+\tau}. \end{cases} \tag{3}$$

Based on the previous discussion, due to $\boldsymbol{x}_i$ itself only containing little information, recent label integration algorithms attempt to exploit the information of $\boldsymbol{x}_i$'s nearest neighbors when inferring the integration label $\hat{y}_i$ for $\boldsymbol{x}_i$. We assume that a specific label integration algorithm exploits the information from $2t+1$ instances when inferring the integrated label for $\boldsymbol{x}_i$, and then the probability $\tilde{p}$ can be calculated using the Bernoulli model as follows:

$$\begin{aligned} \tilde{p} &= \sum_{i=0}^{t} \binom{2t+1}{i} p(+1|\boldsymbol{x}_i)^i * p(-1|\boldsymbol{x}_i)^{2t+1-i} \\ &= \sum_{i=0}^{t} \binom{2t+1}{i} \frac{\tau^{2t+1-i}}{(1+\tau)^{2t+1}}, \end{aligned} \tag{4}$$

where $\tilde{p}$ is the sum of the probabilities of having more negative instances than positive ones in $2t+1$ instances.

Now, according to (4), we can analyze the relationship of $\tilde{p}$ with τ and t, and we show the relationship in Figure 1. From Figure 1, we can obtain two conclusions: (i) when τ is fixed (larger than 1), the larger t is, the closer $\tilde{p}$ is to 1; (ii) when t is fixed, the larger τ is, the closer $\tilde{p}$ is to 1. These conclusions indicate that in class-imbalanced crowdsourcing, the larger τ, or the more instances exploited to infer the integrated label for $\boldsymbol{x}_i$, the more likely that there will be more negative instances than positive ones in exploited instances. Therefore, since recent label integration algorithms rarely consider class imbalance, they are more easily biased toward the negative class when inferring integrated label for $\boldsymbol{x}_i$.

3.3 Limitations of traditional class-imbalanced algorithms

Based on the above analysis, recent label integration algorithms have shortcomings when facing class-imbalanced crowdsourcing. To this end, a strategy that readily comes to mind is to introduce traditional class-imbalanced algorithms from supervised learning into recent algorithms to improve the performance of these algorithms in class-imbalanced crowdsourcing. Referring to [31], traditional class-imbalanced algorithms can be broadly classified into three categories, including undersampling, oversampling and

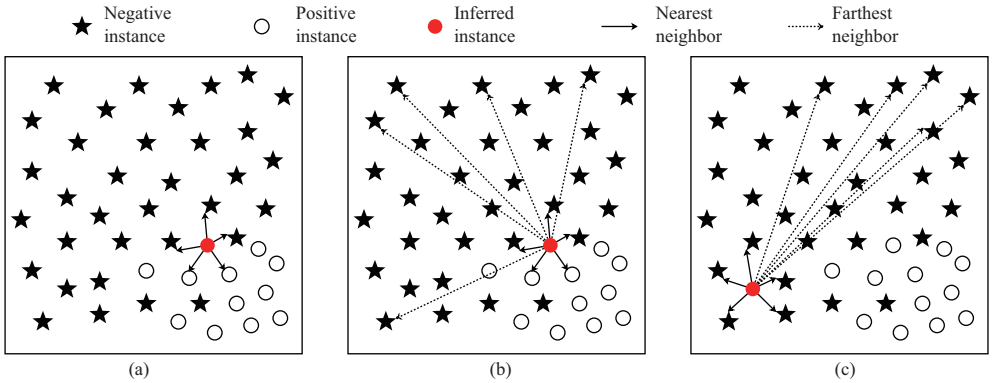

Figure 2 (Color online) Basic idea of FNNWV. (a) Exploiting the nearest neighbors directly without considering class imbalance leads to label integration algorithms that are more easily biased toward the negative class in class-imbalanced crowdsourcing (case1). (b) FNNWV considers the nearest neighbors to be more similar to the inferred instance and thus uses them to vote ayes in weighted voting. Meanwhile, FNNWV considers the farthest neighbors to be more different from the inferred instance and thus uses them to vote nays in weighted voting (case2). (c) In addition, some instances can be accurately inferred based only on the nearest neighbors, and FNNWV does not need to introduce the farthest neighbors for these instances (case3).

threshold-moving. All three categories of algorithms need to estimate τ based on training datasets to achieve rebalance.

However, in class-imbalanced crowdsourcing, we do not have accurately labeled training datasets for learning. Each instance in a crowdsourced dataset only contains a multiple noisy label set. This makes it difficult to estimate τ precisely. Therefore, most of the traditional class-imbalanced algorithms cannot be directly applied to class-imbalanced crowdsourcing. This again indicates the importance of designing effective label integration algorithms for class-imbalanced crowdsourcing.

4 Proposed algorithm

4.1 Motivation

Based on the above discussion, we want to propose a novel label integration algorithm that can be applied effectively to class-imbalanced crowdsourcing. For this purpose, the core problem we have to solve is how to ensure that our algorithm is sufficiently accurate and not biased toward the negative class.

At first, to obtain more valuable label information, inspired by [22], we still try to exploit the label information from neighbors of $\boldsymbol{x}_i$ when inferring the integrated label for $\boldsymbol{x}_i$. However, as we discussed in Section 3, exploiting the nearest neighbors directly without considering class imbalance leads to label integration algorithms that are more easily biased toward the negative class in class-imbalanced crowdsourcing. Specifically, we use Figure 2(a) to demonstrate this phenomenon. As shown in Figure 2(a), we try to find five nearest neighbors for unknown positive instance $\boldsymbol{x}_i$. Among these neighbors, there are two positive instances (circles) that are similar to $\boldsymbol{x}_i$ and three negative instances (pentagons) that are different from $\boldsymbol{x}_i$ but easier to occur. At this point, with MV, the positive class gets two votes and the negative class gets three votes, so $\boldsymbol{x}_i$ is inferred to be a negative instance.

Then, to avoid this phenomenon, we propose a new neighbor query strategy called FNN. We argue that if the assumption that closer instances are more similar holds, then it equally means that farther instances are more different. So in FNN, we query both the nearest neighbors and the farthest neighbors for $\boldsymbol{x}_i$. On the one hand, the nearest neighbors can be used to vote ayes to infer the class to which $\boldsymbol{x}_i$ is most likely to belong. On the other hand, the farthest neighbors can be used to vote nays to exclude the class to which $\boldsymbol{x}_i$ is least likely to belong. After that, we can combine the results of the nearest and farthest neighbors to infer the integrated label of $\boldsymbol{x}_i$. As shown in Figure 2(b), with FNN, we try to find five nearest neighbors and five farthest neighbors for unknown positive instance $\boldsymbol{x}_i$. Among the nearest neighbors, the positive class gets two votes and the negative class gets three votes. Among the farthest neighbors, the negative class loses five votes. In the end, the positive class gets two votes and the negative class gets minus two votes, so $\boldsymbol{x}_i$ is inferred to be a positive instance. Notably, negative instances are easier to occur in both the nearest neighbors and the farthest neighbors. Since the nearest neighbors vote ayes and the farthest neighbors vote nays, this weakens the effect of

negative instances in weighted voting.

In addition, not all inferred instances are as shown in Figure 2(b), so we also considered some other cases. Among these possible cases, the one shown in Figure 2(c) is a special case. In this case, the five nearest neighbors and the five farthest neighbors of $\boldsymbol{x}_i$ are all negative instances. So, in the end, the positive class gets zero votes and the negative class gets zero votes, which makes it difficult to infer the integrated label of $\boldsymbol{x}_i$. Therefore, to avoid this case, we first infer the integrated label for $\boldsymbol{x}_i$ with the nearest neighbors. If the votes are significantly different between the positive and negative classes as in Figure 2(c), this indicates that the integrated label can be accurately inferred based on the nearest neighbors only, and thus the farthest neighbors are not introduced. Conversely, if the votes are closer between the positive and negative classes as in Figure 2(b), this indicates that the integrated label cannot be accurately inferred based on the nearest neighbors only, and then we introduce the farthest neighbors.

Finally, combining all the above analyses, we design a new label integration algorithm called FNNWV for class-imbalanced crowdsourcing. In FNNWV, we design opposite weighting strategies for the nearest and farthest neighbors to further ensure the difference between them.

4.2 FNN

Given a crowdsourced dataset $D = \{(\boldsymbol{x}_i, \boldsymbol{L}_i)\}_{i=1}^{N}$, assume that the inferred instance $\boldsymbol{x}_i$ can be represented as $\{x_{i1}, \ldots, x_{im}, \ldots, x_{iM}\}$, where M denotes the attribute dimension and x_{im} denotes the m-th attribute value of $\boldsymbol{x}_i$. First, we need to calculate the distance between instances two by two and sort the distances. In this paper, the distance metric we use is the heterogeneous Euclidean-overlap metric (HEOM) [32]. HEOM uses normalized Euclidean distance for numerical attributes and overlap distance for nominal attributes. The distance between two instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is given as

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{m=1}^{M} d_m(\boldsymbol{x}_i, \boldsymbol{x}_j)^2}, \tag{5}$$

where $d_m(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denotes the distance between two instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on their m-th attribute (A_m), which can be calculated as follows:

$$d_m(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} 1, & \text{if } x_{im} \text{ or } x_{jm} \text{ is missing, else} \\ 1 - \delta(x_{im}, x_{jm}), & \text{if } A_m \text{ is nominal, else} \\ \dfrac{|x_{im} - x_{jm}|}{\max_m - \min_m}, & \end{cases} \tag{6}$$

where $\max_m - \min_m$ is used to normalize Euclidean distance, $\max_m$ and $\min_m$ denotes the maximum and minimum values of A_m observed from D, respectively.

Then, the nearest neighbors of $\boldsymbol{x}_i$ is denoted as $\{\boldsymbol{x}_{i1}^{\Theta}, \ldots, \boldsymbol{x}_{ik}^{\Theta}, \ldots, \boldsymbol{x}_{iK}^{\Theta}\}$, where the indicator Θ denotes a nearest neighbor, $\boldsymbol{x}_{ik}^{\Theta}$ denotes the k-th nearest neighbor of $\boldsymbol{x}_i$, K denotes the number of the nearest neighbors. Similarly, the farthest neighbors of $\boldsymbol{x}_i$ is denoted as $\{\boldsymbol{x}_{i1}^{\Phi}, \ldots, \boldsymbol{x}_{ik}^{\Phi}, \ldots, \boldsymbol{x}_{iK}^{\Phi}\}$ and the indicator Φ denotes a farthest neighbor. In this paper, the number of nearest neighbors and the number of farthest neighbors are both K. We set K equal to $0.25 \times N$, which remains the same as in [22]. Compared to the traditional KNN, FNN takes a larger value of K. This is because, in traditional classification datasets, we can obtain the true label of each instance in the training set, and thus each neighbor in KNN can provide accurate labeling information. However, in class-imbalanced crowdsourcing datasets, the true label of each instance is unknown. FNN can only obtain the multiple noisy label set of each neighbor. Therefore, FNN needs more neighbors to make inferences to weaken the effect of noise. Of course, when K is too large, it inevitably leads to some unimportant neighbors selected by FNN. To this end, FNNWV weights each neighbor based on the distance and the label similarity to ensure that less important neighbors receive a smaller weight in the voting.

4.3 FNNWV

In FNNWV, at first, to ensure the difference between the nearest and farthest neighbors, we weighted them using the opposite strategy. To a nearest neighbor, the closer it is to $\boldsymbol{x}_i$ or the more similar its multiple noisy label set is to that of $\boldsymbol{x}_i$, the greater its weight should be. But to a farthest neighbor, the

farther it is to $\boldsymbol{x}_i$ or the more different its multiple noisy label set is to that of $\boldsymbol{x}_i$, the greater its weight should be. Therefore, we calculate the weight w_{ik}^{Θ} of the nearest neighbor $\boldsymbol{x}_{ik}^{\Theta}$ as follows:

$$w_{ik}^{\Theta} = \frac{w_{ik1}^{\Theta} + w_{ik2}^{\Theta}}{2}, \tag{7}$$

where w_{ik1}^{Θ} denotes the weight calculated based on the distance and w_{ik2}^{Θ} denotes the weight calculated based on the similarity of the multiple noisy label set. w_{ik1}^{Θ} and w_{ik2}^{Θ} are calculated as follows:

$$w_{ik1}^{\Theta} = 1 - \frac{d\big(\boldsymbol{x}_i, \boldsymbol{x}_{ik}^{\Theta}\big)}{d\big(\boldsymbol{x}_i, \boldsymbol{x}_{iK}^{\Theta}\big)}, \tag{8}$$

$$w_{ik2}^{\Theta} = \begin{cases} 0, & \text{if } \sum_{r=1}^{R}(l_{ir} \neq 0 \wedge l_{ikr}^{\Theta} \neq 0) = 0, \\ \dfrac{\sum_{r=1}^{R} \delta\left(l_{ir}, l_{ikr}^{\Theta}\right)}{\sum_{r=1}^{R}(l_{ir} \neq 0 \wedge l_{ikr}^{\Theta} \neq 0)}, & \text{otherwise}, \end{cases} \tag{9}$$

where l_{ikr}^{Θ} denotes the label assigned by worker u_r for the nearest neighbor $\boldsymbol{x}_{ik}^{\Theta}$. From (8), it is clear that the smaller the distance between $\boldsymbol{x}_{ik}^{\Theta}$ and $\boldsymbol{x}_i$, the larger the weight of $\boldsymbol{x}_{ik}^{\Theta}$. From (9), it is clear that the more similar multiple noisy label sets of $\boldsymbol{x}_{ik}^{\Theta}$ and $\boldsymbol{x}_i$, the larger the weight of $\boldsymbol{x}_{ik}^{\Theta}$. Conversely, the weight w_{ik}^{Φ} of $\boldsymbol{x}_{ik}^{\Phi}$ can be calculated as follows:

$$w_{ik}^{\Phi} = \frac{w_{ik1}^{\Phi} + w_{ik2}^{\Phi}}{2}, \tag{10}$$

where w_{ik1}^{Φ} and w_{ik2}^{Φ} are calculated as follows:

$$w_{ik1}^{\Phi} = \frac{d\big(\boldsymbol{x}_i, \boldsymbol{x}_{ik}^{\Phi}\big)}{d\big(\boldsymbol{x}_i, \boldsymbol{x}_{i1}^{\Phi}\big)}, \tag{11}$$

$$w_{ik2}^{\Phi} = \begin{cases} 0, & \text{if } \sum_{r=1}^{R}(l_{ir} \neq 0 \wedge l_{ikr}^{\Phi} \neq 0) = 0, \\ 1 - \dfrac{\sum_{r=1}^{R} \delta\left(l_{ir}, l_{ikr}^{\Phi}\right)}{\sum_{r=1}^{R}(l_{ir} \neq 0 \wedge l_{ikr}^{\Phi} \neq 0)}, & \text{otherwise}. \end{cases} \tag{12}$$

From (11), it is clear that the larger the distance between $\boldsymbol{x}_{ik}^{\Phi}$ and $\boldsymbol{x}_i$, the larger the weight of $\boldsymbol{x}_{ik}^{\Phi}$. From (12), it is clear that the more different multiple noisy label sets of $\boldsymbol{x}_{ik}^{\Phi}$ and $\boldsymbol{x}_i$, the larger the weight of $\boldsymbol{x}_{ik}^{\Theta}$.

Then, we exploit the nearest neighbors to complete the weighted voting and calculate the normalized difference $nd_{\boldsymbol{x}_i}$ in votes between the positive and negative classes as follows:

$$nd_{\boldsymbol{x}_i} = \frac{\left|\sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta\left(l_{ikr}^{\Theta}, +1\right) - \sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta\left(l_{ikr}^{\Theta}, -1\right)\right|}{\sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta\left(l_{ikr}^{\Theta}, +1\right) + \sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta\left(l_{ikr}^{\Theta}, -1\right)}. \tag{13}$$

The smaller the normalized difference $nd_{\boldsymbol{x}_i}$, the more likely $\boldsymbol{x}_i$ is to satisfy the case shown in Figure 2(b). Conversely, it indicates that the more likely $\boldsymbol{x}_i$ is to satisfy the case shown in Figure 2(c). Therefore, in this paper, if the normalized difference $nd_{\boldsymbol{x}_i}$ is larger than the predefined threshold α, we skip the farthest neighbors and infer the integrated label for $\boldsymbol{x}_i$ as follows:

$$\hat{y}_i = \underset{c \in \{-1,+1\}}{\arg\max} \sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta(l_{ikr}^{\Theta}, c). \tag{14}$$

Otherwise, we need to introduce the farthest neighbors to infer the integrated label as follows:

$$\hat{y}_i = \underset{c \in \{-1,+1\}}{\arg\max} \sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Theta} * \delta(l_{ikr}^{\Theta}, c) - \sum_{k=1}^{K}\sum_{r=1}^{R} w_{ik}^{\Phi} * \delta(l_{ikr}^{\Phi}, c). \tag{15}$$

To summarize the above steps, the overall framework of FNNWV is defined in Figure 3. Based on this overall framework, we design a more detailed algorithmic process for FNNWV, which is summarized in Algorithm 1.

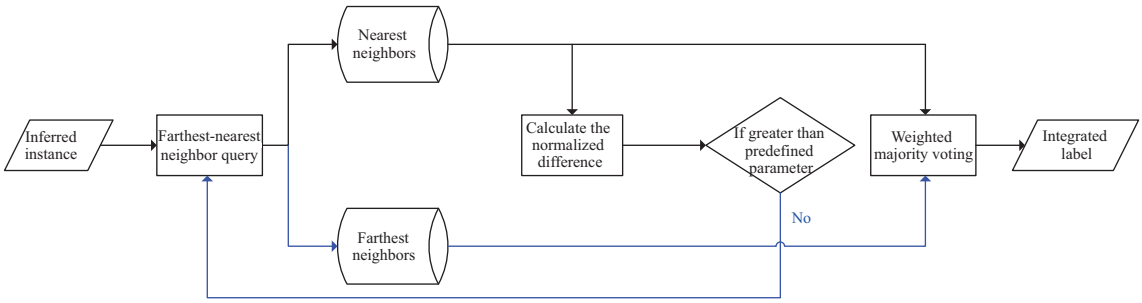

Figure 3 (Color online) Overall framework of FNNWV. Given an inferred instance, FNNWV first queries its nearest neighbors based on distance. Then FNNWV determines whether the normalized difference calculated according to the nearest neighbors exceeds the predefined parameter. If not exceeded, FNNWV queries the farthest neighbor of the inferred instance. Finally, according to queried neighbors, FNNWV infers the integrated label of the inferred instance by weighted MV.

Algorithm 1 FNNWV

Input: $D = \{(\boldsymbol{x}_i, \boldsymbol{L}_i)\}_{i=1}^{N}$ is a crowdsourced dataset; α is a predefined parameter.
Output: $\{\hat{y}_i\}_{i=1}^{N}$ are integrated labels.
1: **for** $i = 1$ to N **do**
2: **for** $j = 1$ to N **do**
3: Calculate the distance $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ by (5);
4: **end for**
5: Sort the distances in ascending order;
6: Find K nearest neighbors $\{\boldsymbol{x}_{ik}^{\Theta}\}_{k=1}^{K}$ for $\boldsymbol{x}_i$;
7: **for** $k = 1$ to K **do**
8: Calculate the weight w_{ik}^{Θ} for $\boldsymbol{x}_{ik}^{\Theta}$ by (7);
9: **end for**
10: Calculate the normalized difference $nd_{\boldsymbol{x}_i}$ for $\boldsymbol{x}_i$ by (13);
11: **if** $nd_{\boldsymbol{x}_i} > \alpha$ **then**
12: Infer the integrated label $\hat{y}_i$ for $\boldsymbol{x}_i$ by (14);
13: **else**
14: Find K farthest neighbors $\{\boldsymbol{x}_{ik}^{\Phi}\}_{k=1}^{K}$ for $\boldsymbol{x}_i$;
15: **for** $k = 1$ to K **do**
16: Calculate the weight w_{ik}^{Φ} for $\boldsymbol{x}_{ik}^{\Phi}$ by (10);
17: **end for**
18: Infer the integrated label $\hat{y}_i$ for $\boldsymbol{x}_i$ by (15);
19: **end if**
20: **end for**
21: Return $\{\hat{y}_i\}_{i=1}^{N}$.

4.4 Time complexity analysis

As shown in Algorithm 1, FNNWV can be divided into three main parts. First, lines 2–9 find the nearest neighbors for $\boldsymbol{x}_i$ by FNN and calculate weights for the nearest neighbors. Among them, the time complexity of calculating distances is $\mathrm{O}(MN)$. The time complexity of sorting the distances is $\mathrm{O}(N \log N)$. The time complexity of calculating weights is $O(KR)$. So the time complexity of lines 2–9 is $\mathrm{O}(N(M + \log N) + KR)$. Then, line 10 calculates the normalized difference $nd_{\boldsymbol{x}_i}$ for $\boldsymbol{x}_i$ and the time complexity of (13) is $O(KR)$. Finally, lines 11–19 infer the integrated label for $\boldsymbol{x}_i$ by weighted voting. Among them, line 12 infers the integrated label $\hat{y}_i$ for $\boldsymbol{x}_i$ by (14) and the time complexity is $O(KR)$. Lines 14–18 infer the integrated label $\hat{y}_i$ for $\boldsymbol{x}_i$ by (15) and the time complexity is $O(KR)$. Therefore, for each instance $\boldsymbol{x}_i$, the time complexity is $O(N(M + \log N) + KR)$, and thus the whole time complexity of FNNWV is $O(N^2(M + \log N) + NKR)$. If only the highest order terms are taken, the time complexity of FNNWV is $O(N^2(M + \log N))$.

5 Experimental results and analysis

The purpose of this section is to validate the effectiveness of our proposed FNNWV. For this purpose, we conduct a series of experiments on simulated and real-world class-imbalanced crowdsourced datasets on the crowd environment and its knowledge analysis (CEKA) [33] platform. In this section, we first illustrate our experimental settings, including the selection of comparison algorithms and the parameter settings of these algorithms. Then, we present the experimental results on simulated class-imbalanced crowdsourced datasets in terms of the F1-score. The F1-score reflects both the precision and recall of

algorithms in inferring integrated labels. Next, we validate the effectiveness of FNNWV again on a real-world class-imbalanced crowdsourced dataset. Furthermore, with this real-world dataset, we construct an ablation experiment to validate the effectiveness of each part of our FNNWV. Finally, we discuss the advantages and disadvantages of FNNWV in light of the full experimental results.

5.1 Experimental setup

In our experiments, we compare FNNWV with five state-of-the-art label integration algorithms, including MV, IWMV, PLAT, MNLDP, and LAWMV. Among them, MV is the baseline. IWMV and PLAT are the classical label integration algorithms, which do not exploit the information of the inferred instance's nearest neighbors. MNLDP and LAWMV are the recent label integration algorithms, which exploit the information of the inferred instance's nearest neighbors. Moreover, PLAT is similar to FNNWV in that it discusses imbalance, although it discusses imbalanced labeling, whereas FNNWV discusses naturally class-imbalanced crowdsourcing. The brief descriptions and parameter settings of all these six algorithms are introduced as follows:

• **MV** [16] is the simplest label integration algorithm, which votes equally using labels of each instance assigned by workers. We use the existing implementation of MV on the CEKA platform.

• **IWMV** [20] iteratively infers integrated labels of instances and estimates the label qualities of workers. We implement IWMV on the CEKA platform and set the number of iterations to 50, which is recommended in the corresponding paper.

• **PLAT** [21] counts the frequencies of positive and negative labels and estimates the decision threshold based on these frequencies. We use the existing implementation of PLAT on the CEKA platform.

• **MNLDP** [23] calculates the weights of nearest neighbors using locally linear embedding and iteratively absorbs the label distributions of neighbors through label propagation. We implement MNLDP on the CEKA platform and set the number of nearest neighbors, the hyper-parameter η, the number of iterations to 5, 0.5, 20, respectively, which are recommended in the corresponding paper.

• **LAWMV** [22] weights nearest neighbors of each instance and augments neighbors' labels to the multiple noisy label set of this instance. We implement LAWMV on the CEKA platform and set the hyper-parameter b to 0.5, which is recommended in the corresponding paper.

• **FNNWV** calculates the weights of the nearest and farthest neighbors of each instance, respectively, and then infers the integrated label for each instance by weighted voting. We implement FNNWV on the CEKA platform and set the predefined parameter α to 0.1.

5.2 Experiments on simulated datasets

Simulation process. In this subsection, we conduct our experiments on the whole 22 class-imbalanced datasets with the imbalance ratio between 1.5 and 9 published by the knowledge extraction based on evolutionary learning (KEEL) platform[1)]. Table 1 shows the details of these datasets that represent a wide range of domains and data characteristics. First, we hide the labels of each class-imbalanced dataset and treat it as a class-imbalanced crowdsourced task. Then, we simulate the crowdsourcing process to obtain the multiple noisy label sets of all instances. Here, we simulate five crowd workers to label the instances and randomly generate the label quality from a uniform distribution in the interval $[0.5, 0.8]$ for each worker. The label quality of a worker is the probability that this worker assigns a true label to an instance in the simulation process. Therefore, the label quality reflects a worker's labeling ability. Considering that in reality, the labeling ability of each worker is different, we randomly generate the label quality for each worker from a uniform distribution. After following this simulated process to obtain a crowdsourced dataset, we use chosen label integration algorithms to infer integrated labels for the instances in the crowdsourced dataset. Since the crowdsourced dataset is class-imbalanced, we use the F1-score as the evaluation metric [34]. Finally, we repeat the experiments ten times independently and report the averages of results.

Experimental results. Table 2 shows the detailed comparison results in terms of the F1-score. The highest value on each dataset is highlighted in bold and the averages of these algorithms are summarized at the bottom of the table. Based on these results, we perform the Wilcoxon signed-rank test [35] to further compare each pair of algorithms. Table 3 summarizes the Wilcoxon test results. In Table 3, the symbol • indicates that the algorithm in the row significantly outperforms the algorithm

1) https://sci2s.ugr.es/keel/imbalanced.php#sub10.

Table 1 Description of 22 class-imbalanced datasets

Dataset	#Attributes	#Instances	Imbalance ratio
ecoli-0_vs_1	7	220	1.86
ecoli1	7	336	3.36
ecoli2	7	336	5.46
ecoli3	7	336	8.6
glass-0-1-2-3_vs_4-5-6	9	214	3.2
glass0	9	214	2.06
glass1	9	214	1.82
glass6	9	214	6.38
haberman	3	306	2.78
iris0	4	150	2
new-thyroid1	5	215	5.14
new-thyroid2	5	215	5.14
page-blocks0	10	5472	8.79
pima	8	768	1.87
segment0	19	2308	6.02
vehicle0	18	846	3.25
vehicle1	18	846	2.9
vehicle2	18	846	2.88
vehicle3	18	846	2.99
wisconsin	9	683	1.86
yeast1	8	1484	2.46
yeast3	8	1484	8.1

in the corresponding column, and the symbol ∘ indicates the exact opposite of that indicated by the symbol •. The significance levels of the lower and upper diagonals are $\alpha = 0.05$ and $\alpha = 0.1$, respectively. These experimental results verify the effectiveness of our proposed FNNWV, and we can summarize the following highlights.

• The average F1-score of FNNWV on all datasets is 76.11%, which is much higher than those of MV (57.74%), IWMV (57.97%), PLAT (57.30%), MNLDP (72.23%), and LAWMV (68.10%). FNNWV achieves the highest F1-score, which indicates that FNNWV is more effective than other existing label integration algorithms in class-imbalanced crowdsourcing.

• The average F1-score of PLAT (57.30%) is close to that of MV (57.74%), which indicates that PLAT cannot handle class-imbalanced crowdsourced datasets very well. Meanwhile, PLAT performs well in imbalanced labeling but performs poorly in class-imbalanced crowdsourcing, which indicates the difference between imbalanced labeling and class-imbalanced crowdsourcing.

• Both MNLDP and LAWMV only exploit the information of the nearest neighbors, while FNNWV exploits the information from both the nearest neighbors and the farthest neighbors. Further, the average F1-score of MNLDP (72.23%) and LAWMV (68.10%) are higher than that of MV (57.74%) but still significantly lower than the average F1-score of FNNWV (76.11%), which indicates the effectiveness of utilizing the information from the farthest neighbors in class-imbalanced crowdsourcing.

• Based on the Wilcoxon test results, FNNWV significantly outperforms all of its competitors in terms of the F1-score, which strongly validates the effectiveness of FNNWV in class-imbalanced crowdsourcing.

Other distributions. In addition, to verify the effectiveness of FNNWV for different worker label quality distributions, we construct another group of experiments to compare the F1-score. In new experiments, we randomly generate the label quality from a normal distribution with $N(0.65, 0.15^2)$ for each worker. Similar to Tables 2 and 3, Table 4 shows the detailed comparison results for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV in terms of the F1-score, and Table 5 summarizes the Wilcoxon test results. The experimental results shown in Tables 4 and 5 also validate the effectiveness of FNNWV in class-imbalanced crowdsourcing.

5.3 Experiments on real-world dataset

Dataset selection. In this subsection, we conduct our experiments on a real-world crowdsourced dataset to further validate the effectiveness of FNNWV in real-world class-imbalanced crowdsourcing. We choose

Table 2 F1-score (%) comparisons on the uniform distribution for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV

Dataset	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
ecoli-0_vs_1	70.54	70.54	70.54	90.87	94.33	**94.85**
ecoli1	61.72	61.72	61.72	76.74	**82.17**	78.21
ecoli2	57.62	57.62	56.12	83.70	**85.43**	83.21
ecoli3	43.31	44.04	38.55	60.15	59.74	**76.30**
glass-0-1-2-3_vs_4-5-6	57.46	58.24	57.46	74.19	**82.69**	82.43
glass0	67.98	68.63	67.98	74.30	75.09	**75.43**
glass1	69.73	70.24	69.73	**74.30**	68.01	72.74
glass6	44.11	44.11	42.76	68.30	71.28	**76.04**
haberman	62.52	**63.01**	62.52	55.07	40.32	59.08
iris0	66.50	65.96	66.50	86.26	98.76	**99.08**
new-thyroid1	50.05	50.84	50.05	77.70	74.31	**84.66**
newthyroid2	48.34	48.42	48.34	74.87	69.52	**82.54**
page-blocks0	33.20	33.20	33.20	**51.41**	16.89	48.66
pima	69.76	69.76	69.76	68.36	65.56	**73.45**
segment0	52.90	52.90	50.75	76.22	82.07	**83.05**
vehicle0	62.84	63.75	62.84	**82.68**	70.38	79.78
vehicle1	59.78	59.78	59.78	62.76	50.22	**63.99**
vehicle2	59.10	59.10	59.10	**78.21**	62.42	65.83
vehicle3	55.33	55.33	55.33	**58.46**	50.10	57.62
wisconsin	68.85	69.02	68.85	85.17	**93.26**	92.40
yeast1	67.16	67.72	67.16	66.37	62.47	**73.77**
yeast3	41.53	41.53	41.53	63.06	43.17	**71.23**
Average	57.74	57.97	57.30	72.23	68.10	**76.11**

Table 3 F1-score (%) comparisons on the uniform distribution using Wilcoxon tests for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV

	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
MV	–	○		○	○	○
IWMV	•	–	•	○	○	○
PLAT		○	–	○	○	○
MNLDP	•	•	•	–		○
LAWMV	•	•	•		–	○
FNNWV	•	•	•	•	•	–

the "NER" dataset [36] as the experimental dataset. The "NER" dataset contains 5985 instances and 27990 multiple noisy labels (assigned by 47 workers from AMT). Among these instances, there are 1203 positive instances and 4782 negative instances, so the "NER" dataset naturally satisfies the characteristics of class-imbalanced crowdsourcing. In addition, because each instance in the "NER" dataset is a sentence, we preprocess the "NER" dataset by StringToWordVector to calculate distances easily. After preprocessing, each instance is represented by a text word vector with 2824 attributes.

F1-score. We first compare the experimental results of FNNWV and its competitors in terms of F1-score. Figure 4(a) shows the F1-score of six algorithms, including MV, IWMV, PLAT, MNLDP, LAWMV and FNNWV, on the "NER" dataset. We can see that FNNWV is significantly better than the other five competitors in terms of the F1-score. Specifically, the F1-score of FNNWV (79.41%) is much higher than those of MV (73.29%), IWMV (76.36%), PLAT (58.04%), MNLDP (61.62%), and LAWMV (78.27%). These experimental results validate the effectiveness of FNNWV in real-world class-imbalanced crowdsourcing.

Running time. Moreover, to verify the efficiency of FNNWV, we also conduct a group of experiments in terms of the running time. Our experiments are conducted on a Windows 10 machine with an AMD Athlon(tm) X4 860K Quad Core Processor @ 3.70 GHz and 16 GB of RAM. Figure 4(b) shows the running time of six algorithms, including MV, IWMV, PLAT, MNLDP, LAWMV, and FNNWV, on the "NER" dataset. We can see that MV and PLAT have the fastest running time, both under 1 s. MNLDP has the slowest running time (607.84 s) due to using the optimization tool to estimate the weights. The difference in running time between LAWMV (306.63 s) and FNNWV (319.08 s) is not significant. These

Table 4 F1-score (%) comparisons on the normal distribution for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV

Dataset	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
ecoli-0_vs_1	64.63	64.82	64.63	85.70	94.22	**94.86**
ecoli1	65.48	65.83	65.48	82.14	**82.41**	80.44
ecoli2	52.75	53.65	50.36	76.58	**83.61**	76.62
ecoli3	33.96	34.27	33.96	50.80	43.11	**63.47**
glass-0-1-2-3_vs_4-5-6	59.71	60.64	59.71	80.16	83.16	**84.31**
glass0	70.17	70.52	70.17	75.49	**77.89**	75.29
glass1	68.84	69.17	68.84	**73.00**	71.08	70.07
glass6	45.39	45.39	45.39	70.76	74.04	**76.88**
haberman	60.86	**62.00**	60.86	52.89	35.15	59.23
iris0	69.63	69.95	69.63	92.60	99.80	**99.80**
new-thyroid1	49.15	49.33	47.44	73.20	72.54	**85.13**
newthyroid2	50.48	50.83	50.48	75.27	74.82	**82.99**
page-blocks0	40.07	40.07	40.07	**63.39**	12.70	46.26
pima	68.72	68.72	68.72	67.90	65.08	**74.31**
segment0	46.93	46.93	44.54	67.00	**75.59**	72.85
vehicle0	60.43	60.43	60.43	**78.78**	69.39	76.37
vehicle1	62.84	62.84	62.84	63.75	50.48	**67.48**
vehicle2	62.06	62.06	62.06	**85.09**	63.38	72.06
vehicle3	60.37	60.68	60.37	**61.75**	51.15	60.79
wisconsin	72.60	72.60	72.60	90.01	**94.23**	93.19
yeast1	65.30	65.30	65.30	64.89	61.03	**71.23**
yeast3	38.99	38.99	36.98	58.54	44.42	**65.99**
Average	57.70	57.96	57.31	72.26	67.24	**74.98**

Table 5 F1-score (%) comparisons on the normal distribution using Wilcoxon tests for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV

	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
MV	–	○		○	○	○
IWMV	●	–	●	○	○	○
PLAT		○	–	○	○	○
MNLDP	●	●	●	–		○
LAWMV	●	●	●		–	○
FNNWV	●	●	●	●	●	–

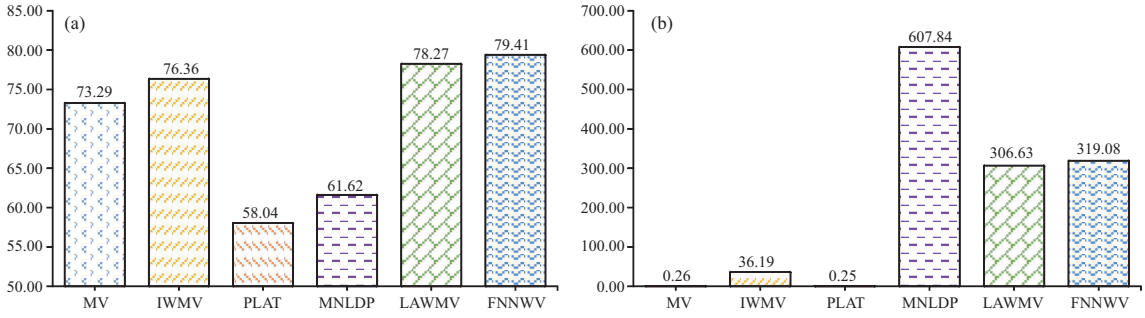

Figure 4 (Color online) (a) F1-score (%) and (b) running time (s) comparisons for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV on the "NER" dataset.

experimental results validate the efficiency of FNNWV, and FNNWV is on par with label integration algorithms that exploit the information of the inferred instance's nearest neighbors.

Ablation experiment. To evaluate the performance of each part in FNNWV, we compare FNNWV with its three variants. We denote its three variants as FNNWV-0, FNNWV-1, and FNNWV-2. FNNWV-0 does not exploit the information of the nearest and farthest neighbors, so FNNWV-0 is equivalent to MV. FNNWV-1 exploits only the information of the nearest neighbors for weighted voting, so FNNWV-1 is equivalent to LAWMV. FNNWV-2 exploits only the information of the farthest neighbors for weighted

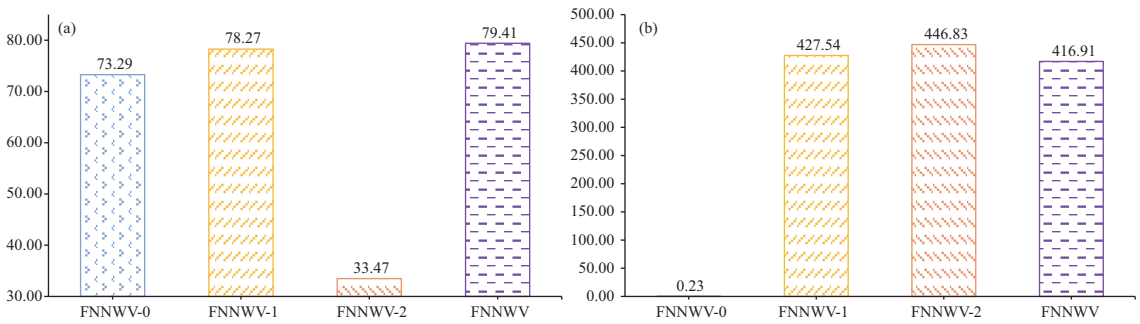

Figure 5 (Color online) (a) F1-score (%) and (b) running time (s) comparisons for FNNWV versus FNNWV-0, FNNWV-1, and FNNWV-2 on the "NER" dataset.

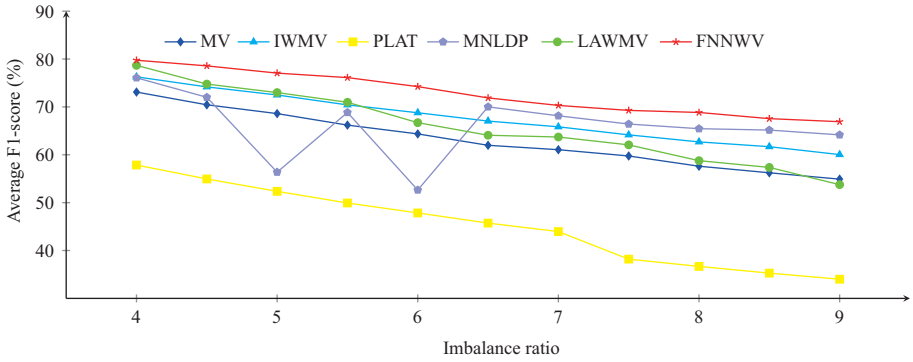

Figure 6 (Color online) Average F1-score (%) comparisons for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV when the imbalance ratio varies from 4 to 9 on the "NER" dataset.

voting. The experimental results in terms of the F1-score and the running time are shown in Figure 5. As seen from Figure 5(a), the F1-score of FNNWV-1 (78.27%) is higher than that of FNNWV-0 (73.29%), which indicates the effectiveness of exploiting the information from only the nearest neighbors. The F1-score of FNNWV-2 (33.47%) is much lower than that of FNNWV-0 (73.29%), which indicates that the integrated labels cannot be accurately inferred by exploiting the information from only the farthest neighbors. The F1-score of FNNWV (79.41%) is higher than its three variants, which again indicates the effectiveness of exploiting the information from both the nearest neighbors and the farthest neighbors. As seen from Figure 5(b), the running time of FNNWV-0 (0.23 s) is much lower than those of FNNWV-1 (427.54 s), FNNWV-2(446.83 s), and FNNWV(416.91 s). This indicates that the neighbor query is the main factor affecting the running time of FNNWV. The small difference in running time between FNNWV-1, FNNWV-2, and FNNWV indicates that whether or not the farthest neighbors are queried during the neighbor query has little effect on the running time of the algorithm.

Parameter sensitivity analysis. In addition to verifying the effectiveness and efficiency of FNNWV, the impact of the imbalance ratio on the performance of FNNWV has also received our attention. To continuously change and increase the imbalance ratio, we randomly remove a certain percentage of positive instances on the "NER" dataset. In this way, we observe the performance of FNNWV and its competitors on the "NER" dataset when the imbalance ratio takes values in {4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9}. To reduce the effect of randomness on the observed results, each experiment with different imbalance ratios is repeated 10 times. Figure 6 shows the average F1-score of FNNWV and its competitors on different imbalance ratios. From Figure 6, it can be found that the average F1-score of all algorithms gradually decreases as the imbalance ratio increases. But in contrast, FNNWV can always achieve the best performance with the same imbalance ratio conditions. This indicates that the effectiveness of FNNWV is not sensitive to changes in the imbalance ratio.

5.4 Analysis and discussion

In this paper, we focus on class-imbalanced crowdsourcing and then propose FNNWV, which exploits the information of the nearest and farthest neighbors for weighted voting. To verify the effectiveness of FNNWV, we construct extensive experiments based on the simulated and real-world class-imbalanced

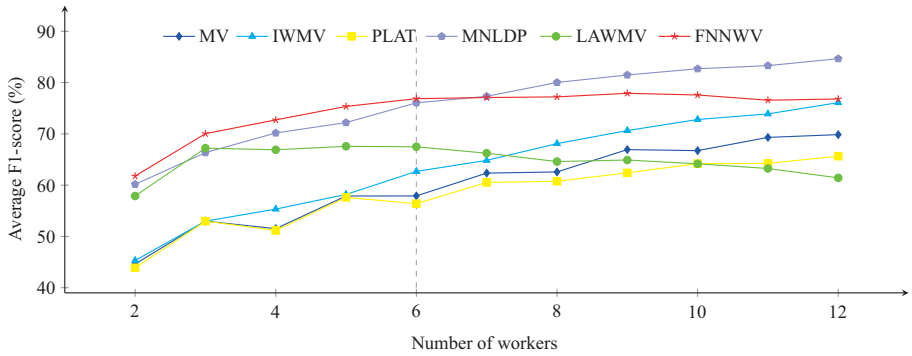

Figure 7 (Color online) Average F1-score (%) comparisons for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV when the number of workers varies from 2 to 12 on 22 simulated datasets.

Table 6 Description of 13 class-balanced datasets

Dataset	#Attributes	#Instances	Imbalance ratio
balance-scale_0_vs_2	4	576	1
credit-a	15	690	1.25
heart-c_0_vs_1	13	303	1.2
heart-statlog	13	270	1.25
iris_0_vs_1	4	100	1
kr-vs-kp	36	3196	1.09
lymph_1_vs_2	18	142	1.33
mushroom	22	8124	1.07
segment_0_vs_1	19	660	1
sonar	60	208	1.14
vehicle_0_vs_1	18	429	1.02
vowel_0_vs_1	13	180	1
waveform_0_vs_1	40	3345	1.02

crowdsourcing. The above experimental results preliminarily verify the effectiveness of FNNWV. In this subsection, we will further analyze and discuss the advantages and disadvantages of FNNWV.

Advantages. Experimental results on both simulated experiments and real-world experiments show that our FNNWV can achieve better performance in class-imbalance crowdsourcing compared to existing algorithms. This means that in the future, FNNWV is a better choice if we face a naively class-imbalanced crowdsourced dataset. In addition, compared to algorithms that only exploit the information of the nearest neighbors (MNLDP and LAWMV), the distances need to be sorted regardless of whether the information of the farthest neighbors is exploited. Therefore, FNNWV does not significantly increase the time complexity of the algorithm. Currently, FNNWV only briefly discusses the performance of exploiting the information from both the nearest neighbors and the farthest neighbors, which means that FNNWV is very scalable and can be combined with existing excellent algorithms like MNLDP in the future.

In the above simulation experiments, the number of workers is fixed at five. To observe the relationship between the number of workers and the performance of different label integration algorithms in class-imbalanced crowdsourcing, we construct a new group of experiments on 22 simulated datasets. In this group of experiments, the number of workers varies from 2 to 12. Except for the number of workers, the other experimental settings remain the same as in Simulation process. Figure 7 shows the average F1-score of FNNWV and its competitors on 22 simulated datasets. From Figure 7, it can be seen that when the number of workers does not exceed six, FNNWV can achieve the highest average F1-score. This indicates that FNNWV is more effective in class-imbalanced crowdsourcing when the number of workers is small and hence can reduce the cost of labeling. When the number of workers exceeds six, FNNWV can also achieve a higher average F1-score than MV, IWMV, PLAT, and LAWMV. And our FNNWV is not particularly sensitive to the number of workers.

In addition to class-imbalanced crowdsourcing, to further validate the effectiveness of our algorithm, we simultaneously observe the performance of FNNWV and its competitors on 13 class-balanced datasets. The description of these datasets is listed in Table 6. All of these datasets are published by the CEKA

Table 7 Integration accuracy (%) comparisons for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV on 13 class-balanced datasets

Dataset	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
balance-scale_0_vs_2	79.58	79.90	72.83	92.36	95.38	**95.49**
credit-a	77.25	77.88	66.13	81.77	**88.86**	87.12
heart-c_0_vs_1	75.18	75.18	75.18	78.84	86.04	**86.53**
heart-statlog	74.52	74.78	74.07	80.22	85.26	**85.85**
iris_0_vs_1	74.50	73.50	71.30	91.60	98.90	**99.80**
kr-vs-kp	79.02	79.02	79.02	**89.70**	87.66	87.31
lymph_1_vs_2	74.65	73.66	74.44	80.92	83.87	**84.15**
mushroom	77.66	77.66	77.66	**96.38**	93.50	90.87
segment_0_vs_1	73.30	73.30	68.36	85.55	98.74	**99.20**
sonar	69.28	69.28	65.53	**75.00**	72.84	73.17
vehicle_0_vs_1	76.97	**77.39**	70.65	72.28	76.71	75.50
vowel_0_vs_1	73.67	73.67	71.78	**87.17**	75.89	70.94
waveform_0_vs_1	78.45	78.45	76.35	89.90	**93.42**	90.74
Average	75.69	75.67	72.56	84.75	**87.47**	86.67

Table 8 Integration accuracy (%) comparisons using Wilcoxon tests for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV on 13 class-balanced datasets

	MV	IWMV	PLAT	MNLDP	LAWMV	FNNWV
MV	–		•	○	○	○
IWMV		–	•	○	○	○
PLAT	○	○	–	○	○	○
MNLDP	•	•	•	–	○	
LAWMV	•	•	•		–	
FNNWV	•	•	•			–

platform. Among these datasets, "credit-a", "heart-statlog", "kr-vs-kp", "mushroom", and "sonar" are five natural binary class-balanced datasets, and the other eight datasets are generated by sampling two classes of instances whose numbers are similar from multi-class datasets. For example, "balance-scale_0_vs_2" represents a binary class-balanced dataset generated by sampling all instances of class 0 and class 2 from the multi-class dataset "balance-scale". We use the same simulation process as in Simulation process for experiments. Similar to other experiments on class-balanced datasets [22], we use the integration accuracy as the evaluation metric. Table 7 shows the detailed comparison results for FNNWV versus MV, IWMV, PLAT, MNLDP, and LAWMV in terms of the integration accuracy, and Table 8 summarizes the Wilcoxon test results. The experimental results shown in Tables 7 and 8 validate the effectiveness of FNNWV in class-balanced crowdsourcing. Therefore, we can see that FNNWV can not only achieve better results in class-imbalanced crowdsourcing but also be on par with the state-of-the-art label integration algorithms in class-balanced crowdsourcing.

Disadvantages. According to the experimental results shown in Figure 5, we can find that inferring from only the farthest neighbors is not effective, but inferring from both the nearest neighbors and the farthest neighbors achieves good performance. However, as illustrated in Figure 2(c), inferring from both the nearest neighbors and the farthest neighbors is not always effective. Therefore, the conditions for using the farthest neighbors should be discussed. In this paper, to keep the algorithm simple, we use a predefined parameter α as a threshold to decide whether or not to use the farthest neighbors. Currently, we have empirically fixed α to 0.1. In the future, we can try to improve the performance of FNNWV by learning an adaptive α through optimization tools. Alternatively, we can develop more research on how to use the information from the farthest neighbors.

In addition, to simplify the analysis and discussion about class-imbalanced crowdsourcing, this paper focuses only on binary class-imbalanced crowdsourced tasks. When a class-imbalanced crowdsourced task is a multi-class classification task, more cases need to be taken into account. For example, in the case shown in Figure 8, a class-imbalanced crowdsourced task has three classes. As shown in Figure 8, the five nearest neighbors of unknown positive instance $\boldsymbol{x}_i$ contain two positive instances (circles) and three negative instances (pentagons), and the five farthest neighbors of $\boldsymbol{x}_i$ are all another type of negative instances (hexagons). So, in the end, the positive class denoted by a circle gets two votes, the negative

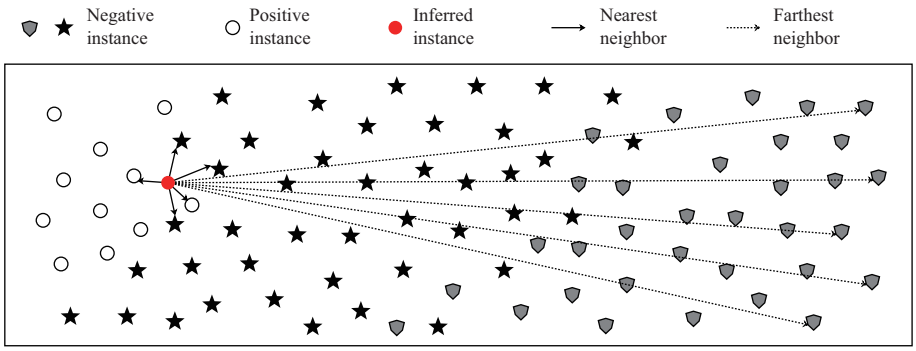

Figure 8 (Color online) Special case in multi-class class-imbalanced crowdsourcing.

class denoted by a pentagon gets three votes, and the negative class denoted by a hexagon gets minus five votes. It can be seen that, even if we introduce the farthest neighbors to multi-class class-imbalanced crowdsourced tasks, we may still incorrectly infer the integrated label of $\boldsymbol{x}_i$. Therefore, there are still many challenges in extending FNNWV to multi-class class-imbalanced crowdsourcing.

6 Conclusion and future work

In this paper, we propose a novel label integration algorithm called FNNWV for class-imbalanced crowdsourcing. We first demonstrate the shortcomings of recent algorithms in the face of class-imbalanced crowdsourcing. Then, we explain in detail our motivation and define the algorithmic process for FNNWV. Finally, we construct extensive experiments to verify the effectiveness of FNNWV. Experimental results show that FNNWV can achieve better performance in class-imbalanced crowdsourcing compared to existing algorithms.

As discussed in previous sections, when our FNNWV infers the integrated label for an instance, it is important to decide whether or not to use its farthest neighbors. Currently, we use a predefined parameter α as a threshold to decide, and α is empirically fixed to 0.1. We believe that the use of optimization tools to learn an adaptive α could further improve the performance of the current FNNWV. This is the main research direction for our future work. In addition, besides threshold learning, we will also develop more research on how to use the information from the farthest neighbors.

Acknowledgements This work was partially supported by National Natural Science Foundation of China (Grant No. 62276241) and Foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, China (Grant No. AI2022004).

References

1 Jiang L X, Zhang L G, Li C Q, et al. A correlation-based feature weighting filter for naive Bayes. IEEE Trans Knowl Data Eng, 2019, 31: 201–213
2 Zhang H, Jiang L X, Li C Q. Attribute augmented and weighted naive Bayes. Sci China Inf Sci, 2022, 65: 222101
3 Li S Y, Huang S J, Chen S. Crowdsourcing aggregation with deep Bayesian learning. Sci China Inf Sci, 2021, 64: 130104
4 Li C Q, Jiang L X, Xu W Q. Noise correction to improve data and model quality for crowdsourcing. Eng Appl Artif Intell, 2019, 82: 184–191
5 Yu G X, Tu J Z, Wang J, et al. Active multilabel crowd consensus. IEEE Trans Neural Netw Learn Syst, 2021, 32: 1448–1459
6 Zhang J. Knowledge learning with crowdsourcing: a brief review and systematic perspective. IEEE CAA J Autom Sin, 2022, 9: 749–762
7 Yang W J, Li C Q, Jiang L X. Learning from crowds with robust support vector machines. Sci China Inf Sci, 2023, 66: 132103
8 Xu W Q, Jiang L X, Li C Q. Improving data and model quality in crowdsourcing using cross-entropy-based noise correction. Inf Sci, 2021, 546: 803–814
9 Dong Y, Jiang L X, Li C Q. Improving data and model quality in crowdsourcing using co-training-based noise correction. Inf Sci, 2022, 583: 174–188
10 Li S Y, Jiang Y, Chawla N V, et al. Multi-label learning from crowds. IEEE Trans Knowl Data Eng, 2019, 31: 1369–1382
11 Dizaji K G, Gao H C, Yang Y H, et al. Robust cumulative crowdsourcing framework using new incentive payment function and joint aggregation model. IEEE Trans Neural Netw Learn Syst, 2020, 31: 4610–4621
12 Li C Q, Sheng V S, Jiang L X, et al. Noise filtering to improve data and model quality for crowdsourcing. Knowl-Based Syst, 2016, 107: 96–103
13 Sheng V S, Zhang J. Machine learning with crowdsourcing: a brief summary of the past research and future directions. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, 2019. 9837–9843
14 Wang W, Zhou Z H. Crowdsourcing label quality: a theoretical analysis. Sci China Inf Sci, 2015, 58: 112103
15 Zhang J, Sheng V S, Wu J, et al. Multi-class ground truth inference in crowdsourcing with clustering. IEEE Trans Knowl Data Eng, 2016, 28: 1080–1085

16 Sheng V S, Provost F J, Ipeirotis P G. Get another label? Improving data quality and data mining using multiple, noisy labelers. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, 2008. 614–622
17 Ma F L, Li Y L, Li Q, et al. Faitcrowd: fine grained truth discovery for crowdsourced data aggregation. In: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, 2015. 745–754
18 Zhang J, Sheng V S, Li T, et al. Improving crowdsourced label quality using noise correction. IEEE Trans Neural Netw Learn Syst, 2018, 29: 1675–1688
19 Zhang Y, Jiang L X, Li C Q. Attribute augmentation-based label integration for crowdsourcing. Front Comput Sci, 2023, 17: 175331
20 Li H W, Yu B. Error rate bounds and iterative weighted majority voting for crowdsourcing. 2014. ArXiv:1411.4086
21 Zhang J, Wu X D, Sheng V S. Imbalanced multiple noisy labeling. IEEE Trans Knowl Data Eng, 2015, 27: 489–503
22 Chen Z Q, Jiang L X, Li C Q. Label augmented and weighted majority voting for crowdsourcing. Inf Sci, 2022, 606: 397–409
23 Jiang L X, Zhang H, Tao F N, et al. Learning from crowds with multiple noisy label distribution propagation. IEEE Trans Neural Netw Learn Syst, 2022, 33: 6558–6568
24 Karger D R, Oh S, Shah D. Budget-optimal task allocation for reliable crowdsourcing systems. Opera Res, 2014, 62: 1–24
25 Yin L A, Han J H, Zhang W N, et al. Aggregating crowd wisdoms with label-aware autoencoders. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, 2017. 1325–1331
26 Tian T, Zhu J, You Q B. Max-margin majority voting for learning from crowds. IEEE Trans Pattern Anal Mach Intell, 2019, 41: 2480–2494
27 Li Y, Rubinstein B I P, Cohn T. Exploiting worker correlation for label aggregation in crowdsourcing. In: Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, 2019. 3886–3895
28 Tao D P, Cheng J, Yu Z T, et al. Domain-weighted majority voting for crowdsourcing. IEEE Trans Neural Netw Learn Syst, 2019, 30: 163–174
29 Sheng V S, Zhang J, Gu B, et al. Majority voting and pairing with multiple noisy labeling. IEEE Trans Knowl Data Eng, 2019, 31: 1355–1368
30 Yang W J, Li C Q, Jiang L X. Learning from crowds with decision trees. Knowl Inf Syst, 2022, 64: 2123–2140
31 Zhou Z H. Machine Learning. Beijing: Tsinghua University Press, 2016
32 Wilson D R, Martinez T R. Improved heterogeneous distance functions. J Artif Intell Res, 1997, 6: 1–34
33 Zhang J, Sheng V S, Nicholson B, et al. CEKA: a tool for mining the wisdom of crowds. J Mach Learn Res, 2015, 16: 2853–2858
34 Huang H, Xu H H, Wang X H, et al. Maximum F1-score discriminative training criterion for automatic mispronunciation detection. IEEE ACM Trans Audio Speech Lang Process, 2015, 23: 787–797
35 Demsar J. Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res, 2006, 7: 1–30
36 Rodrigues F, Pereira F, Ribeiro B. Sequence labeling with multiple annotators. Mach Learn, 2014, 95: 165–181