# DATA AND DOMAIN UNCERTAINTY IN MACHINE LEARNING

by

Artin Majdi

---

A Dissertation Submitted to the Faculty of the

Graduate Interdisciplinary Program
in Applied Mathematics

In Partial Fulfillment of the Requirements
For the Degree of

Doctor of Philosophy

In the Graduate College

The University of Arizona

May 24, 2023

## Statement by Author

# Dedication

For God.

# Acknowledgments

First and foremost, thanks go to co-advisors Anna Mehmbuhr and Mbaqanga Uffda. Much of what I have learned in this project has been learned through them; their patience has been endless and their knowledge has been irreplaceable. Susan Kho and Vincenzo Mitti, as well, have given me valuable feedback as I have revised my dissertation this spring.

Summer 2009, fall 2009, and spring 2010 research assistantships were funded by the University of Arizona Department of Mathematics NSF VIGRE (Vertical InteGration in Research and Education) grant. The remaining semesters of my PhD years were funded by teaching assistantships. I thank my college-algebra, trigonometry, and calculus students for helping me to see mathematics from both sides of the classroom.

# Table of Contents

# List of Figures

# List of Tables

[1]Dept of Electrical & Computer Engineering, The University of Arizona, Tucson, 86719, AZ, USA

# Abstract

Crowdsourcing systems have been used to accumulate massive amounts of labeled data for applications such as computer vision and natural language processing. However, because crowdsourced labeling is inherently dynamic and uncertain, developing a technique that can work in most situations is extremely challenging. In this paper, we propose a novel method called "crowd-certain", which provides a more accurate and reliable aggregation of labels, ultimately leading to improved overall performance in both crowdsourcing and ensemble learning scenarios. The proposed method uses the consistency and accuracy of the annotators as a measure of their reliability relative to other annotators. The experimental results show that the proposed technique generates a weight that closely follows the annotator's degree of reliability. Moreover, the proposed method uses the consistency and accuracy of annotators as a measure of their reliability relative to other annotators. Experiments performed on a variety of crowdsourcing datasets indicate that the proposed method outperforms prior methods in terms of accuracy, with significant improvement over all investigated benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene), particularly when few annotators are available.

Supervised learning, crowdsourcing, confidence score, soft weighted majority voting, label aggregation, annotator quality, error rate estimation, multi-class classification, ensemble learning, uncertainty measurement

## 0.1 Introduction

Supervised learning techniques require a large amount of labeled data to train models to classify new data [?, ?]. Traditionally, data labeling has been assigned to experts in the domain or well-trained annotators [?]. Although this method produces high-quality labels, it is inefficient and costly [?, ?]. Social networking provides an innovative solution to the labeling problem by allowing data to be labeled by online crowd annotators. This has become feasible, as crowdsourcing services such as Amazon Mechanical Turk (formerly CrowdFlower) have grown in popularity. Crowdsourcing systems have been used to accumulate large amounts of labeled data for applications such as computer vision [?, ?] and natural language processing [?]. However, because of individual differences in preferences and cognitive abilities, the quality of labels acquired by a single crowd annotator is typically low, thus jeopardizing applications that rely on these data. This is because crowd annotators are not necessarily domain experts and may lack the necessary training or expertise to produce high-quality labels. Aggregation after repeated labeling is one method for handling annotators with various abilities. Label aggregation is a process used to infer an aggregated label for a data instance from a multi-label set [?]. Several studies have demonstrated the efficacy of repeated labeling [?, ?]. Repeat labeling is a technique in which the same data are labeled by multiple annotators, and the results are combined to estimate an aggregated label using majority voting (MV) or other techniques. In the case of MV, an aggregated label is the label that receives the most votes from the annotators for a given data instance. This can help reduce the impact of biases or inconsistencies made by annotators. Several factors, such as problem-specific characteristics, the quality of the labels created by the annotators, and the amount of data available, can influence the effectiveness of the aggregation methodologies. Consequently, it is difficult to identify a clear winner among the different techniques. For example, in binary labeling, one study [?] discovered that Raykar's [?] technique outperformed other aggregation techniques. However, according to another study [?], the traditional Dawid-Skene (DS) model [?] was more reliable in multi-class settings (where data instances can be labeled as belonging to multiple classes). Furthermore, regardless of the aggregation technique used, the performance of many aggregation techniques in real-world datasets remains unsatisfactory [?]. This can be attributed to the complexity of these datasets, which often do not align with the assumptions and limitations of different methods. For example, real-world datasets may present issues such as labeling inaccuracies, class imbalances, or overwhelming sizes that challenge efficient processing with available resources. These factors can adversely affect the effectiveness of label aggregation techniques, potentially yielding less than optimal results for real-world datasets. Prior information may be used to enhance the label aggregation procedure. This can include domain knowledge, the use of quality control measures and techniques that account for the unique characteristics of annotators and data. Knowing the reliability of certain annotators, it is possible to draw more accurate conclusions about labels [?]. For instance, in the label aggregation process, labels produced by more reliable annotators (such as domain experts) may be given greater weight. The results of the label aggregation process can also be validated using expert input [?]. During the labeling process, domain experts can provide valuable guidance and oversight to ensure that the labels produced are accurate and consistent. The agnostic requirement for general-purpose label aggregation is that label aggregation cannot use information outside the labels themselves. This requirement is not satisfied in most label aggregation techniques [?]. The agnostic requirement ensures that the label aggregation technique is as general as possible and applicable to a wide range of domains with minimal or no additional context. The uncertainty of annotators during labeling can provide valuable prior knowledge to determine the appropriate amount of confidence to grant each annotator while still adhering to the requirement of a general-purpose label aggregation technique. We developed a method for estimating the reliability of different annotators based on the annotator's own consistency during labeling and their accuracy with respect to other annotators. We propose a novel method called "crowd-certain", which provides a more accurate aggregation of labels, ultimately leading to improved overall performance in both crowdsourcing and ensemble learning scenarios. The experimental results show that the proposed technique generates a weight that closely follows the annotator's degree of reliability. Furthermore, the proposed method uses the consistency and accuracy of annotators as a measure of their reliability relative to other annotators. Experiments performed on a variety of crowdsourcing datasets indicate that the proposed method outperforms prior methods in terms of accuracy with a significant improvement over all investigated benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene), particularly when few annotators are available.

The remainder of this paper is organized as follows. Section 2 examines related work involving label aggregation algorithms. In Section 3, we provide a detailed explanation of our proposed technique. Section 4 presents the experiments and findings. Finally, Section 5 summarizes the findings and identifies the main directions for future research.

## 0.2 Related Work

Numerous label aggregation algorithms have been developed to capture the complexity of crowdsourced labeling systems, including techniques based on annotator reliability [?, ?], confusion matrices [?, ?], intentions [?, ?], biases [?, ?, ?], and correlations [?]. However, because crowdsourced labeling is inherently dynamic and uncertain, developing a technique that can work in most situations is extremely challenging. Many techniques [?, ?, ?, ?, ?] utilize the Dawid and Skene (DS) generative model [?]. Ghosh [?] extended the DS model by using singular value decomposition (SVD) to calculate the reliability of the annotator. Similarly to Ghosh [?], Dalvi [?] used SVD to estimate true labels with a focus on the sparsity of the labeling matrix. In crowdsourcing, it is common for the labeling matrix to be sparse, meaning that not all annotators have labeled all the data. This may be due to several factors, such as the cost of labeling all data instances or the annotators' time constraints. Karger [?] described an iterative strategy for binary labeling based on a one-coin model [?]. Karger [?] extends the one-coin model to multi-class labeling by converting the problem into $k-1$ binary problems (solved iteratively), where $k$ is the number of classes. The MV technique assumes that all annotators are equally reliable. For segmentation, Warfield [?] proposed simultaneous truth and performance level estimation (STAPLE), a label fusion method based on expectation maximization. STAPLE "weighs" expert opinions during label aggregation by modeling their reliability. Since then, many variants of this technique have been proposed [?, ?, ?, ?, ?, ?, ?, ?, ?]. The problem with these label aggregation approaches is that they require the computation of a unique set of weights for each sample, necessitating the re-evaluation of the annotators' weights when a new instance is added. Among the numerous existing label aggregation strategies, MV remains the most efficient and widely used approach [?]. If we assume that all annotators are equally reliable and that their errors are independent of one another, then, according to the theory of large numbers, the likelihood that the MV is accurate increases as the number of annotators increases. However, the assumption that all annotators are equally competent and independent may not always hold. Furthermore, MV does not provide any additional information on the degree of disagreement among the annotators (As an example, consider the scenario where four of seven doctors think patient A needs immediate surgery, while all seven think patient B needs immediate surgery; MV will simply label "yes" in both cases). To address this problem, additional measures such as inter-annotator agreement (IAA) have been used [?]. IAA is a measurement of the agreement among multiple annotators who label the same data instance. Typically, IAA is calculated using statistical measures, such as Cohen's kappa, Fleiss's kappa, or Krippendorff's alpha [?]. These measures consider both the observed agreement between the annotators and the expected agreement owing to random chance. IAA can also be visualized using a confusion matrix or annotation heatmap, which illustrates the distribution of labels assigned by the annotators. This can help identify instances where the annotators disagree or are uncertain and can guide further analysis to improve the annotation [?]. Recently, Sheng [?] proposed a technique that provided a confidence score along with an aggregated label. The main problem with this approach is that it assumes that all annotators are equally capable when calculating the confidence score. Tao [?] improved Sheng's approach by assigning different weights to annotators for each instance. This weighting method combines the specific quality $s_{\alpha}^{(i)}$ for the annotator $\alpha$ and instance $i$ and the overall quality $\tau_{\alpha}$ across all instances. Inspired by Li's technique [?], Tao evaluates the similarity between the annotator labels for each instance. To derive the specific quality $s_{\alpha}^{(i)}$, Tao counts the number of annotators who assigned the same label as the annotator $\alpha$ for that instance. To calculate the overall quality $\tau_{\alpha}$, Tao performs a 10-fold cross-validation to train each of the 10 classifiers on a different subset of data using the labels provided by the annotator $\alpha$ as true labels and then assigns the average accuracy across all remaining instances as $\tau_{\alpha}$. The final weight for annotator $\alpha$ and instance $i$ is then calculated using the sigmoid function $\gamma_{i,\alpha} = \tau_{\alpha}\left(1 + \left(s_{\alpha}^{(i)}\right)^2\right)$. However, Tao's technique [?] has some drawbacks. It relies on the labels of other annotators to estimate $s_{\alpha}^{(i)}$. However, different annotators have varying levels of competence (reliability) when labeling the data, and therefore, relying on their labels to measure $s_{\alpha}^{(i)}$ will result in propagating the errors and biases of their labels during weight estimation. Furthermore, Tao's technique [?] relies on the labels provided by each annotator $\alpha$ to estimate their respective $\tau_{\alpha}$ by assuming that the trained classifiers can learn the inherent characteristics of the datasets even in the absence of ground truth labels. While that may be true in some cases, it typically leads to suboptimal measurement and the propagation of biases and errors, from both the annotator's labels and the classifier, into weight estimation.

## 0.3 Methods

We propose a novel method called "crowd-certain" which focuses on leveraging uncertainty measurements to improve decision-making in crowdsourcing and ensemble learning scenarios. Crowd-Certain employs a weighted soft majority voting approach, where the weights are determined based on the uncertainty associated with each annotator's labels. Initially, we use uncertainty measurement techniques to calculate the degree of consistency of each annotator during labeling. Furthermore, to ensure that the proposed technique does not calculate a high weight for annotators who are consistently wrong (for example, when a specific annotator always mislabels a specific class, and hence demonstrates a high consistency while having low accuracy), we extend the proposed technique by penalizing the annotators for instances in which they disagree with the aggregated label obtained using the MV. To mitigate the reliance on training a classifier on an annotator's labels, which may be inaccurate, we train an ensemble of classifiers for each annotator. In addition, we report two confidence scores along with the aggregated label to provide additional context for each calculated aggregate label. We report a single weight for all instances in the dataset. As demonstrated in the experimental sections, the proposed crowd-certain method is not only comparable to other techniques in terms of accuracy for scenarios with a large number of annotators, but also provides a significant improvement in accuracy for scenarios where the number of annotators may be limited. Furthermore, by assigning a single weight to each annotator for all instances in the dataset, the model can assign labels to new test instances without recalculating the annotator weights. This is especially advantageous in situations where annotators are scarce as it enables the model to make accurate predictions with minimal dependence on the annotator input. This characteristic of the crowd-certain method can significantly reduce the time and resources required for labeling in practical applications. When deploying the model in real-world scenarios such as medical diagnosis, fraud detection, or sentiment analysis, it could be advantageous to be able to assign labels to new instances without constantly recalculating annotator weights.

### 0.3.1 Glossary of Symbols

Let us denote the following parameters:

$N$: Number of instances.

$M$: Number of annotators.

$y_k^{(i)} \in \{0, 1\}$: True label for the $k$-th class for instance $i$.

$z_{\alpha,k}^{(i)} \in \{0, 1\}$: Label given by annotator $\alpha$ for $k$-th class for instance $i$.

$\underset{\alpha}{\mathrm{MV}}\left(z_{\alpha,k}^{(i)}\right)$: Majority voting technique (the label that receives the most votes) applied to annotator labels for class $k$ and instance $i$.

$\pi_{\alpha,k}$: Reliability score to generate sample labels for annotator $\alpha$ for class $k$. For example, it may be obtained from a uniform distribution in the interval 0.4 to 1, i.e., $\pi_{\alpha,k} \sim U(0.4, 1)$.

$X^{(i)}$: Data for instance $i$.

$Y^{(i)} = \left\{y_1^{(i)}, y_2^{(i)}, \ldots, y_K^{(i)}\right\}$: True label set, for instance $i$. For example, consider a dataset that is labeled for the presence of cats, dogs, and rabbits in any given instance. If a given instance $X^{(i)}$ has cats and dogs but not rabbits, then $Y^{(i)} = \{1, 1, 0\}$.

$Z_\alpha^{(i)} = \left\{z_{a,1}^{(i)},\ z_{a,2}^{(i)},\ \ldots,\ z_{a,K}^{(i)}\right\}$: Label set given by the annotator $\alpha$ for instance $i$.

$K$: number of categories (aka classes) in a multi-class multi-label problem. For example, if we have a dataset labeled for the presence of cats, dogs, and rabbits in any given instance, then $K = 3$.

$\rho^{(i)}$: Randomly generated number between 0 and 1 for instance $i$. It is obtained from a uniform distribution, i.e., $\rho^{(i)} \sim U(0, 1)$ This number will be used to determine, for each instance $i$, whether the true label should be assigned to each fictitious annotator's label. For each class $k$ if the annotator's reliability score for that class $\Pi_{\alpha,k}$ is greater than $\rho^{(i)}$, the true label $y_k^{(i)}$ will be assigned; otherwise, an incorrect label $1 - y_k^{(i)}$ will be assigned.

$\Pi_\alpha = \left\{\pi_{\alpha,1},\ \pi_{\alpha,2},\ \ldots,\ \pi_{\alpha,K}\right\}$: set of $K$ reliability scores for annotator $\alpha$.

$\mathbb{X} = \left\{X^{(i)}\right\}_{i=1}^{N}$: Set of all instances.

$\mathbb{Y} = \left\{Y^{(i)}\right\}_{i=1}^{N}$: Set of all true labels.

$\mathbb{Z}_\alpha = \left\{Z_\alpha^{(i)}\right\}_{i=1}^{N}$: Set of all labels for the annotator $\alpha$.

$\widehat{\mathbb{Y}} = \left\{\widehat{Y}^{(i)}\right\}_{i=1}^{N}$: Set of all aggregated labels.

$\mathbb{P} = \left\{\rho^{(i)}\right\}_{i=1}^{N}$: Set of $N$ randomly generated numbers.

$\mathbb{D} = \{\mathbb{X}, \mathbb{Y}\}$: Dataset containing all instances and all true labels.

$\mathbb{D}_\alpha = \{\mathbb{X}, \mathbb{Z}_\alpha\}$: Dataset containing the labels given by the annotator $\alpha$.

$\mathbb{D}_\alpha^{\text{train}}$, $\mathbb{D}_\alpha^{\text{test}}$: Train and test crowd datasets randomly selected from $\mathbb{D}_\alpha$ where $\mathbb{D}_\alpha^{\text{train}} U \mathbb{D}_\alpha^{\text{test}} = \mathbb{D}_\alpha$ and $\mathbb{D}_\alpha^{\text{train}} \bigcap \mathbb{D}_\alpha^{\text{test}} = \varnothing$

$F_\alpha^{(g)}(\cdot)$: Classifier $g$ trained on dataset $\mathbb{D}_\alpha^{\text{train}}$ with random seed number $g$ (which is also the classifier index)

$P_\alpha^{(i,g)} = \left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}$: Predicted probability set obtained in the output of the classifier $F_\alpha^{(g)}(\cdot)$ representing the probability that each class $k$ is present in the sample.

$\theta_{\alpha,k}^{(g)}$: Binarization threshold. To obtain this, we can utilize any existing thresholding technique (for example, in one technique, we analyze the ROC curve and find the corresponding threshold where the difference between the true positive rate (sensitivity) and false positive rate (1-specificity) is maximum; Alternatively, we could simply use 0.5).

$t_{\alpha,k}^{(i,g)} = \begin{cases} 1 & \text{if } p_{\alpha,k}^{(i,g)} \geq \theta_{\alpha,k}^{(g)}, \\ 0 & \text{otherwise.} \end{cases}$: Predicted label obtained by binarizing $p_{\alpha,k}^{(i,g)}$.

$\eta_{\alpha,k}^{(i)} = \underset{g}{\mathrm{MV}}\left(t_{\alpha,k}^{(i,g)}\right)$: The output of the majority vote applied to the predicted labels obtained by the $G$ classifiers.

$u_{\alpha,k}^{(i)}$: Uncertainty score.

$c_{\alpha,k}^{(i)}$: Consistency score.

$\omega_{\alpha,k}$: Estimated weight for annotator $\alpha$ and class $k$.

$\nu_k^{(i)} = \frac{1}{M}\sum_\alpha \omega_{\alpha,k}\, \eta_{\alpha,k}^{(i)}$: Final aggregated label for class $k$ and instance $i$.

### 0.3.2 Risk Calculation

Label aggregation is frequently used in various machine learning tasks, such as classification and regression, when multiple annotators assign labels to the same data points. The aggregation model refers to the underlying function that maps a set of multiple labels obtained by different annotators, into one aggregated label. In the context of label aggregation, this model can be a neural network, a decision tree, or any other machine learning algorithm capable of learning to aggregate labels provided by multiple annotators. The objective of this study is to develop an aggregation model capable of accurately determining true labels despite potential disagreements among annotators. One common method to achieve this involves minimizing the total error (or disagreement) between the annotators' assigned labels and the true labels, as follows:

$$E = \sum_{i=1}^{N} \sum_{a=1}^{M} \left( \sum_{k=1}^{K} \delta\left(y_k^{(i)}, z_{\alpha,k}^{(i)}\right) \right) \tag{0.3.1}$$

where $\delta$ is the Kronecker delta function.

Although error is a crucial aspect in determining the aggregation model's performance, it treats false positives and false negatives with equal weight. However, in many practical scenarios, it is essential to weigh false positives and false negatives differently depending on the specific context and potential consequences of each type of misclassification. The concept of risk allows us to achieve this by incorporating a loss function, which assigns different weights to different types of errors. In this way, risk serves as a weighted calculation of error, enabling us to better evaluate the performance of an aggregation model and its generalization capability.

Let us denote loss function, $\mathcal{L}(\cdot)$, as a function that quantifies the discrepancy between the predicted labels and the true labels, accounting for the varying importance of different types of errors. Risk, denoted as $R(h)$, represents the expected value of a loss function over the entire dataset, capturing the performance of the aggregation model on all possible data instances. In practice, our goal is to minimize the risk to achieve optimal performance on unseen data. However, since we only have access to a limited dataset (empirical distribution), we instead work with the empirical risk. This limitation may arise because of the need to reserve a portion of our data for testing and validation or because no dataset can fully capture all possible data instances in the real world. However, minimizing risk alone could result in overfitting, in which the aggregation model learns the noise in the training data rather than the underlying patterns, resulting in poor generalization to unseen data. To improve generalizability, it is necessary to employ regularization techniques to strike a balance between the complexity of the aggregation model and its ability to fit the training data.

Risk measurement enables us to assess the aggregation model's performance in terms of accuracy, overfitting (when risk is minimized but the model performs poorly on unseen data), and model complexity.

Assume that the aggregation model $h(\cdot)$ is a function that takes a set of $M$ label sets $Z^{(i)}$ for each instance $i$ in the training data and calculates an aggregated label set $\widehat{Y}^{(i)}$ as an estimate of the true label set $Y^{(i)}$. Our goal is to find an aggregation model $h(\cdot)$ that minimizes risk as follows:

$$R(h) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(Y^{(i)}, h\left(\left\{Z_{\alpha}^{(i)}\right\}_{\alpha=1}^{M}\right)\right) \tag{0.3.2}$$

In this context, $\mathcal{L}(\cdot)$ represents an arbitrary loss function, which quantifies the discrepancy between predicted labels and true labels while accounting for the varying importance of different types of errors.

Our goal is to choose an aggregation model $\widehat{h}$ that minimizes the risk, following the principle of risk minimization [?], as shown below:

$$\widehat{h} = \underset{h}{\operatorname{argmin}}, R(h) \tag{0.3.3}$$

### 0.3.3 Generating Annotators' Label Sets from Ground Truth

In order to evaluate the proposed crowd-certain technique (with and without penalization) as well as other aggregation techniques, we create $M$ fictitious annotators. To synthesize a multi-annotator dataset from a dataset with existing ground truth, we use a uniform distribution in the interval from 0.4 to 1, i.e., $\pi_{\alpha,k} \sim U\,(0.4, 1)$ (however other ranges can also be used) to obtain $M \times K$ reliability values $\Pi$, where $K$ is the number of classes. (Note that an annotator may be skilled at labeling dogs, but not rabbits.) Then we use these reliability values to generate the crowd label set $Z_{\alpha}^{(i)}$ from the ground truth labels for each instance $i$.

For each annotator $\alpha$, each instance $i$ and class $k$ in the dataset is assigned its true label with probability $\pi_{\alpha,k}$ and the opposite label with probability $(1-\pi_{\alpha,k})$. To generate the labels for each annotator $\alpha$, a random number $0 < \rho^{(i)} < 1$ is generated for each instance $i$ in the dataset. Then $\forall \alpha, k$ if $\rho^{(i)} \le \pi_{\alpha,k}$ then the true label is used for that instance and class for the annotator $\alpha$; otherwise, the incorrect label is used.

The calculated annotator labels $z_{\alpha,k}^{(i)}$ for each annotator $\alpha$, instance $i$ and class $k$ are as follows:

$$z_{\alpha,k}^{(i)} = \begin{cases} y_k^{(i)} & \text{if } \rho^{(i)} \leq \pi_{\alpha,k}, \\ 1 - y_k^{(i)} & \text{if } \rho^{(i)} > \pi_{\alpha,k}, \end{cases} \quad \forall i, a, k \tag{0.3.4}$$

To evaluate the proposed techniques over all data instances, a k-fold cross-validation is employed.

### 0.3.4 Uncertainty Measurement

A common approach to measure uncertainty is to increase the number of data instances $X$ in the test dataset $\mathbb{D}_\alpha^{\text{test}}$ to create multiple variations of each sample data $X^{(i)}$ [?]. In this approach, for each instance $i$, we apply randomly generated spatial transformations and additive noise to the input data $X^{(i)}$ to obtain a transformed sample and repeat this process $G$ times to obtain a set of $G$ transformed samples. However, this approach is mostly suitable for cases where the input data is images or volume slices. Since the datasets used in this study consist of feature vectors instead of images or volume slices, this approach cannot be used. To address this problem, we introduced a modified uncertainty measurement approach, in which instead of augmenting the data instances $X^{(i)}$, we feed the same sample data to different classifiers. For the choice of classifier, we can either use a probability-based classifier such as random forest and train it under $G$ different random states or train various classifiers and address the problem in a manner similar to ensemble learning (using a set of $G$ different classification techniques such as random forest, SVM, CNN, Adaboost, etc.). In either case, we obtain a set of $G$ classifiers $\left\{F_\alpha^{(g)}(\cdot)\right\}_{g=1}^{G}$ for each annotator $\alpha$. The classifier $F_\alpha^{(g)}(\cdot)$ is a pre-trained or pre-designed model that has been trained on a labeled training dataset $\mathbb{D}_\alpha^{\text{train}}$. This training process enables $F_\alpha^{(g)}(\cdot)$ to learn the underlying patterns in the data and make predictions on unseen instances. After training, we feed the test samples $X^{(i)} \in \mathbb{X}^{\text{test}}$ to the $g$-th classifier $F_\alpha^{(g)}(\cdot)$ as test cases. The classifier $F_\alpha^{(g)}(\cdot)$ then outputs a set of predicted probabilities $\left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}$

representing the probability that class $k$ is present in the sample. Consequently, we obtain a collection of $G$ predicted probability sets $\left\{\left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}\right\}_{g=1}^{G}$ for each annotator $\alpha$ and instance $i$. The set $\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}$ contains the predicted probabilities for class $k$, annotator $\alpha$, and instance $i$. Disagreements between predicted probabilities $\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}$ can be used to estimate uncertainty. The reason for using classifiers rather than using the crowdsourced labels directly is two-fold. Using a probabilistic classifier helps us calculate uncertainty based on each annotator's labeling patterns that the classifier learns. Furthermore, this approach provides us with a set of pre-trained classifiers $\left\{\left\{F_{\alpha}^{(g)}(\cdot)\right\}_{g=1}^{G}\right\}_{a=1}^{M}$ that can be readily utilized on any new data instances without the need for those samples to be labeled by the original annotators. The index value $g \in \{1, 2, \ldots, G\}$ is used as the random seed value during training of the $g$th classifier for all annotators. Define $t_{\alpha,k}^{(i,g)}$ as the predicted label obtained by binarizing the predicted probabilities $p_{\alpha,k}^{(i,g)}$ using the threshold $\theta_{\alpha,k}^{(g)}$ as shown in the Glossary of Symbols section. Uncertainty measures are used to quantify the level of uncertainty or confidence associated with the predictions of a model. In this work, we need to measure the uncertainty $u_{\alpha,k}^{(i)}$ associated with the model predictions. Some common uncertainty measurement measures are as follows.

*Entropy* Entropy is a widely used measure of uncertainty in classification problems. In an ensemble of classifiers, entropy serves as a quantitative measure of the uncertainty or disorder present in the probability distribution of the predicted class labels. A higher entropy value indicates a greater degree of uncertainty in the predictions, as the predictions of the individual classifiers in the ensemble are significantly different. In contrast, a lower entropy value denotes reduced uncertainty as the ensemble assigns very similar probabilities to a particular class, indicating strong agreement among the classifiers and increased confidence in their collective prediction. The formula for calculating entropy is as follows:

$$u_{\alpha,k}^{(i)} = H\left(\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right) = -\sum_{g} p_{\alpha,k}^{(i,g)} \log\left(p_{\alpha,k}^{(i,g)}\right) \tag{0.3.5}$$

*Standard Deviation* In regression problems, standard deviation is often used to quantify uncertainty. It measures the dispersion of predicted values around the mean. A greater standard deviation indicates greater uncertainty of the prediction. For a set of predicted values $\{t_{\alpha,k}^{(i,g)}\}_{g=1}^{G}$ with mean value $\mu$, the standard deviation is defined as

$$u_{\alpha,k}^{(i)} = \mathrm{SD}\left(\left\{t_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right) = \sqrt{\frac{1}{G-1}\sum_{g=1}^{G}\left(t_{\alpha,k}^{(i,g)} - \mu\right)^2}, \quad \mu = \frac{1}{G}\sum_{g=1}^{G} t_{\alpha,k}^{(i,g)} \tag{0.3.6}$$

**Predictive Interval:** A predictive interval provides a range within which a future observation is likely to fall with a certain level of confidence. For example, a 95% predictive interval indicates that there is a 95% likelihood that the true value falls within that range. A greater uncertainty corresponds to wider intervals. In the context of multiple classifiers, the predictive intervals can be calculated by considering the quantiles of the classifier output. For a predefined confidence level $\gamma$ (e.g., 95%), for a specific class $k$, we need to find the quantiles $Q_L^k$ and $Q_U^k$ of the probability distribution of class $k$ predicted by the $G$ classifiers. The uncertainty can be represented by the width of the predictive interval:

$$\begin{gathered} P\left(Q_L^k \leq p_{\alpha,k}^{(i,g)} \leq Q_U^k\right) = \gamma \\ u_{\alpha,k}^{(i)} = Q_L^k - Q_U^k \end{gathered} \tag{0.3.7}$$

The steps to calculate the predictive interval are as follows:

1. Collect the class $k$ probabilities predicted by all $G$ classifiers for a given instance. Then sort the values in ascending order. Let us call this set $P_{\alpha,k}^{(i)} = \text{sorted}\left(\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right), \quad \forall \alpha, k, i.$

2. Calculate the lower and upper quantile indices based on the chosen confidence level $\gamma$. The lower quantile index is $L = \text{ceil}\left(\frac{G}{2}(1-\gamma)\right)$, and the upper quantile index is $U = \text{floor}\left(\frac{G}{2}(1+\gamma)\right)$, where ceil and floor are the ceiling and floor functions, respectively.

3. Find the values corresponding to the lower and upper quantile indices in the sorted $P_{\alpha,k}^{(i)}$. These values are the lower and upper quantiles $Q_L^k$ and $Q_U^k$.

4. Now we have the predictive interval $P\left(Q_L^k <= p_{\alpha,k}^{(i,g)} <= Q_U^k\right) = \gamma$, where $Q_L^k$ and $Q_U^k$ represent the bounds of the interval containing the $\alpha$ proportion of the probability mass.

*Monte Carlo Dropout* The Monte Carlo dropout [?] can be used to estimate uncertainty in neural networks by applying the dropout at test time. Multiple forward passes with dropout generate a distribution of predictions from which uncertainty can be derived using any of the aforementioned techniques (standard deviation, entropy, etc.).

*Bayesian Approaches* Bayesian methods offer a probabilistic framework to estimate the parameters of the model and make predictions. These methods explicitly model uncertainty by considering prior beliefs about the model parameters and then updating those beliefs based on the observed data. In Bayesian modeling, the model parameters are treated as random variables and a posterior distribution is estimated using these parameters. The following are two common Bayesian approaches for measuring the uncertainty in classification problems.

- **Bayesian model averaging (BMA):** BMA accounts for model uncertainty by combining the predictions of various models using their posterior probabilities as weighting factors. Instead of selecting a single "best" model, BMA acknowledges the possibility of multiple plausible models, each with its own strengths and weaknesses [?]. The steps to implement BMA are as follows. Select a set of candidate models that represent different hypotheses regarding the data-generating process underlying the data. These models may be of various types, such as linear regression, decision trees, neural networks, or any other model suited to the specific problem at hand. Using the available data, train each candidate model. Calculate the posterior probabilities of the models. Using the posterior probabilities of each model as weights, calculate the weighted average of each model's predictions. The weighted average is the BMA prediction for the input instance and class.

- **Bayesian neural networks (BNNs):** BNNs [?] are an extension of conventional neural networks in which the weights and biases of the network are treated as random variables. The primary distinction between BNNs and conventional neural networks is that BNNs model uncertainty directly in the weights and biases. The posterior distributions of the network weights and biases (learned during training) capture the uncertainty, which can then be utilized to generate predictive distributions for each class. This enables multiple predictions to be generated by sampling these predictive distributions, which can be used to quantify the uncertainty associated with each class.

*Committee-Based Methods* Committee-based method [?] involves training multiple models (a committee) and aggregating their predictions. The disagreement between committee members' predictions can be used as a measure of uncertainty. Examples include bagging and boosting ensemble methods and models, such as random forests.

$$u_{\alpha,k}^{(i)} = \text{VarCommittee}\left(P_{\alpha,k}^{(i)}\right) = \frac{1}{G}\sum_{g=1}^{G}\left(p_{\alpha,k}^{(i,g)} - \mu\right)^2, \quad \mu = \frac{1}{G-1}\sum_{g=1}^{G} p_{\alpha,k}^{(i,g)} \tag{0.3.8}$$

*Conformal Prediction:* Conformal prediction [?] is a method of constructing prediction regions that maintain a predefined level of confidence. These regions can be used to quantify the uncertainty associated with the prediction of a model.

Steps to calculate the nonconformity score:

[1.]For each classifier $g$ and each class $k$, calculate the nonconformity score. Here, score_function measures the conformity of the prediction with the true label. In the context of this study, the true label can be replaced by $\eta_{\alpha,k}^{(i)}$. A common choice for score_function is the absolute difference between the predicted probability and the true label, but other options can be used depending on the specific problem and requirements. Define the nonconformity score as $\zeta_k^g = \text{score\_function}\left(p_{\alpha,k}^{(i,g)}, y_k^{(i)}\right)$ Calculate the p-value $\text{pv}_k$ for each class $k$ as the proportion of classifiers with nonconformity scores greater than or equal to a predefined threshold $\text{T}_k$ : $\text{p-values}(k) = \frac{|\{g: \zeta_k^g \geq T_k\}|}{G}$ The p-values calculated for each class $k$ represent the uncertainty associated with that class. A higher p-value indicates a higher level of agreement among the classifiers for a given class, whereas a lower p-value suggests greater uncertainty or disagreement.

The uncertainty measures discussed above are only some of the available options. Selecting an appropriate measure depends on factors such as the problem domain, the chosen model, and the specific requirements of a given application. For this study, we use the variance technique shown in Equation (**??**) as our uncertainty measurement due to its simplicity. However, other measures could also be employed as suitable alternatives.

### 0.3.5 Crowd-Certain: Uncertainty-Based Weighted Soft Majority Voting

*Consistency Measurement* Define $c_{\alpha,k}^{(i)}$ as the consistency score for annotator $\alpha$, class $k$ and instance $i$. We calculate this consistency score using the uncertainty score $u_{\alpha,k}^{(i)}$ explained in the previous section. We use two approaches to calculate $c_{\alpha,k}^{(i)}$ from $u_{\alpha,k}^{(i)}$.

[1.]The first approach is to simply subtract the uncertainty from 1 as follows:

$$c_{\alpha,k}^{(i)} = 1 - u_{\alpha,k}^{(i)} \quad , \ \forall i, \alpha, k \tag{0.3.9}$$

In a second approach (shown in Equation (??)), we penalize annotators for instances in which their predicted label $\eta_{\alpha,k}^{(i)}$ (explained in the Glossary of Symbols section) does not match the MV of all annotator labels $\underset{\alpha}{\text{MV}}\left(z_{\alpha,\,k}^{(i)}\right)$. As previously discussed, instead of directly working with the annotator's labels $z_{\alpha,k}^{(i)}$, we use the predicted labels obtained from the ensemble of classifiers $\eta_{\alpha,k}^{(i)}$. This methodology does not require repeating the crowd-labeling process for new data samples. In particular, we are likely not to have access to the same crowd of annotators employed in the training dataset.

$$c_{\alpha,k}^{(i)} = \begin{cases} 1 - u_{\alpha,k}^{(i)} & \text{if } \eta_{\alpha,k}^{(i)} = \text{MV}_{\alpha}(\eta_{\alpha,k}^{(i)}) \\ 0 & \text{otherwise} \end{cases} \tag{0.3.10}$$

*Weight Measurement* Furthermore, we define the annotators' weights $\omega_{\alpha,k}$ for each class k as the mean of their respective consistency scores $c_{\alpha,k}^{(i)}$ over all instances. The weights are normalized between 0 and 1 as follows:

$$\omega_{\alpha,k} = \frac{\psi_{\alpha,k}}{\sum_{\alpha=1}^{M} \psi_{\alpha,k}} \quad \text{where} \quad \psi_{\alpha,k} = \frac{1}{N} \sum_{i=1}^{N} c_{\alpha,k}^{(i)} \tag{0.3.11}$$

*Aggregated Label Calculation* Finally, the aggregated label $v_k^{(i)}$ for each instance $i$ and class $k$ is the weighted average of the predicted labels $\eta_{\alpha,k}^{(i)}$ for each annotator $\alpha$:

$$v_k^{(i)} = \begin{cases} 1 & \text{if } \left(\sum_{\alpha=1}^{M} \omega_{\alpha,k}\, \eta_{\alpha,k}^{(i)}\right) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, k \tag{0.3.12}$$

*Confidence Score Calculation* In previous section we showed how to calculate the aggregated label $v_k^{(i)}$ (shown in Equation (??)). Define $F_k^{(i)}$ as the confidence score for instance $i$ and class $k$. In this section, we calculate two confidence scores $F_k^{(i)}$, based on how many different annotators agree on the reported label $v_k^{(i)}$, which will be reported alongside the aggregated label $v_k^{(i)}$. The confidence scores show the level of confidence we should place on the aggregated labels.

To calculate this confidence score, we modify the two techniques used by Sheng [?] and Tao [?] to incorporate our calculated weight $\omega_{\alpha,k}$ shown in Equation (**??**) for each worker $\alpha$.

[1.]**uwMV-Freq:** In this approach, the confidence score $F^{(i)}$ is defined as the weighted sum of labels belonging to annotators whose label is the same as the final aggregated label. It is calculated as

$$F_k^{(i)} = \sum\nolimits_{\alpha=1}^{M} \omega_{\alpha,k}\ \delta\left(\eta_{\alpha,k}^{(i)}\ ,\ \ v_k^{(i)}\right) \tag{0.3.13}$$

where $\delta$ is the Kronecker delta function. **uwMV-Beta:** In this approach, the CDF of the beta distribution at the decision threshold of 0.5 is used to calculate a confidence score $F^{(i)}$. To calculate the two shape parameters of the beta distributions $\alpha^{(i)}$ and $\beta^{(i)}$, we use a weighted sum of all correct and incorrect aggregated labels, respectively:

$$\begin{aligned} l_k^{(i)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha,k}\ \delta\left(\eta_{\alpha,k}^{(i)}, v_k^{(i)}\right) \\ u_k^{(i)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha,k}\ \delta\left(\eta_{\alpha,k}^{(i)}, 1 - v_k^{(i)}\right) \end{aligned} \tag{0.3.14}$$

$$F_k^{(i)} = I_{0.5}\left(l_k^{(i)}, u_k^{(i)}\right) = \sum_{t=\lfloor l_k^{(i)} \rfloor}^{T-1} \frac{(T-1)!}{t!(T-1-t)!} 0.5^{T-1} \tag{0.3.15}$$

where $T = \left\lfloor l_k^{(i)} + u_k^{(i)} \right\rfloor$ and $\lfloor \cdot \rfloor$ is the floor function.

## 0.4 Results

In this section, we assess the efficacy of our proposed strategy. To evaluate our proposed technique, we conducted a series of experiments comparing the proposed technique with existing state-of-the-art techniques such as MV, Tao [?], and Sheng [?], as well as with other crowdsourcing methodologies reported in the crowd-kit package [?] including Gold Majority Voting, MMSR [?], Wawa, Zero-Based Skill, GLAD [?], and Dawid Skene [?].

### 0.4.1 Datasets:

We report the performance of our proposed techniques on various datasets. These datasets cover a wide range of domains and have varying characteristics in terms of the number of features, samples, and class distributions. Table **??** provides an overview of the datasets used. All datasets are obtained from the University of California, Irvine (UCI) repository [?].

TABLE 1. Descriptions of the datasets used.

| **Dataset** | **#Features** | **#Samples** | **#Positives** | **#Negatives** |
|---|---|---|---|---|
| kr-vs-kp | 36 | 3196 | 1669 | 1527 |
| mushroom | 22 | 8124 | 4208 | 3916 |
| iris | 4 | 100 | 50 | 50 |
| spambase | 58 | 4601 | 1813 | 2788 |
| tic-tac-toe | 10 | 958 | 332 | 626 |
| sick | 30 | 3772 | 231 | 3541 |
| waveform | 41 | 5000 | 1692 | 3308 |
| car | 6 | 1728 | 518 | 1210 |
| vote | 16 | 435 | 267 | 168 |
| ionosphere | 34 | 351 | 126 | 225 |

- The **kr-vs-kp** dataset represents the King Rook-King Pawn on a7 in chess. The positive class indicates a victory for white (1,669 instances, or 52%), while the negative class indicates a defeat for white (1,527 instances, 48%).

- The **mushroom** dataset is based on the Audubon Society Field Guide for North American Mushrooms (1981) and includes 21 attributes related to mushroom characteristics such as cap shape, surface, odor, and ring type.

- The **Iris** Plants Dataset comprises three classes, each with 50 instances, representing different iris plant species. The dataset contains four numerical attributes in centimeters: sepal length, sepal width, petal length, and petal width.
- The **Spambase** dataset consists of 57 attributes, each representing the frequency of a term appearing in an email, such as the "address".
- The **tic-tac-toe** endgame dataset encodes all possible board configurations for the game, with "x" playing first. It contains nine attributes corresponding to the tic-tac-toe squares: x, o, and b (blank).
- The **Sick** dataset includes thyroid disease records from the Garvan Institute and J. Ross Quinlan of the New South Wales Institute in Sydney, Australia. 3,772 instances with 30 attributes (seven continuous and 23 discrete) and 5.4% missing data. Age, pregnancy, TSH, T3, TT4, etc.
- The **waveform** dataset generator comprises 41 attributes and three wave types, with each class consisting of two "base" waves.
- The **Car** Evaluation Dataset rates cars on price, buying, maintenance, comfort, doors, capacity, luggage, boot size, and safety using a simple hierarchical decision model. The dataset consists of 1,728 instances categorized as unacceptable, acceptable, good, and very good.
- The 1984 US Congressional **Voting** Records Dataset shows how members voted on 16 CQA-identified critical votes. Votes are divided into nine categories, simplified to yea, nay, or unknown disposition. The dataset has two classes: Democrats (267) and Republicans (168).
- The Johns Hopkins **Ionosphere** dataset contains data collected near Goose Bay, Labrador, using a phased array of 16 high-frequency antennas. "Good"

radar returns show ionosphere structure, while “bad” returns are ionosphere-free. The dataset includes 351 instances with 34 attributes categorized as good or bad.

All datasets were transformed into a two-class binary problem for comparison with existing benchmarks. For instance, only the first and second classes were used in the “waveform” dataset, and the first two classes were utilized in the “Iris” dataset. In this study, we generated multiple fictitious label sets for each dataset to simulate the crowdsourcing concept of collecting several crowd labels for each instance. This was achieved by selecting random samples in the datasets using a uniform distribution and altering their corresponding true labels to incorrect ones, while maintaining the original distribution of the ground-truth labels. The probability of each instance containing the correct true label was determined using a uniform distribution, allowing us to create synthetic label sets for each annotator that preserved the underlying structure and difficulty of the original classification problem. By creating datasets with varying levels of accuracy, we aim to evaluate the performance of our proposed method under different conditions, such as varying annotator expertise and reliability. This process allowed us to assess the ability of our method to handle diverse real-world crowdsourcing scenarios and gain insight into its general applicability and effectiveness in improving overall classification accuracy.

### 0.4.2 Benchmarks:

Tao [?] and Sheng [?] techniques were implemented in Python to evaluate their performance. Furthermore, the crowd-kit package (A General-Purpose Crowdsourcing Computational Quality Control Toolkit for Python) [?] was used to implement the remaining benchmark techniques, including Gold Majority Voting, MMSR [?], Wawa, Zero-Based Skill, GLAD [?], and Dawid Skene [?].

- **Golden majority voting** estimates the probability that each annotator possesses the correct label and then calculates the probability of each label per occurrence based on the weight assigned to each label. Assume that 10,000 jobs are performed by 3,000 annotators as well as 100 instances of ground truth. First, the percentage of correct labels for each annotator is calculated. The remaining labels were then estimated based on the weights.

- **Wawa** (Annotator Agreement with Aggregate), also referred to as "inter-rater agreement", is a commonly used statistic for non-testing problems. This indicates the average frequency with which each annotator's response matches the aggregate response for each instance.

- **Zero-Based-Skill** employs a weighted majority vote (WMV). After processing a collection of instances, it re-evaluated the abilities of the annotators based on the accuracy of their responses. This process is repeated until the labels no longer change or the maximum number of iterations is reached.

- Descriptions of the other techniques can be found in their respective references.

### 0.4.3 Weight Measurement:

After generating the multi-label sets, we employed both the proposed and state-of-the-art approaches to obtain the aggregated labels. We experimented with two approaches for classifier selection, as explained in Section 3.4.1. We found no significant differences in the overall outcomes and thus chose the second approach, which utilized the Random Forest classification technique, to save processing time and reduce the number of required Python package dependencies. Ten Random Forests, each with four trees and a maximum depth of four, were trained in different random states for each annotator $\alpha$, as detailed in Section 3.

**Annotators' reliability vsėstimated weight** $\omega_{\alpha,k}$

Figure **??** depicts the relationship between the randomly assigned annotators' reliability value ($\pi_{\alpha,k}$) and their corresponding estimated weights, $\omega_{\alpha,k}$. In the case of Tao's method, the figure displays the average weights across all instances. As seen, when the reliability of an annotator surpasses a specific threshold, the weight measured by Tao's technique plateaus, whereas the proposed method exhibits a considerably stronger correlation. Individual data points represent the actual measured weights, and the curve represents the regression line.

### 0.4.4 Confidence-score

The results present a box plot of the average accuracy for different numbers of annotators, ranging from three to ten. Figure **??** illustrates the average accuracy of the crowd-certain technique with penalization for both the "freq" and "Beta" confidence measurement approaches. A noticeable difference in the accuracy was observed. However, statistical analysis did not reveal a significant difference between these approaches.

Figure **??** displays the average accuracy using the "freq" confidence measurement strategy for the proposed crowd-certain technique with and without penalization. The penalization method occurs by penalizing annotators for inaccurate labeling before measuring their weights, as demonstrated in Equation (**??**). The penalized version of the proposed technique shows an improvement in average accuracy and a reduction in variance.

Figure **??** shows the average accuracy distribution ("freq" strategy) of the proposed penalized versus Tao and Sheng for a different number of annotators using the kernel density estimation technique. This demonstrates that the proposed technique outperforms both Tao and Sheng on nine of the ten datasets, with higher average accuracy and less fluctuation over different annotator counts. Table **??** shows the statistical data measured between the proposed penalized technique and Tao's method. As can

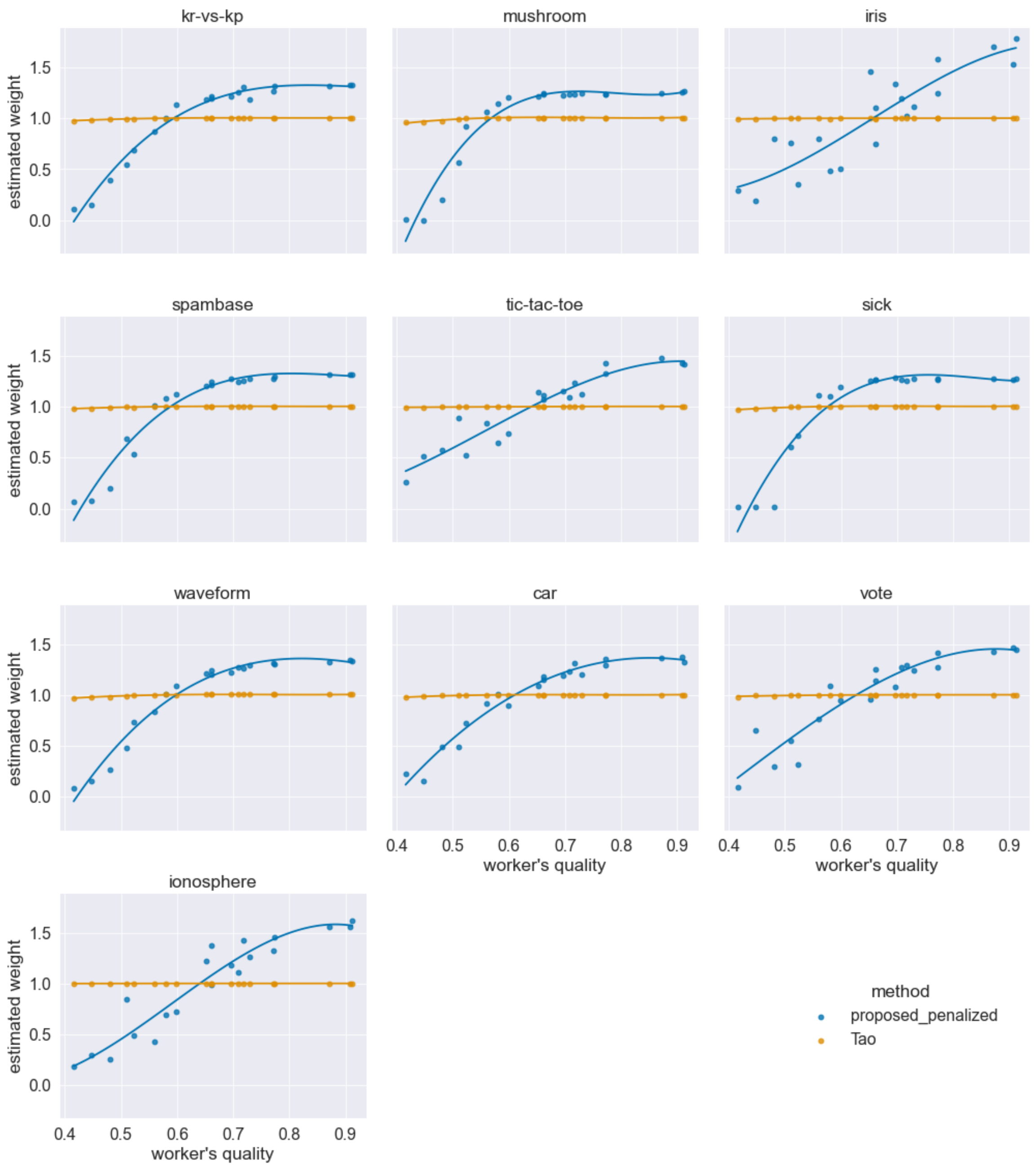

Figure 1. Comparison of estimated weight with respect to annotators' degree of reliability for the proposed aggregation technique "proposed-penalized" and Tao [?] for 10 different datasets.

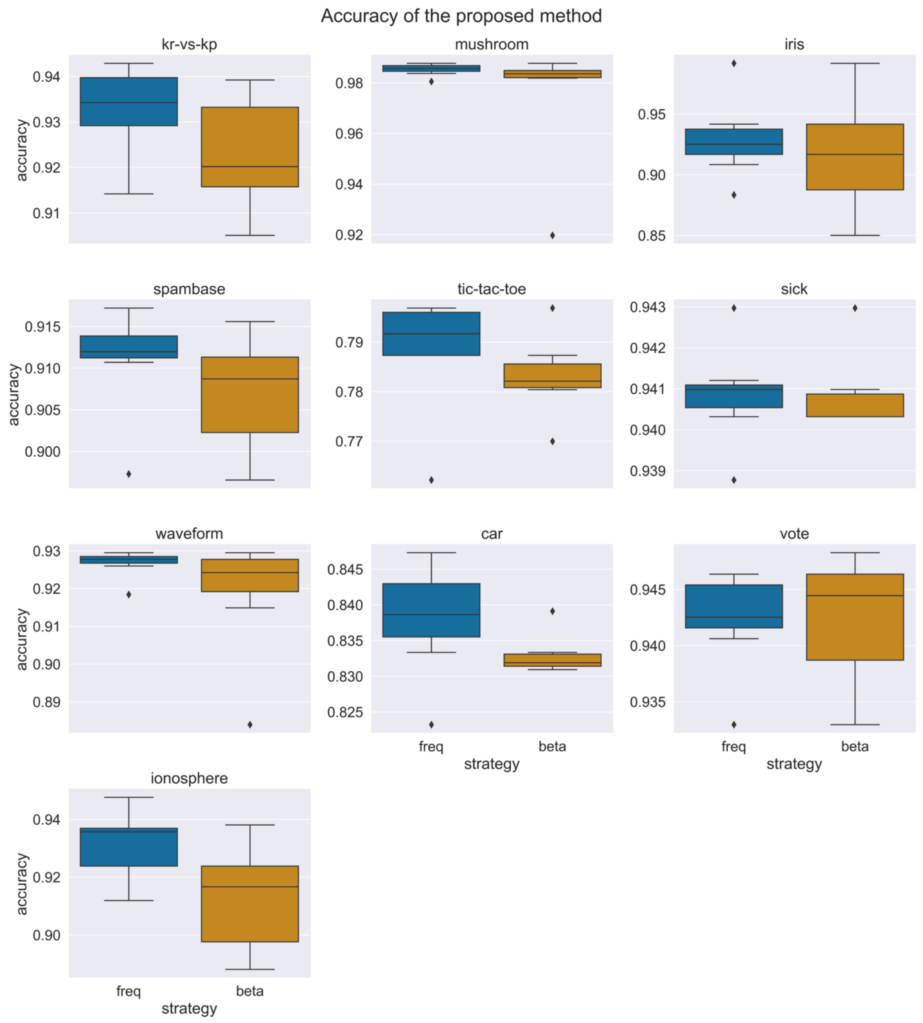

FIGURE 2. Comparison of the measured average accuracy for the two confidence-score measurement techniques in ten different datasets (using the proposed crowd-certain technique with penalization) in different numbers of annotators (from 3 up to 10).

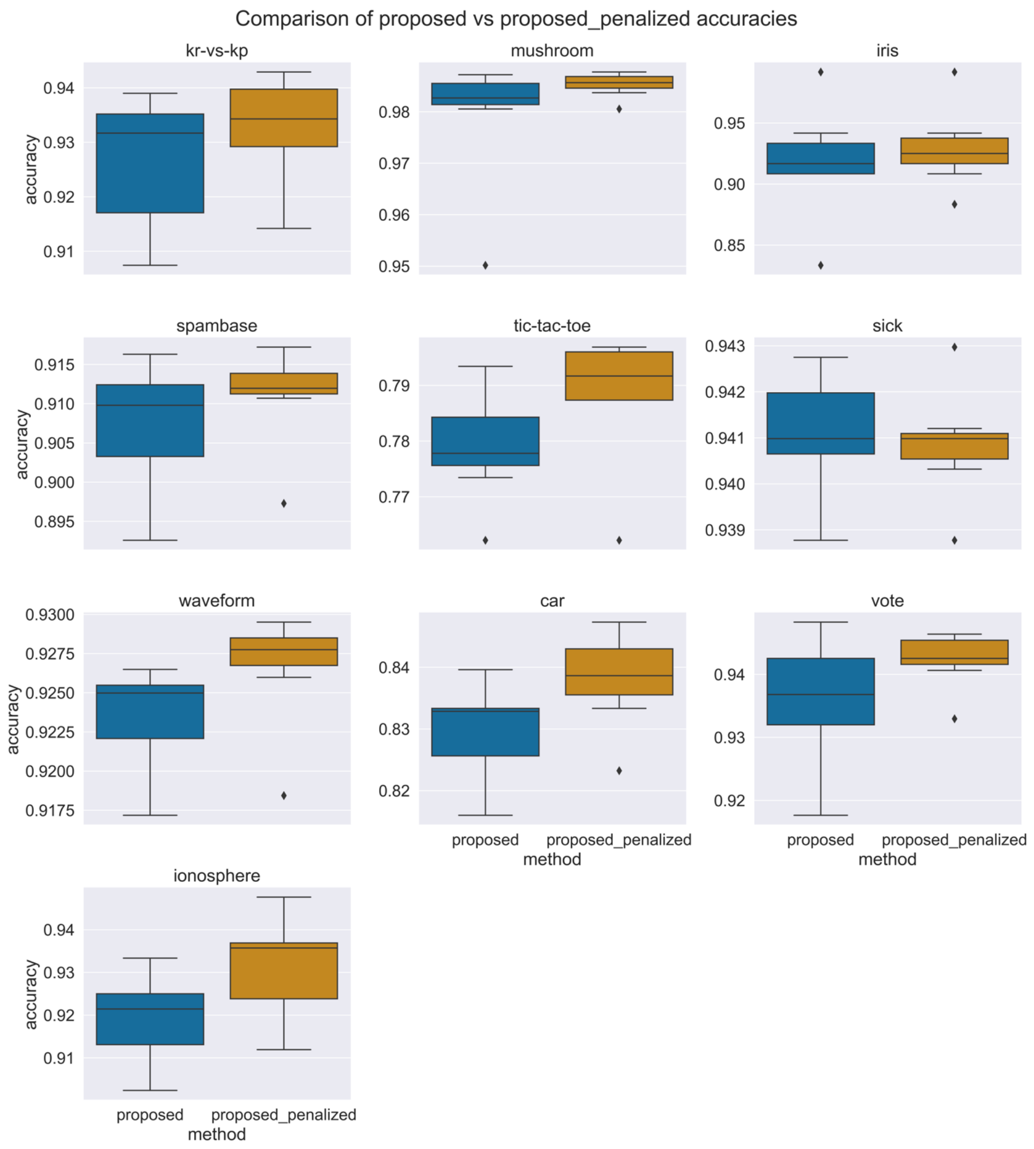

FIGURE 3. Comparison of the measured average accuracy for the proposed crowd-certain technique with and without penalization and using the ‘freq’ confidence-score strategy on ten different datasets with different numbers of annotators (starting from 3 to 10).

Table 2. Statistical tests between the proposed-penalized technique and Tao [?] for the 'freq' confidence measurement strategy.

| **Independent t-test** | **Diff** | **Degrees of freedom** | **t** | **2-sided p-value** | **Diff<0 p-value** | **Diff>0 p-value** | **Cohen d** | **Hedge's g** | **Glass's delta** | **Pearson r** |
|---|---|---|---|---|---|---|---|---|---|---|
| Kr-vs-kp | -0.044 | 12 | -6.171 | 0 | 0 | 1 | -3.299 | -3.088 | -2.743 | 0.872 |
| mushroom | -0.012 | 12 | -2.781 | 0.017 | 0.008 | 0.992 | -1.487 | -1.392 | -1.076 | 0.626 |
| iris | -0.012 | 12 | -0.506 | 0.622 | 0.311 | 0.689 | -0.271 | -0.253 | -0.226 | 0.145 |
| spambase | -0.031 | 12 | -3.691 | 0.003 | 0.002 | 0.998 | -1.973 | -1.847 | -1.458 | 0.729 |
| tic-tac-toe | -0.036 | 12 | -4.612 | 0.001 | 0 | 1 | -2.466 | -2.308 | -2.156 | 0.8 |
| sick | 0.002 | 12 | 0.294 | 0.774 | 0.613 | 0.387 | 0.157 | 0.147 | 0.112 | 0.085 |
| waveform | -0.025 | 12 | -5.118 | 0 | 0 | 1 | -2.736 | -2.561 | -2.022 | 0.828 |
| car | -0.008 | 12 | -0.779 | 0.451 | 0.226 | 0.774 | -0.416 | -0.39 | -0.309 | 0.219 |
| vote | -0.033 | 12 | -5.352 | 0 | 0 | 1 | -2.861 | -2.678 | -2.112 | 0.84 |
| ionosphere | -0.061 | 12 | -6.047 | 0 | 0 | 1 | -3.232 | -3.026 | -2.586 | 0.868 |

Table 3. Statistical tests between the proposed-penalized and Tao [?] technique for the "Beta" confidence measurement strategy.

| **Independent t-test** | **Diff** | **Degrees of freedom** | **t** | **2-sided p-value** | **Diff $<$ 0 p-value** | **Diff $>$ 0 p-value** | **Cohen d** | **Hedge's g** | **Glass's delta** | **Pearson r** |
|---|---|---|---|---|---|---|---|---|---|---|
| kr-vs-kp | -0.04 | 12 | -5.702 | 0 | 0 | 1 | -3.048 | -2.854 | -2.921 | 0.855 |
| mushroom | -0.009 | 12 | -0.793 | 0.443 | 0.222 | 0.778 | -0.424 | -0.397 | -0.477 | 0.223 |
| iris | -0.002 | 12 | -0.091 | 0.929 | 0.465 | 0.535 | -0.048 | -0.045 | -0.048 | 0.026 |
| spambase | -0.03 | 12 | -3.547 | 0.004 | 0.002 | 0.998 | -1.896 | -1.775 | -1.421 | 0.715 |
| tic-tac-toe | -0.033 | 12 | -4.326 | 0.001 | 0 | 1 | -2.312 | -2.165 | -1.781 | 0.781 |
| sick | 0.001 | 12 | 0.14 | 0.891 | 0.554 | 0.446 | 0.074 | 0.07 | 0.053 | 0.04 |
| waveform | -0.023 | 12 | -2.672 | 0.02 | 0.01 | 0.99 | -1.428 | -1.337 | -1.442 | 0.611 |
| car | -0.008 | 12 | -0.871 | 0.401 | 0.2 | 0.8 | -0.465 | -0.436 | -0.332 | 0.244 |
| vote | -0.034 | 12 | -5.198 | 0 | 0 | 1 | -2.779 | -2.601 | -2.081 | 0.832 |
| ionosphere | -0.052 | 12 | -3.875 | 0.002 | 0.001 | 0.999 | -2.071 | -1.939 | -1.742 | 0.746 |

be seen from these results, the proposed technique has a significant improvement over the seven datasets, while delivering similar results for the other three datasets (p-value $< 0.05$).

Figure **??** shows the average accuracy distribution ("Beta" strategy) of the proposed penalized versus Tao and Sheng strategies for different numbers of annotators using kernel density estimation. This demonstrates that the proposed technique outperforms both Tao and Sheng on seven out of ten datasets, with higher average accuracy and less fluctuation over different annotator counts. Furthermore, Table **??** shows the statistical data measured between the proposed penalized technique and Tao, showing a significant improvement in six datasets, while performing similarly in the remaining datasets (p-value $< 0.05$).

Figure **??** shows the average accuracy for the "ionosphere" dataset for various annotator counts (horizontal axis). As can be seen, the proposed strategies consid-

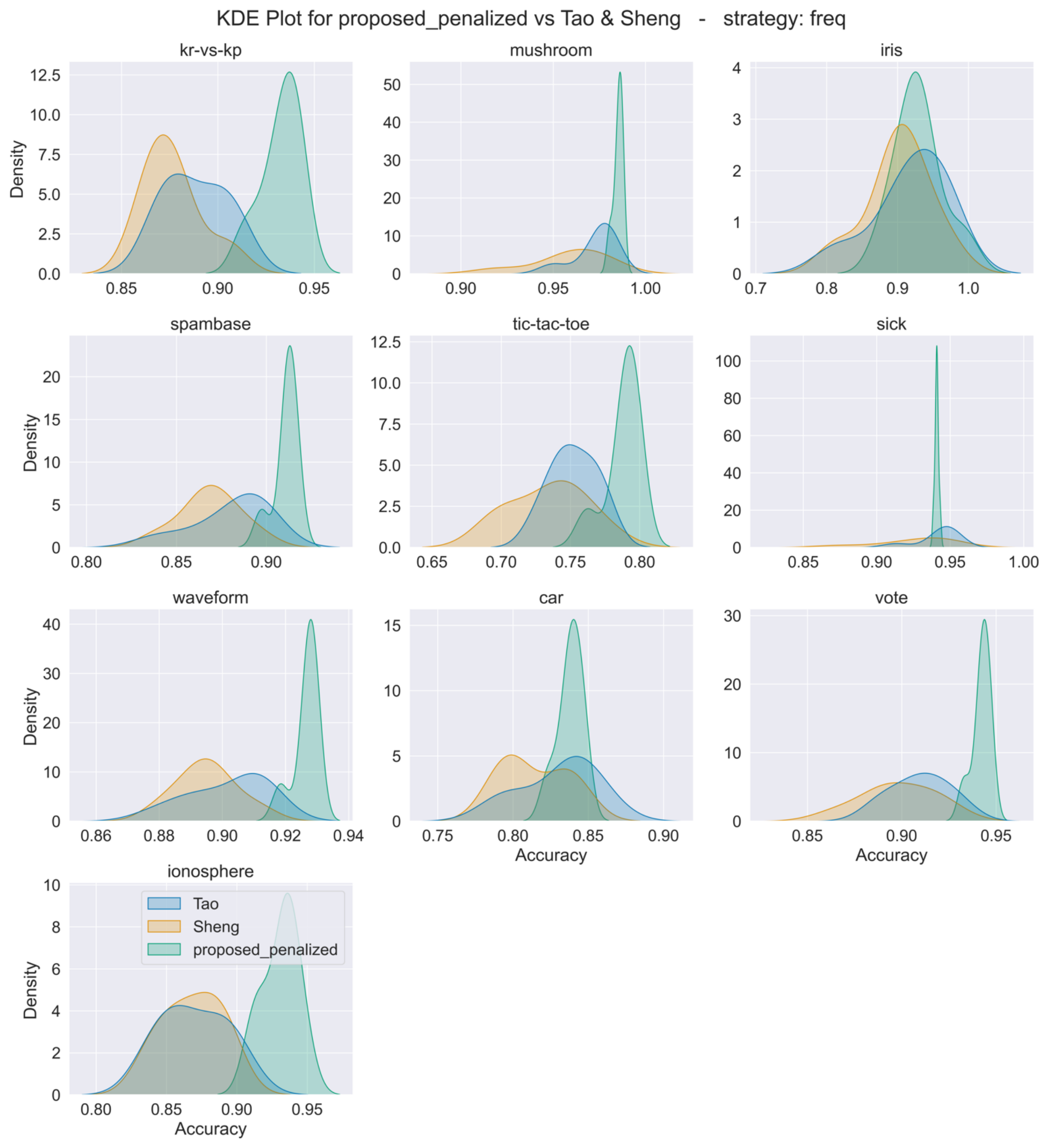

FIGURE 4. Measured accuracy distribution of the proposed-penalized aggregation technique uwMV-Freq, compared to wMV-Freq (Tao [?]), and MV-Freq (Sheng [?]) for different numbers of annotators, using the kernel density estimation technique

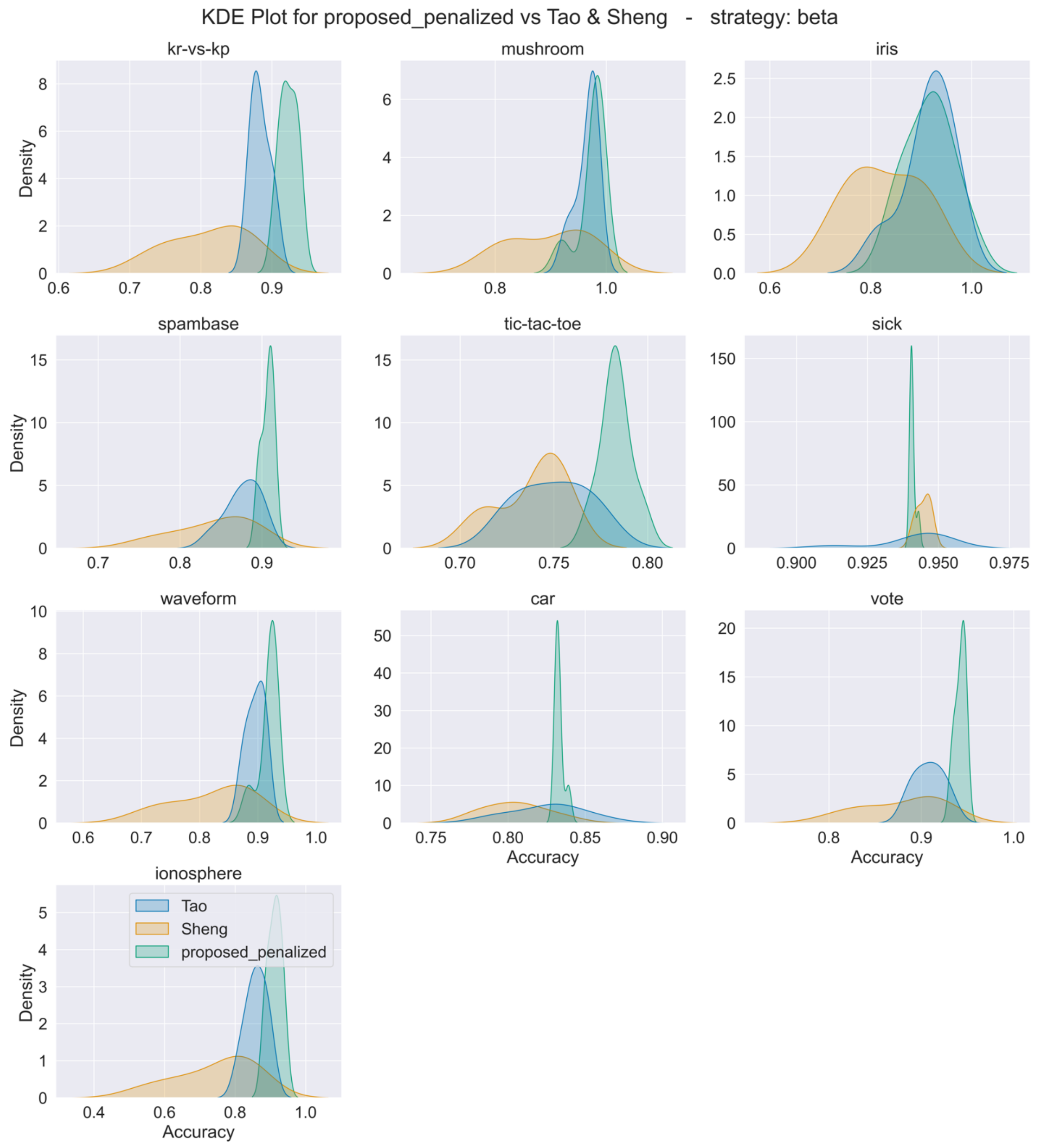

FIGURE 5. Measured accuracy distribution of the proposed-penalized aggregation technique uwMV-Beta, compared to wMV-Beta (Tao [?]), and MV- Beta (Sheng [?]) for different numbers of annotators, using the kernel density estimation technique.

erably improve accuracy while utilizing a small number of annotators. Furthermore, Figure **??** shows the average accuracy of the three annotators (the smallest number of annotators that could perform consensus voting) on all ten datasets. Similarly, for the ionosphere dataset, we observed a similar trend in achieving higher accuracy on nine of the ten datasets compared to all benchmarks. It is important to note that the "freq" confidence measurement strategy is used to report the proposed techniques, as well as the Tao and Sheng results. Furthermore, the measured p-value calculated for the measured average accuracy (using three annotators) over different datasets showed a significant improvement for the proposed technique with and without penalization over all remaining benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, Dawid Skene).

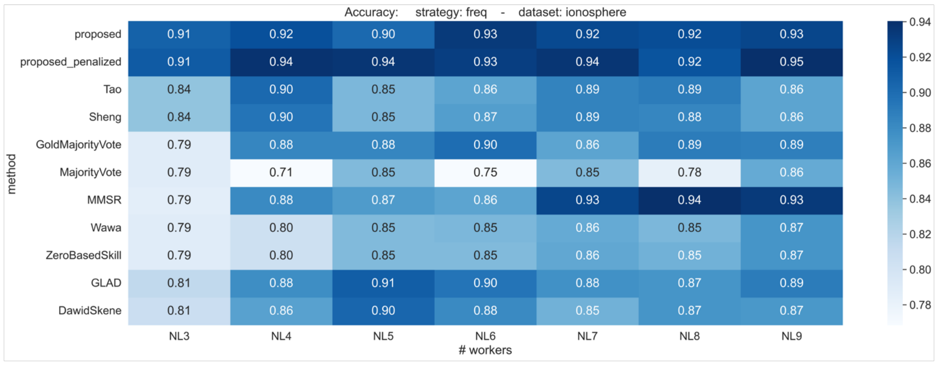

FIGURE 6. Average accuracy for the proposed aggregation techniques compared to the benchmarks for different numbers of annotators (horizontal axis) in the ionosphere dataset.

## 0.5 Discussion

Label aggregation is a critical component of crowdsourcing and ensemble learning strategies. Many generic label aggregation algorithms fall short because they do not account for the varying reliability of the annotators. In response to this, we have de-

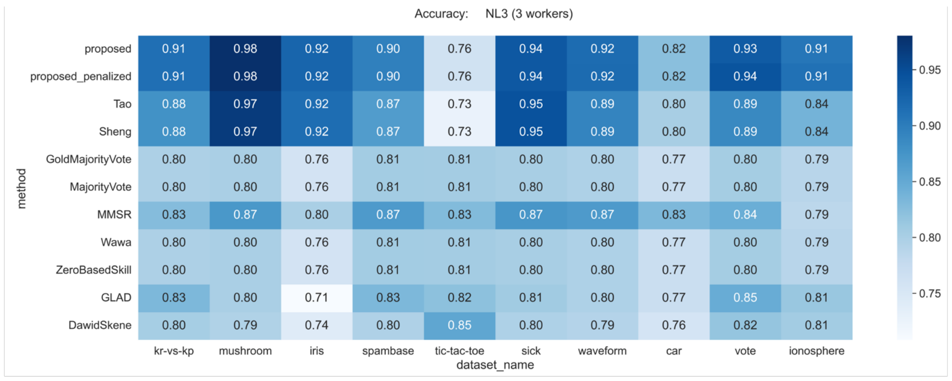

FIGURE 7. Average accuracy of the proposed aggregation techniques compared to the benchmarks for different datasets using three annotators.

veloped a novel label aggregation method that measures annotator reliability based on their consistency and accuracy, in relation to other annotators. We utilized uncertainty estimates to assign each annotator a more accurate weight, which correlates with their agreement with others and their consistency during labeling. In the second approach, we improved our initial strategy by penalizing annotator reliability estimates based on their inconsistencies in labeling. The first part of the proposed algorithm (calculating weights based on consistency) is essential because non-expert annotators often exhibit more irregular consistency during labeling than experts, as they are not trained to identify specific features. This measure helps to differentiate skilled and unskilled annotators. The goal of the second part of the algorithm (penalty for voting against the majority) is to prevent the algorithm from assigning disproportionately high weights to annotators who are consistently incorrect. For example, if annotators consistently mislabel a specific bird species, the second condition penalizes them for their error, despite their consistency. Furthermore, our method reports a single weight for the entire dataset instead of individual weights for each instance. This enables the reuse of calculated weights for future unlabeled test samples without needing to re-acquire labels or retrain classifiers each time new data need labeling. While we have not assessed our method in multi-label scenarios, the proposed techniques are anticipated to perform comparably on multi-label datasets, considering that all steps of the proposed approach involve per-class calculations. Experiments conducted on various crowdsourcing datasets demonstrate that our proposed methods outperform existing techniques in terms of accuracy and variance, especially when there are few annotators available.

## 0.6 Availability of data and materials

The code can be found in crowd-certain

## 0.7 Appendices

### List of abbreviations

### Competing interests

### Acknowledgements

# Crowd-Certain: Uncertainty-Based Weighted Soft Majority Voting with Applications in Crowdsourcing and Ensemble Learning

Jeffrey J. Rodriguez, Ph.D.
The University of Arizona, May 24, 2023

Director:

Crowdsourcing systems have been used to accumulate massive amounts of labeled data for applications such as computer vision and natural language processing. However, because crowdsourced labeling is inherently dynamic and uncertain, developing a technique that can work in most situations is extremely challenging. In this paper, we propose a novel method called "crowd-certain", which provides a more accurate and reliable aggregation of labels, ultimately leading to improved overall performance in both crowdsourcing and ensemble learning scenarios. The proposed method uses the consistency and accuracy of the annotators as a measure of their reliability relative to other annotators. The experimental results show that the proposed technique generates a weight that closely follows the annotator's degree of reliability. Moreover, the proposed method uses the consistency and accuracy of annotators as a measure of their reliability relative to other annotators. Experiments performed on a variety of crowdsourcing datasets indicate that the proposed method outperforms prior methods in terms of accuracy, with significant improvement over all investigated benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene), particularly when few annotators are available.

Supervised learning, crowdsourcing, confidence score, soft weighted majority voting, label aggregation, annotator quality, error rate estimation, multi-class classification, ensemble learning, uncertainty measurement

## 0.8 Introduction

Supervised learning techniques require a large amount of labeled data to train models to classify new data [?, ?]. Traditionally, data labeling has been assigned to experts in the domain or well-trained annotators [?]. Although this method produces high-quality labels, it is inefficient and costly [?, ?]. Social networking provides an innovative solution to the labeling problem by allowing data to be labeled by online crowd annotators. This has become feasible, as crowdsourcing services such as Amazon Mechanical Turk (formerly CrowdFlower) have grown in popularity. Crowdsourcing systems have been used to accumulate large amounts of labeled data for applications such as computer vision [?, ?] and natural language processing [?]. However, because of individual differences in preferences and cognitive abilities, the quality of labels acquired by a single crowd annotator is typically low, thus jeopardizing applications that rely on these data. This is because crowd annotators are not necessarily domain experts and may lack the necessary training or expertise to produce high-quality labels. Aggregation after repeated labeling is one method for handling annotators with various abilities. Label aggregation is a process used to infer an aggregated label for a data instance from a multi-label set [?]. Several studies have demonstrated the efficacy of repeated labeling [?, ?]. Repeat labeling is a technique in which the same data are labeled by multiple annotators, and the results are combined to estimate an aggregated label using majority voting (MV) or other techniques. In the case of MV, an aggregated label is the label that receives the most votes from the annotators for a given data instance. This can help reduce the impact of biases or inconsistencies made by annotators. Several factors, such as problem-specific characteristics, the quality of the labels created by the annotators, and the amount of data available, can influence the effectiveness of the aggregation methodologies. Consequently, it is difficult to identify a clear winner among the different techniques. For example, in binary labeling, one study [?] discovered that Raykar's [?] technique outperformed other aggregation techniques. However, according to another study [?], the traditional Dawid-Skene (DS) model [?] was more reliable in multi-class settings (where data instances can be labeled as belonging to multiple classes). Furthermore, regardless of the aggregation technique used, the performance of many aggregation techniques in real-world datasets remains unsatisfactory [?]. This can be attributed to the complexity of these datasets, which often do not align with the assumptions and limitations of different methods. For example, real-world datasets may present issues such as labeling inaccuracies, class imbalances, or overwhelming sizes that challenge efficient processing with available resources. These factors can adversely affect the effectiveness of label aggregation techniques, potentially yielding less than optimal results for real-world datasets. Prior information may be used to enhance the label aggregation procedure. This can include domain knowledge, the use of quality control measures and techniques that account for the unique characteristics of annotators and data. Knowing the reliability of certain annotators, it is possible to draw more accurate conclusions about labels [?]. For instance, in the label aggregation process, labels produced by more reliable annotators (such as domain experts) may be given greater weight. The results of the label aggregation process can also be validated using expert input [?]. During the labeling process, domain experts can provide valuable guidance and oversight to ensure that the labels produced are accurate and consistent. The agnostic requirement for general-purpose label aggregation is that label aggregation cannot use information outside the labels themselves. This requirement is not satisfied in most label aggregation techniques [?]. The agnostic requirement ensures that the label aggregation technique is as general as possible and applicable to a wide range of domains with minimal or no additional context. The uncertainty of annotators during labeling can provide valuable prior knowledge to determine the appropriate amount of confidence to grant each annotator while still adhering to the requirement of a general-purpose label aggregation technique. We developed a method for estimating the reliability of different annotators based on the annotator's own consistency during labeling and their accuracy with respect to other annotators. We propose a novel method called "crowd-certain", which provides a more accurate aggregation of labels, ultimately leading to improved overall performance in both crowdsourcing and ensemble learning scenarios. The experimental results show that the proposed technique generates a weight that closely follows the annotator's degree of reliability. Furthermore, the proposed method uses the consistency and accuracy of annotators as a measure of their reliability relative to other annotators. Experiments performed on a variety of crowdsourcing datasets indicate that the proposed method outperforms prior methods in terms of accuracy with a significant improvement over all investigated benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene), particularly when few annotators are available.

The remainder of this paper is organized as follows. Section 2 examines related work involving label aggregation algorithms. In Section 3, we provide a detailed explanation of our proposed technique. Section 4 presents the experiments and findings. Finally, Section 5 summarizes the findings and identifies the main directions for future research.

## 0.9 Related Work

Numerous label aggregation algorithms have been developed to capture the complexity of crowdsourced labeling systems, including techniques based on annotator reliability [?, ?], confusion matrices [?, ?], intentions [?, ?], biases [?, ?, ?], and correlations [?]. However, because crowdsourced labeling is inherently dynamic and uncertain, developing a technique that can work in most situations is extremely challenging. Many techniques [?, ?, ?, ?, ?] utilize the Dawid and Skene (DS) generative model [?]. Ghosh [?] extended the DS model by using singular value decomposition (SVD) to calculate the reliability of the annotator. Similarly to Ghosh [?], Dalvi [?]

used SVD to estimate true labels with a focus on the sparsity of the labeling matrix. In crowdsourcing, it is common for the labeling matrix to be sparse, meaning that not all annotators have labeled all the data. This may be due to several factors, such as the cost of labeling all data instances or the annotators' time constraints. Karger [?] described an iterative strategy for binary labeling based on a one-coin model [?]. Karger [?] extends the one-coin model to multi-class labeling by converting the problem into $k-1$ binary problems (solved iteratively), where $k$ is the number of classes. The MV technique assumes that all annotators are equally reliable. For segmentation, Warfield [?] proposed simultaneous truth and performance level estimation (STAPLE), a label fusion method based on expectation maximization. STAPLE "weighs" expert opinions during label aggregation by modeling their reliability. Since then, many variants of this technique have been proposed [?, ?, ?, ?, ?, ?, ?, ?]. The problem with these label aggregation approaches is that they require the computation of a unique set of weights for each sample, necessitating the re-evaluation of the annotators' weights when a new instance is added. Among the numerous existing label aggregation strategies, MV remains the most efficient and widely used approach [?]. If we assume that all annotators are equally reliable and that their errors are independent of one another, then, according to the theory of large numbers, the likelihood that the MV is accurate increases as the number of annotators increases. However, the assumption that all annotators are equally competent and independent may not always hold. Furthermore, MV does not provide any additional information on the degree of disagreement among the annotators (As an example, consider the scenario where four of seven doctors think patient A needs immediate surgery, while all seven think patient B needs immediate surgery; MV will simply label "yes" in both cases). To address this problem, additional measures such as inter-annotator agreement (IAA) have been used [?]. IAA is a measurement of the agreement among multiple annotators who label the same data instance. Typically, IAA is calculated using statistical measures, such as Cohen's kappa, Fleiss's kappa, or Krippendorff's alpha [?]. These measures consider both the observed agreement between the annotators and the expected agreement owing to random chance. IAA can also be visualized using a confusion matrix or annotation heatmap, which illustrates the distribution of labels assigned by the annotators. This can help identify instances where the annotators disagree or are uncertain and can guide further analysis to improve the annotation [?]. Recently, Sheng [?] proposed a technique that provided a confidence score along with an aggregated label. The main problem with this approach is that it assumes that all annotators are equally capable when calculating the confidence score. Tao [?] improved Sheng's approach by assigning different weights to annotators for each instance. This weighting method combines the specific quality $s_\alpha^{(i)}$ for the annotator $\alpha$ and instance $i$ and the overall quality $\tau_\alpha$ across all instances. Inspired by Li's technique [?], Tao evaluates the similarity between the annotator labels for each instance. To derive the specific quality $s_\alpha^{(i)}$, Tao counts the number of annotators who assigned the same label as the annotator $\alpha$ for that instance. To calculate the overall quality $\tau_\alpha$, Tao performs a 10-fold cross-validation to train each of the 10 classifiers on a different subset of data using the labels provided by the annotator $\alpha$ as true labels and then assigns the average accuracy across all remaining instances as $\tau_\alpha$. The final weight for annotator $\alpha$ and instance $i$ is then calculated using the sigmoid function $\gamma_{i,\alpha} = \tau_\alpha \left(1 + \left(s_\alpha^{(i)}\right)^2\right)$. However, Tao's technique [?] has some drawbacks. It relies on the labels of other annotators to estimate $s_\alpha^{(i)}$. However, different annotators have varying levels of competence (reliability) when labeling the data, and therefore, relying on their labels to measure $s_\alpha^{(i)}$ will result in propagating the errors and biases of their labels during weight estimation. Furthermore, Tao's technique [?] relies on the labels provided by each annotator $\alpha$ to estimate their respective $\tau_\alpha$ by assuming that the trained classifiers can learn the inherent characteristics of the datasets even in the absence of ground truth labels. While that may be true in some cases, it typically leads to suboptimal measurement and the propagation of biases and errors, from both the annotator's labels and the classifier, into weight estimation.

## 0.10 Methods

We propose a novel method called “crowd-certain” which focuses on leveraging uncertainty measurements to improve decision-making in crowdsourcing and ensemble learning scenarios. Crowd-Certain employs a weighted soft majority voting approach, where the weights are determined based on the uncertainty associated with each annotator’s labels. Initially, we use uncertainty measurement techniques to calculate the degree of consistency of each annotator during labeling. Furthermore, to ensure that the proposed technique does not calculate a high weight for annotators who are consistently wrong (for example, when a specific annotator always mislabels a specific class, and hence demonstrates a high consistency while having low accuracy), we extend the proposed technique by penalizing the annotators for instances in which they disagree with the aggregated label obtained using the MV. To mitigate the reliance on training a classifier on an annotator’s labels, which may be inaccurate, we train an ensemble of classifiers for each annotator. In addition, we report two confidence scores along with the aggregated label to provide additional context for each calculated aggregate label. We report a single weight for all instances in the dataset. As demonstrated in the experimental sections, the proposed crowd-certain method is not only comparable to other techniques in terms of accuracy for scenarios with a large number of annotators, but also provides a significant improvement in accuracy for scenarios where the number of annotators may be limited. Furthermore, by assigning a single weight to each annotator for all instances in the dataset, the model can assign labels to new test instances without recalculating the annotator weights. This is especially advantageous in situations where annotators are scarce as it enables the model to make accurate predictions with minimal dependence on the annotator input. This characteristic of the crowd-certain method can significantly reduce the time and resources required for labeling in practical applications. When deploying the model in real-world scenarios such as medical diagnosis, fraud detection, or sentiment analysis, it could be advantageous to be able to assign labels to new instances without constantly recalculating annotator weights.

### 0.10.1 Glossary of Symbols

Let us denote the following parameters:

$N$: Number of instances.

$M$: Number of annotators.

$y_k^{(i)} \in \{0, 1\}$: True label for the $k$-th class for instance $i$.

$z_{\alpha,k}^{(i)} \in \{0, 1\}$: Label given by annotator $\alpha$ for $k$-th class for instance $i$.

$\underset{\alpha}{\mathrm{MV}}\left(z_{\alpha,k}^{(i)}\right)$: Majority voting technique (the label that receives the most votes) applied to annotator labels for class $k$ and instance $i$.

$\pi_{\alpha,k}$: Reliability score to generate sample labels for annotator $\alpha$ for class $k$. For example, it may be obtained from a uniform distribution in the interval 0.4 to 1, i.e., $\pi_{\alpha,k} \sim U(0.4, 1)$.

$X^{(i)}$: Data for instance $i$.

$Y^{(i)} = \left\{y_1^{(i)}, y_2^{(i)}, \ldots, y_K^{(i)}\right\}$: True label set, for instance $i$. For example, consider a dataset that is labeled for the presence of cats, dogs, and rabbits in any given instance. If a given instance $X^{(i)}$ has cats and dogs but not rabbits, then $Y^{(i)} = \{1, 1, 0\}$.

$Z_\alpha^{(i)} = \left\{z_{a,1}^{(i)},\ z_{a,2}^{(i)},\ \ldots,\ z_{a,K}^{(i)}\right\}$: Label set given by the annotator $\alpha$ for instance $i$.

$K$: number of categories (aka classes) in a multi-class multi-label problem. For example, if we have a dataset labeled for the presence of cats, dogs, and rabbits in any given instance, then $K = 3$.

$\rho^{(i)}$: Randomly generated number between 0 and 1 for instance $i$. It is obtained from a uniform distribution, i.e., $\rho^{(i)} \sim U(0,1)$ This number will be used to determine, for each instance $i$, whether the true label should be assigned to each fictitious annotator's label. For each class $k$ if the annotator's reliability score for that class $\Pi_{\alpha,k}$ is greater than $\rho^{(i)}$, the true label $y_k^{(i)}$ will be assigned; otherwise, an incorrect label $1 - y_k^{(i)}$ will be assigned.

$\Pi_\alpha = \left\{\pi_{\alpha,1},\ \pi_{\alpha,2},\ \ldots,\ \pi_{\alpha,K}\right\}$: set of $K$ reliability scores for annotator $\alpha$.

$\mathbb{X} = \left\{X^{(i)}\right\}_{i=1}^{N}$: Set of all instances.

$\mathbb{Y} = \left\{Y^{(i)}\right\}_{i=1}^{N}$: Set of all true labels.

$\mathbb{Z}_\alpha = \left\{Z_\alpha^{(i)}\right\}_{i=1}^{N}$: Set of all labels for the annotator $\alpha$.

$\widehat{\mathbb{Y}} = \left\{\widehat{Y}^{(i)}\right\}_{i=1}^{N}$: Set of all aggregated labels.

$\mathbb{P} = \left\{\rho^{(i)}\right\}_{i=1}^{N}$: Set of $N$ randomly generated numbers.

$\mathbb{D} = \{\mathbb{X}, \mathbb{Y}\}$: Dataset containing all instances and all true labels.

$\mathbb{D}_\alpha = \{\mathbb{X}, \mathbb{Z}_\alpha\}$: Dataset containing the labels given by the annotator $\alpha$.

$\mathbb{D}_\alpha^{\text{train}}$, $\mathbb{D}_\alpha^{\text{test}}$: Train and test crowd datasets randomly selected from $\mathbb{D}_\alpha$ where $\mathbb{D}_\alpha^{\text{train}} U \mathbb{D}_\alpha^{\text{test}} = \mathbb{D}_\alpha$ and $\mathbb{D}_\alpha^{\text{train}} \bigcap \mathbb{D}_\alpha^{\text{test}} = \varnothing$

$F_\alpha^{(g)}(\cdot)$: Classifier $g$ trained on dataset $\mathbb{D}_\alpha^{\text{train}}$ with random seed number $g$ (which is also the classifier index)

$P_\alpha^{(i,g)} = \left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}$: Predicted probability set obtained in the output of the classifier $F_\alpha^{(g)}(\cdot)$ representing the probability that each class $k$ is present in the sample.

$\theta^{(g)}_{\alpha,k}$: Binarization threshold. To obtain this, we can utilize any existing thresholding technique (for example, in one technique, we analyze the ROC curve and find the corresponding threshold where the difference between the true positive rate (sensitivity) and false positive rate (1-specificity) is maximum; Alternatively, we could simply use 0.5).

$t^{(i,g)}_{\alpha,k} = \begin{cases} 1 & \text{if } p^{(i,g)}_{\alpha,k} \geq \theta^{(g)}_{\alpha,k}, \\ 0 & \text{otherwise.} \end{cases}$: Predicted label obtained by binarizing $p^{(i,g)}_{\alpha,k}$.

$\eta^{(i)}_{\alpha,k} = \underset{g}{\text{MV}}\left(t^{(i,g)}_{\alpha,k}\right)$: The output of the majority vote applied to the predicted labels obtained by the $G$ classifiers.

$u^{(i)}_{\alpha,k}$: Uncertainty score.

$c^{(i)}_{\alpha,k}$: Consistency score.

$\omega_{\alpha,k}$: Estimated weight for annotator $\alpha$ and class $k$.

$\nu^{(i)}_{k} = \frac{1}{M} \sum_{\alpha} \omega_{\alpha,k} \, \eta^{(i)}_{\alpha,k}$: Final aggregated label for class $k$ and instance $i$.

### 0.10.2 Risk Calculation

Label aggregation is frequently used in various machine learning tasks, such as classification and regression, when multiple annotators assign labels to the same data points. The aggregation model refers to the underlying function that maps a set of multiple labels obtained by different annotators, into one aggregated label. In the context of label aggregation, this model can be a neural network, a decision tree, or any other machine learning algorithm capable of learning to aggregate labels provided by multiple annotators. The objective of this study is to develop an aggregation model capable of accurately determining true labels despite potential disagreements among annotators. One common method to achieve this involves minimizing the total error (or disagreement) between the annotators' assigned labels and the true labels, as follows:

$$E = \sum_{i=1}^{N} \sum_{a=1}^{M} \left( \sum_{k=1}^{K} \delta \left( y_k^{(i)}, z_{\alpha,k}^{(i)} \right) \right) \tag{0.10.1}$$

where $\delta$ is the Kronecker delta function.

Although error is a crucial aspect in determining the aggregation model's performance, it treats false positives and false negatives with equal weight. However, in many practical scenarios, it is essential to weigh false positives and false negatives differently depending on the specific context and potential consequences of each type of misclassification. The concept of risk allows us to achieve this by incorporating a loss function, which assigns different weights to different types of errors. In this way, risk serves as a weighted calculation of error, enabling us to better evaluate the performance of an aggregation model and its generalization capability.

Let us denote loss function, $\mathcal{L}(\cdot)$, as a function that quantifies the discrepancy between the predicted labels and the true labels, accounting for the varying importance of different types of errors. Risk, denoted as $R(h)$, represents the expected value of a loss function over the entire dataset, capturing the performance of the aggregation model on all possible data instances. In practice, our goal is to minimize the risk to achieve optimal performance on unseen data. However, since we only have access to a limited dataset (empirical distribution), we instead work with the empirical risk. This limitation may arise because of the need to reserve a portion of our data for testing and validation or because no dataset can fully capture all possible data instances in the real world. However, minimizing risk alone could result in overfitting, in which the aggregation model learns the noise in the training data rather than the underlying patterns, resulting in poor generalization to unseen data. To improve generalizability, it is necessary to employ regularization techniques to strike a balance between the complexity of the aggregation model and its ability to fit the training data.

Risk measurement enables us to assess the aggregation model's performance in terms of accuracy, overfitting (when risk is minimized but the model performs poorly on unseen data), and model complexity.

Assume that the aggregation model $h(\cdot)$ is a function that takes a set of $M$ label sets $Z^{(i)}$ for each instance $i$ in the training data and calculates an aggregated label set $\widehat{Y}^{(i)}$ as an estimate of the true label set $Y^{(i)}$. Our goal is to find an aggregation model $h(\cdot)$ that minimizes risk as follows:

$$R(h) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(Y^{(i)}, h\left(\left\{Z_{\alpha}^{(i)}\right\}_{\alpha=1}^{M}\right)\right) \tag{0.10.2}$$

In this context, $\mathcal{L}(\cdot)$ represents an arbitrary loss function, which quantifies the discrepancy between predicted labels and true labels while accounting for the varying importance of different types of errors.

Our goal is to choose an aggregation model $\widehat{h}$ that minimizes the risk, following the principle of risk minimization [?], as shown below:

$$\widehat{h} = \underset{h}{\text{argmin}}, R(h) \tag{0.10.3}$$

### 0.10.3 Generating Annotators' Label Sets from Ground Truth

In order to evaluate the proposed crowd-certain technique (with and without penalization) as well as other aggregation techniques, we create $M$ fictitious annotators. To synthesize a multi-annotator dataset from a dataset with existing ground truth, we use a uniform distribution in the interval from 0.4 to 1, i.e., $\pi_{\alpha,k} \sim U\,(0.4, 1)$ (however other ranges can also be used) to obtain $M \times K$ reliability values $\Pi$, where $K$ is the number of classes. (Note that an annotator may be skilled at labeling dogs, but not rabbits.) Then we use these reliability values to generate the crowd label set $Z_{\alpha}^{(i)}$ from the ground truth labels for each instance $i$.

For each annotator $\alpha$, each instance $i$ and class $k$ in the dataset is assigned its true label with probability $\pi_{\alpha,k}$ and the opposite label with probability $(1-\pi_{\alpha,k})$. To generate the labels for each annotator $\alpha$, a random number $0 < \rho^{(i)} < 1$ is generated for each instance $i$ in the dataset. Then $\forall \alpha, k$ if $\rho^{(i)} \leq \pi_{\alpha,k}$ then the true label is used for that instance and class for the annotator $\alpha$; otherwise, the incorrect label is used.

The calculated annotator labels $z^{(i)}_{\alpha,k}$ for each annotator $\alpha$, instance $i$ and class $k$ are as follows:

$$z^{(i)}_{\alpha,k} = \begin{cases} y^{(i)}_k & \text{if } \rho^{(i)} \leq \pi_{\alpha,k}, \\ 1 - y^{(i)}_k & \text{if } \rho^{(i)} > \pi_{\alpha,k}, \end{cases} \quad \forall i, a, k \tag{0.10.4}$$

To evaluate the proposed techniques over all data instances, a k-fold cross-validation is employed.

### 0.10.4 Uncertainty Measurement

A common approach to measure uncertainty is to increase the number of data instances $X$ in the test dataset $\mathbb{D}^{\text{test}}_{\alpha}$ to create multiple variations of each sample data $X^{(i)}$ [?]. In this approach, for each instance $i$, we apply randomly generated spatial transformations and additive noise to the input data $X^{(i)}$ to obtain a transformed sample and repeat this process $G$ times to obtain a set of $G$ transformed samples. However, this approach is mostly suitable for cases where the input data is images or volume slices. Since the datasets used in this study consist of feature vectors instead of images or volume slices, this approach cannot be used. To address this problem, we introduced a modified uncertainty measurement approach, in which instead of augmenting the data instances $X^{(i)}$, we feed the same sample data to different classifiers. For the choice of classifier, we can either use a probability-based classifier such as random forest and train it under $G$ different random states or train various classifiers and address the problem in a manner similar to ensemble learning (using a set of $G$ different classification techniques such as random forest, SVM, CNN, Adaboost, etc.). In either case, we obtain a set of $G$ classifiers $\left\{F^{(g)}_{\alpha}(\cdot)\right\}_{g=1}^{G}$ for each annotator

$\alpha$. The classifier $F_{\alpha}^{(g)}(\cdot)$ is a pre-trained or pre-designed model that has been trained on a labeled training dataset $\mathbb{D}_{\alpha}^{\text{train}}$. This training process enables $F_{\alpha}^{(g)}(\cdot)$ to learn the underlying patterns in the data and make predictions on unseen instances. After training, we feed the test samples $X^{(i)} \in \mathbb{X}^{\text{test}}$ to the $g$-th classifier $F_{\alpha}^{(g)}(\cdot)$ as test cases. The classifier $F_{\alpha}^{(g)}(\cdot)$ then outputs a set of predicted probabilities $\left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}$ representing the probability that class $k$ is present in the sample. Consequently, we obtain a collection of $G$ predicted probability sets $\left\{\left\{p_{\alpha,k}^{(i,g)}\right\}_{k=1}^{K}\right\}_{g=1}^{G}$ for each annotator $\alpha$ and instance $i$. The set $\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}$ contains the predicted probabilities for class $k$, annotator $\alpha$, and instance $i$. Disagreements between predicted probabilities $\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}$ can be used to estimate uncertainty. The reason for using classifiers rather than using the crowdsourced labels directly is two-fold. Using a probabilistic classifier helps us calculate uncertainty based on each annotator's labeling patterns that the classifier learns. Furthermore, this approach provides us with a set of pre-trained classifiers $\left\{\left\{F_{\alpha}^{(g)}(\cdot)\right\}_{g=1}^{G}\right\}_{a=1}^{M}$ that can be readily utilized on any new data instances without the need for those samples to be labeled by the original annotators. The index value $g \in \{1, 2, \ldots, G\}$ is used as the random seed value during training of the $g$th classifier for all annotators. Define $t_{\alpha,k}^{(i,g)}$ as the predicted label obtained by binarizing the predicted probabilities $p_{\alpha,k}^{(i,g)}$ using the threshold $\theta_{\alpha,k}^{(g)}$ as shown in the Glossary of Symbols section. Uncertainty measures are used to quantify the level of uncertainty or confidence associated with the predictions of a model. In this work, we need to measure the uncertainty $u_{\alpha,k}^{(i)}$ associated with the model predictions. Some common uncertainty measurement measures are as follows.

*Entropy* Entropy is a widely used measure of uncertainty in classification problems. In an ensemble of classifiers, entropy serves as a quantitative measure of the uncertainty or disorder present in the probability distribution of the predicted class labels. A higher entropy value indicates a greater degree of uncertainty in the predictions, as the predictions of the individual classifiers in the ensemble are significantly different. In contrast, a lower entropy value denotes reduced uncertainty as the ensemble assigns very similar probabilities to a particular class, indicating strong agreement among the classifiers and increased confidence in their collective prediction. The formula for calculating entropy is as follows:

$$u_{\alpha,k}^{(i)} = H\left(\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right) = -\sum_{g} p_{\alpha,k}^{(i,g)} \log\left(p_{\alpha,k}^{(i,g)}\right) \tag{0.10.5}$$

*Standard Deviation* In regression problems, standard deviation is often used to quantify uncertainty. It measures the dispersion of predicted values around the mean. A greater standard deviation indicates greater uncertainty of the prediction. For a set of predicted values $\{t_{\alpha,k}^{(i,g)}\}_{g=1}^{G}$ with mean value $\mu$, the standard deviation is defined as

$$u_{\alpha,k}^{(i)} = \mathrm{SD}\left(\left\{t_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right) = \sqrt{\frac{1}{G-1}\sum_{g=1}^{G}\left(t_{\alpha,k}^{(i,g)} - \mu\right)^2}, \quad \mu = \frac{1}{G}\sum_{g=1}^{G} t_{\alpha,k}^{(i,g)} \tag{0.10.6}$$

**Predictive Interval:** A predictive interval provides a range within which a future observation is likely to fall with a certain level of confidence. For example, a 95% predictive interval indicates that there is a 95% likelihood that the true value falls within that range. A greater uncertainty corresponds to wider intervals. In the context of multiple classifiers, the predictive intervals can be calculated by considering the quantiles of the classifier output. For a predefined confidence level $\gamma$ (e.g., 95%), for a specific class $k$, we need to find the quantiles $Q_L^k$ and $Q_U^k$ of the probability distribution of class $k$ predicted by the $G$ classifiers. The uncertainty can be represented by the width of the predictive interval:

$$\begin{gathered} P\left(Q_L^k \leq p_{\alpha,k}^{(i,g)} \leq Q_U^k\right) = \gamma \\ u_{\alpha,k}^{(i)} = Q_L^k - Q_U^k \end{gathered} \tag{0.10.7}$$

The steps to calculate the predictive interval are as follows:

1. Collect the class $k$ probabilities predicted by all $G$ classifiers for a given instance. Then sort the values in ascending order. Let us call this set $P_{\alpha,k}^{(i)} = \text{sorted}\left(\left\{p_{\alpha,k}^{(i,g)}\right\}_{g=1}^{G}\right), \quad \forall \alpha, k, i.$

2. Calculate the lower and upper quantile indices based on the chosen confidence level $\gamma$. The lower quantile index is $L = \text{ceil}\left(\frac{G}{2}(1-\gamma)\right)$, and the upper quantile index is $U = \text{floor}\left(\frac{G}{2}(1+\gamma)\right)$, where ceil and floor are the ceiling and floor functions, respectively.

3. Find the values corresponding to the lower and upper quantile indices in the sorted $P_{\alpha,k}^{(i)}$. These values are the lower and upper quantiles $Q_L^k$ and $Q_U^k$.

4. Now we have the predictive interval $P\left(Q_L^k <= p_{\alpha,k}^{(i,g)} <= Q_U^k\right) = \gamma$, where $Q_L^k$ and $Q_U^k$ represent the bounds of the interval containing the $\alpha$ proportion of the probability mass.

*Monte Carlo Dropout* The Monte Carlo dropout [?] can be used to estimate uncertainty in neural networks by applying the dropout at test time. Multiple forward passes with dropout generate a distribution of predictions from which uncertainty can be derived using any of the aforementioned techniques (standard deviation, entropy, etc.).

*Bayesian Approaches* Bayesian methods offer a probabilistic framework to estimate the parameters of the model and make predictions. These methods explicitly model uncertainty by considering prior beliefs about the model parameters and then updating those beliefs based on the observed data. In Bayesian modeling, the model parameters are treated as random variables and a posterior distribution is estimated using these parameters. The following are two common Bayesian approaches for measuring the uncertainty in classification problems.

- **Bayesian model averaging (BMA):** BMA accounts for model uncertainty by combining the predictions of various models using their posterior probabilities as weighting factors. Instead of selecting a single "best" model, BMA acknowledges the possibility of multiple plausible models, each with its own strengths and weaknesses [?]. The steps to implement BMA are as follows. Select a set of candidate models that represent different hypotheses regarding the data-generating process underlying the data. These models may be of various types, such as linear regression, decision trees, neural networks, or any other model suited to the specific problem at hand. Using the available data, train each candidate model. Calculate the posterior probabilities of the models. Using the posterior probabilities of each model as weights, calculate the weighted average of each model's predictions. The weighted average is the BMA prediction for the input instance and class.

- **Bayesian neural networks (BNNs):** BNNs [?] are an extension of conventional neural networks in which the weights and biases of the network are treated as random variables. The primary distinction between BNNs and conventional neural networks is that BNNs model uncertainty directly in the weights and biases. The posterior distributions of the network weights and biases (learned during training) capture the uncertainty, which can then be utilized to generate predictive distributions for each class. This enables multiple predictions to be generated by sampling these predictive distributions, which can be used to quantify the uncertainty associated with each class.

*Committee-Based Methods* Committee-based method [?] involves training multiple models (a committee) and aggregating their predictions. The disagreement between committee members' predictions can be used as a measure of uncertainty. Examples include bagging and boosting ensemble methods and models, such as random forests.

$$u_{\alpha,k}^{(i)} = \text{VarCommittee}\left(P_{\alpha,k}^{(i)}\right) = \frac{1}{G}\sum_{g=1}^{G}\left(p_{\alpha,k}^{(i,g)} - \mu\right)^2, \quad \mu = \frac{1}{G-1}\sum_{g=1}^{G} p_{\alpha,k}^{(i,g)} \qquad (0.10.8)$$

*Conformal Prediction:* Conformal prediction [?] is a method of constructing prediction regions that maintain a predefined level of confidence. These regions can be used to quantify the uncertainty associated with the prediction of a model.

Steps to calculate the nonconformity score:

[1.]For each classifier $g$ and each class $k$, calculate the nonconformity score. Here, score_function measures the conformity of the prediction with the true label. In the context of this study, the true label can be replaced by $\eta_{\alpha,k}^{(i)}$. A common choice for score_function is the absolute difference between the predicted probability and the true label, but other options can be used depending on the specific problem and requirements. Define the nonconformity score as $\zeta_k^g = \text{score\_function}\left(p_{\alpha,k}^{(i,g)}, y_k^{(i)}\right)$ Calculate the p-value $\text{pv}_k$ for each class $k$ as the proportion of classifiers with nonconformity scores greater than or equal to a predefined threshold $\text{T}_k$ : $\text{p-values}(k) = \frac{|\{g:\, \zeta_k^g \geq T_k\}|}{G}$ The p-values calculated for each class $k$ represent the uncertainty associated with that class. A higher p-value indicates a higher level of agreement among the classifiers for a given class, whereas a lower p-value suggests greater uncertainty or disagreement.

The uncertainty measures discussed above are only some of the available options. Selecting an appropriate measure depends on factors such as the problem domain, the chosen model, and the specific requirements of a given application. For this study, we use the variance technique shown in Equation (??) as our uncertainty measurement due to its simplicity. However, other measures could also be employed as suitable alternatives.

### 0.10.5 Crowd-Certain: Uncertainty-Based Weighted Soft Majority Voting

*Consistency Measurement* Define $c_{\alpha,k}^{(i)}$ as the consistency score for annotator $\alpha$, class $k$ and instance $i$. We calculate this consistency score using the uncertainty score $u_{\alpha,k}^{(i)}$ explained in the previous section. We use two approaches to calculate $c_{\alpha,k}^{(i)}$ from $u_{\alpha,k}^{(i)}$.

[1.]The first approach is to simply subtract the uncertainty from 1 as follows:

$$c_{\alpha,k}^{(i)} = 1 - u_{\alpha,k}^{(i)} \quad , \ \forall i, \alpha, k \tag{0.10.9}$$

In a second approach (shown in Equation (**??**)), we penalize annotators for instances in which their predicted label $\eta_{\alpha,k}^{(i)}$ (explained in the Glossary of Symbols section) does not match the MV of all annotator labels $\underset{\alpha}{\mathrm{MV}}\left(z_{\alpha,\,k}^{(i)}\right)$. As previously discussed, instead of directly working with the annotator's labels $z_{\alpha,k}^{(i)}$, we use the predicted labels obtained from the ensemble of classifiers $\eta_{\alpha,k}^{(i)}$. This methodology does not require repeating the crowd-labeling process for new data samples. In particular, we are likely not to have access to the same crowd of annotators employed in the training dataset.

$$c_{\alpha,k}^{(i)} = \begin{cases} 1 - u_{\alpha,k}^{(i)} & \text{if } \eta_{\alpha,k}^{(i)} = \mathrm{MV}_{\alpha}(\eta_{\alpha,k}^{(i)}) \\ 0 & \text{otherwise} \end{cases} \tag{0.10.10}$$

*Weight Measurement* Furthermore, we define the annotators' weights $\omega_{\alpha,k}$ for each class k as the mean of their respective consistency scores $c_{\alpha,k}^{(i)}$ over all instances. The weights are normalized between 0 and 1 as follows:

$$\omega_{\alpha,k} = \frac{\psi_{\alpha,k}}{\sum_{\alpha=1}^{M} \psi_{\alpha,k}} \quad \text{where} \quad \psi_{\alpha,k} = \frac{1}{N} \sum_{i=1}^{N} c_{\alpha,k}^{(i)} \tag{0.10.11}$$

*Aggregated Label Calculation* Finally, the aggregated label $v_k^{(i)}$ for each instance $i$ and class $k$ is the weighted average of the predicted labels $\eta_{\alpha,k}^{(i)}$ for each annotator $\alpha$:

$$v_k^{(i)} = \begin{cases} 1 & \text{if } \left(\sum_{\alpha=1}^{M} \omega_{\alpha,k}\, \eta_{\alpha,k}^{(i)}\right) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, k \tag{0.10.12}$$

*Confidence Score Calculation* In previous section we showed how to calculate the aggregated label $v_k^{(i)}$ (shown in Equation (**??**)). Define $F_k^{(i)}$ as the confidence score for instance $i$ and class $k$. In this section, we calculate two confidence scores $F_k^{(i)}$, based on how many different annotators agree on the reported label $v_k^{(i)}$, which will be reported alongside the aggregated label $v_k^{(i)}$. The confidence scores show the level of confidence we should place on the aggregated labels.

To calculate this confidence score, we modify the two techniques used by Sheng [**?**] and Tao [**?**] to incorporate our calculated weight $\omega_{\alpha,k}$ shown in Equation (**??**) for each worker $\alpha$.

[1.]**uwMV-Freq:** In this approach, the confidence score $F^{(i)}$ is defined as the weighted sum of labels belonging to annotators whose label is the same as the final aggregated label. It is calculated as

$$F_k^{(i)} = \sum_{\alpha=1}^{M} \omega_{\alpha,k}\, \delta\left(\eta_{\alpha,k}^{(i)} \;,\; v_k^{(i)}\right) \tag{0.10.13}$$

where $\delta$ is the Kronecker delta function. **uwMV-Beta:** In this approach, the CDF of the beta distribution at the decision threshold of 0.5 is used to calculate a confidence score $F^{(i)}$. To calculate the two shape parameters of the beta distributions $\alpha^{(i)}$ and $\beta^{(i)}$, we use a weighted sum of all correct and incorrect aggregated labels, respectively:

$$\begin{aligned} l_k^{(i)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha,k}\, \delta\left(\eta_{\alpha,k}^{(i)}, v_k^{(i)}\right) \\ u_k^{(i)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha,k}\, \delta\left(\eta_{\alpha,k}^{(i)}, 1 - v_k^{(i)}\right) \end{aligned} \tag{0.10.14}$$

$$F_k^{(i)} = I_{0.5}\left(l_k^{(i)}, u_k^{(i)}\right) = \sum_{t=\lfloor l_k^{(i)} \rfloor}^{T-1} \frac{(T-1)!}{t!(T-1-t)!} 0.5^{T-1} \tag{0.10.15}$$

where $T = \left\lfloor l_k^{(i)} + u_k^{(i)} \right\rfloor$ and $\lfloor \cdot \rfloor$ is the floor function.

## 0.11 Results

In this section, we assess the efficacy of our proposed strategy. To evaluate our proposed technique, we conducted a series of experiments comparing the proposed technique with existing state-of-the-art techniques such as MV, Tao [?], and Sheng [?], as well as with other crowdsourcing methodologies reported in the crowd-kit package [?] including Gold Majority Voting, MMSR [?], Wawa, Zero-Based Skill, GLAD [?], and Dawid Skene [?].

### 0.11.1 Datasets:

We report the performance of our proposed techniques on various datasets. These datasets cover a wide range of domains and have varying characteristics in terms of the number of features, samples, and class distributions. Table **??** provides an overview of the datasets used. All datasets are obtained from the University of California, Irvine (UCI) repository [?].

TABLE 4. Descriptions of the datasets used.

| **Dataset** | **#Features** | **#Samples** | **#Positives** | **#Negatives** |
|---|---|---|---|---|
| kr-vs-kp | 36 | 3196 | 1669 | 1527 |
| mushroom | 22 | 8124 | 4208 | 3916 |
| iris | 4 | 100 | 50 | 50 |
| spambase | 58 | 4601 | 1813 | 2788 |
| tic-tac-toe | 10 | 958 | 332 | 626 |
| sick | 30 | 3772 | 231 | 3541 |
| waveform | 41 | 5000 | 1692 | 3308 |
| car | 6 | 1728 | 518 | 1210 |
| vote | 16 | 435 | 267 | 168 |
| ionosphere | 34 | 351 | 126 | 225 |

- The **kr-vs-kp** dataset represents the King Rook-King Pawn on a7 in chess. The positive class indicates a victory for white (1,669 instances, or 52%), while the negative class indicates a defeat for white (1,527 instances, 48%).

- The **mushroom** dataset is based on the Audubon Society Field Guide for North American Mushrooms (1981) and includes 21 attributes related to mushroom characteristics such as cap shape, surface, odor, and ring type.
- The **Iris** Plants Dataset comprises three classes, each with 50 instances, representing different iris plant species. The dataset contains four numerical attributes in centimeters: sepal length, sepal width, petal length, and petal width.
- The **Spambase** dataset consists of 57 attributes, each representing the frequency of a term appearing in an email, such as the "address".
- The **tic-tac-toe** endgame dataset encodes all possible board configurations for the game, with "x" playing first. It contains nine attributes corresponding to the tic-tac-toe squares: x, o, and b (blank).
- The **Sick** dataset includes thyroid disease records from the Garvan Institute and J. Ross Quinlan of the New South Wales Institute in Sydney, Australia. 3,772 instances with 30 attributes (seven continuous and 23 discrete) and 5.4% missing data. Age, pregnancy, TSH, T3, TT4, etc.
- The **waveform** dataset generator comprises 41 attributes and three wave types, with each class consisting of two "base" waves.
- The **Car** Evaluation Dataset rates cars on price, buying, maintenance, comfort, doors, capacity, luggage, boot size, and safety using a simple hierarchical decision model. The dataset consists of 1,728 instances categorized as unacceptable, acceptable, good, and very good.
- The 1984 US Congressional **Voting** Records Dataset shows how members voted on 16 CQA-identified critical votes. Votes are divided into nine categories, simplified to yea, nay, or unknown disposition. The dataset has two classes: Democrats (267) and Republicans (168).

- The Johns Hopkins **Ionosphere** dataset contains data collected near Goose Bay, Labrador, using a phased array of 16 high-frequency antennas. “Good” radar returns show ionosphere structure, while “bad” returns are ionosphere-free. The dataset includes 351 instances with 34 attributes categorized as good or bad.

All datasets were transformed into a two-class binary problem for comparison with existing benchmarks. For instance, only the first and second classes were used in the “waveform” dataset, and the first two classes were utilized in the “Iris” dataset. In this study, we generated multiple fictitious label sets for each dataset to simulate the crowdsourcing concept of collecting several crowd labels for each instance. This was achieved by selecting random samples in the datasets using a uniform distribution and altering their corresponding true labels to incorrect ones, while maintaining the original distribution of the ground-truth labels. The probability of each instance containing the correct true label was determined using a uniform distribution, allowing us to create synthetic label sets for each annotator that preserved the underlying structure and difficulty of the original classification problem. By creating datasets with varying levels of accuracy, we aim to evaluate the performance of our proposed method under different conditions, such as varying annotator expertise and reliability. This process allowed us to assess the ability of our method to handle diverse real-world crowdsourcing scenarios and gain insight into its general applicability and effectiveness in improving overall classification accuracy.

### 0.11.2 Benchmarks:

Tao [?] and Sheng [?] techniques were implemented in Python to evaluate their performance. Furthermore, the crowd-kit package (A General-Purpose Crowdsourcing Computational Quality Control Toolkit for Python) [?] was used to implement the remaining benchmark techniques, including Gold Majority Voting, MMSR [?], Wawa, Zero-Based Skill, GLAD [?], and Dawid Skene [?].

- **Golden majority voting** estimates the probability that each annotator possesses the correct label and then calculates the probability of each label per occurrence based on the weight assigned to each label. Assume that 10,000 jobs are performed by 3,000 annotators as well as 100 instances of ground truth. First, the percentage of correct labels for each annotator is calculated. The remaining labels were then estimated based on the weights.

- **Wawa** (Annotator Agreement with Aggregate), also referred to as "inter-rater agreement", is a commonly used statistic for non-testing problems. This indicates the average frequency with which each annotator's response matches the aggregate response for each instance.

- **Zero-Based-Skill** employs a weighted majority vote (WMV). After processing a collection of instances, it re-evaluated the abilities of the annotators based on the accuracy of their responses. This process is repeated until the labels no longer change or the maximum number of iterations is reached.

- Descriptions of the other techniques can be found in their respective references.

### 0.11.3 Weight Measurement:

After generating the multi-label sets, we employed both the proposed and state-of-the-art approaches to obtain the aggregated labels. We experimented with two approaches for classifier selection, as explained in Section 3.4.1. We found no significant differences in the overall outcomes and thus chose the second approach, which utilized the Random Forest classification technique, to save processing time and reduce the number of required Python package dependencies. Ten Random Forests, each with four trees and a maximum depth of four, were trained in different random states for each annotator $\alpha$, as detailed in Section 3.

**Annotators' reliability vsėstimated weight $\omega_{\alpha,k}$**

Figure **??** depicts the relationship between the randomly assigned annotators' reliability value ($\pi_{\alpha,k}$) and their corresponding estimated weights, $\omega_{\alpha,k}$. In the case of Tao's method, the figure displays the average weights across all instances. As seen, when the reliability of an annotator surpasses a specific threshold, the weight measured by Tao's technique plateaus, whereas the proposed method exhibits a considerably stronger correlation. Individual data points represent the actual measured weights, and the curve represents the regression line.

### 0.11.4 Confidence-score

The results present a box plot of the average accuracy for different numbers of annotators, ranging from three to ten. Figure **??** illustrates the average accuracy of the crowd-certain technique with penalization for both the "freq" and "Beta" confidence measurement approaches. A noticeable difference in the accuracy was observed. However, statistical analysis did not reveal a significant difference between these approaches.

Figure **??** displays the average accuracy using the "freq" confidence measurement strategy for the proposed crowd-certain technique with and without penalization. The penalization method occurs by penalizing annotators for inaccurate labeling before measuring their weights, as demonstrated in Equation (**??**). The penalized version of the proposed technique shows an improvement in average accuracy and a reduction in variance.

Figure **??** shows the average accuracy distribution ("freq" strategy) of the proposed penalized versus Tao and Sheng for a different number of annotators using the kernel density estimation technique. This demonstrates that the proposed technique outperforms both Tao and Sheng on nine of the ten datasets, with higher average accuracy and less fluctuation over different annotator counts. Table **??** shows the statistical

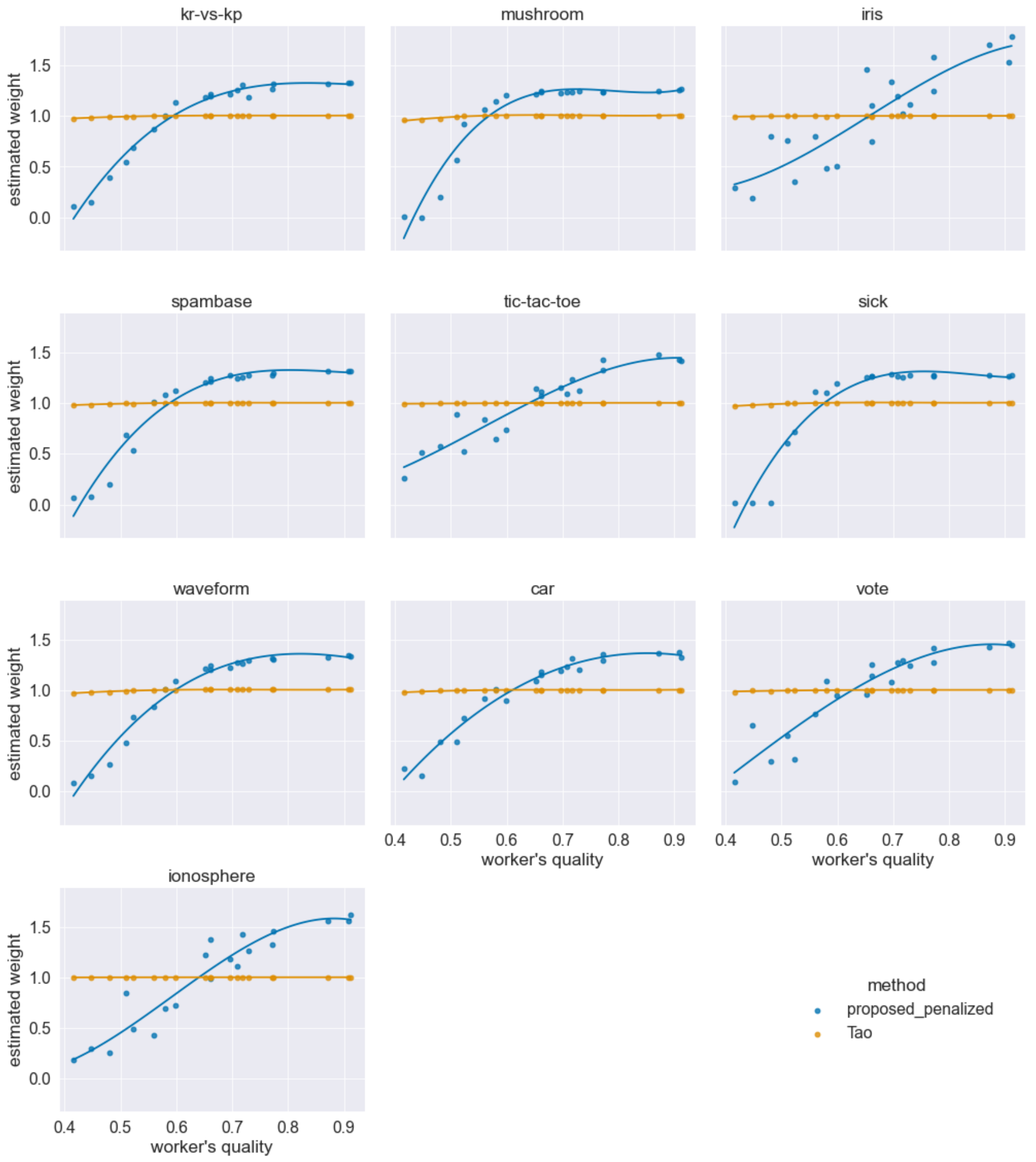

FIGURE 8. Comparison of estimated weight with respect to annotators' degree of reliability for the proposed aggregation technique "proposed-penalized" and Tao [?] for 10 different datasets.

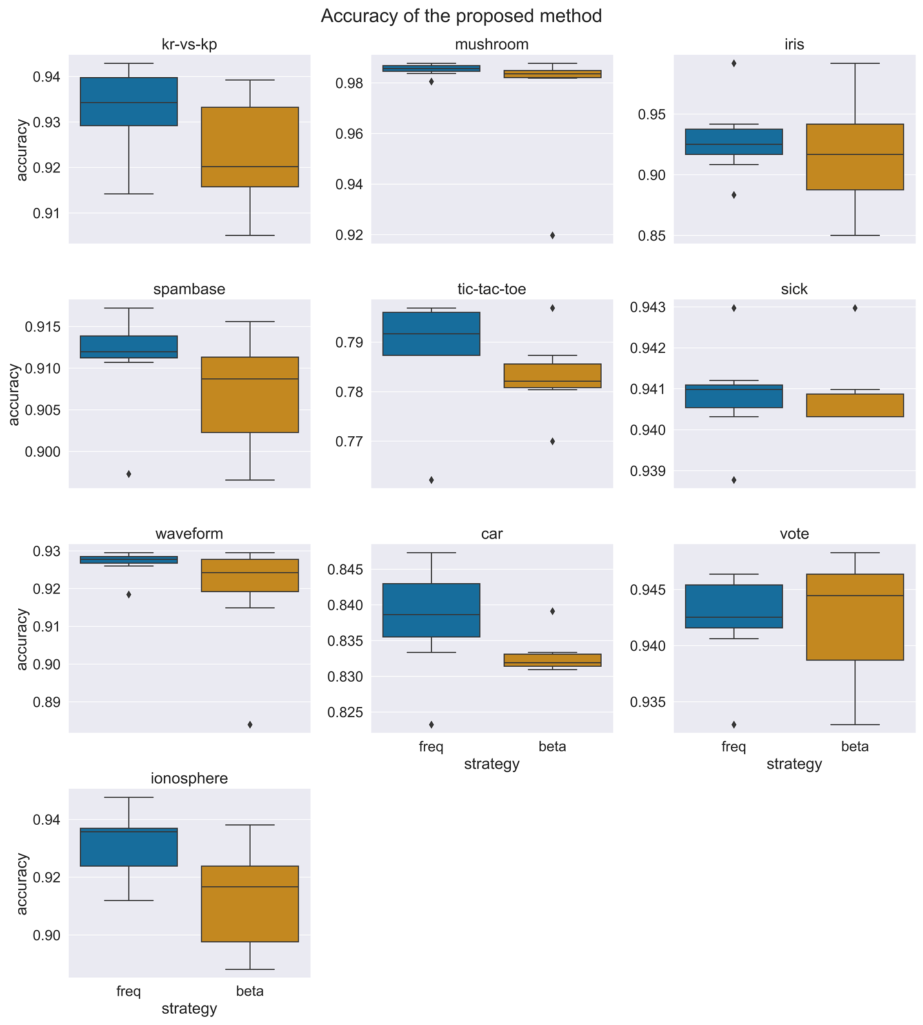

FIGURE 9. Comparison of the measured average accuracy for the two confidence-score measurement techniques in ten different datasets (using the proposed crowd-certain technique with penalization) in different numbers of annotators (from 3 up to 10).

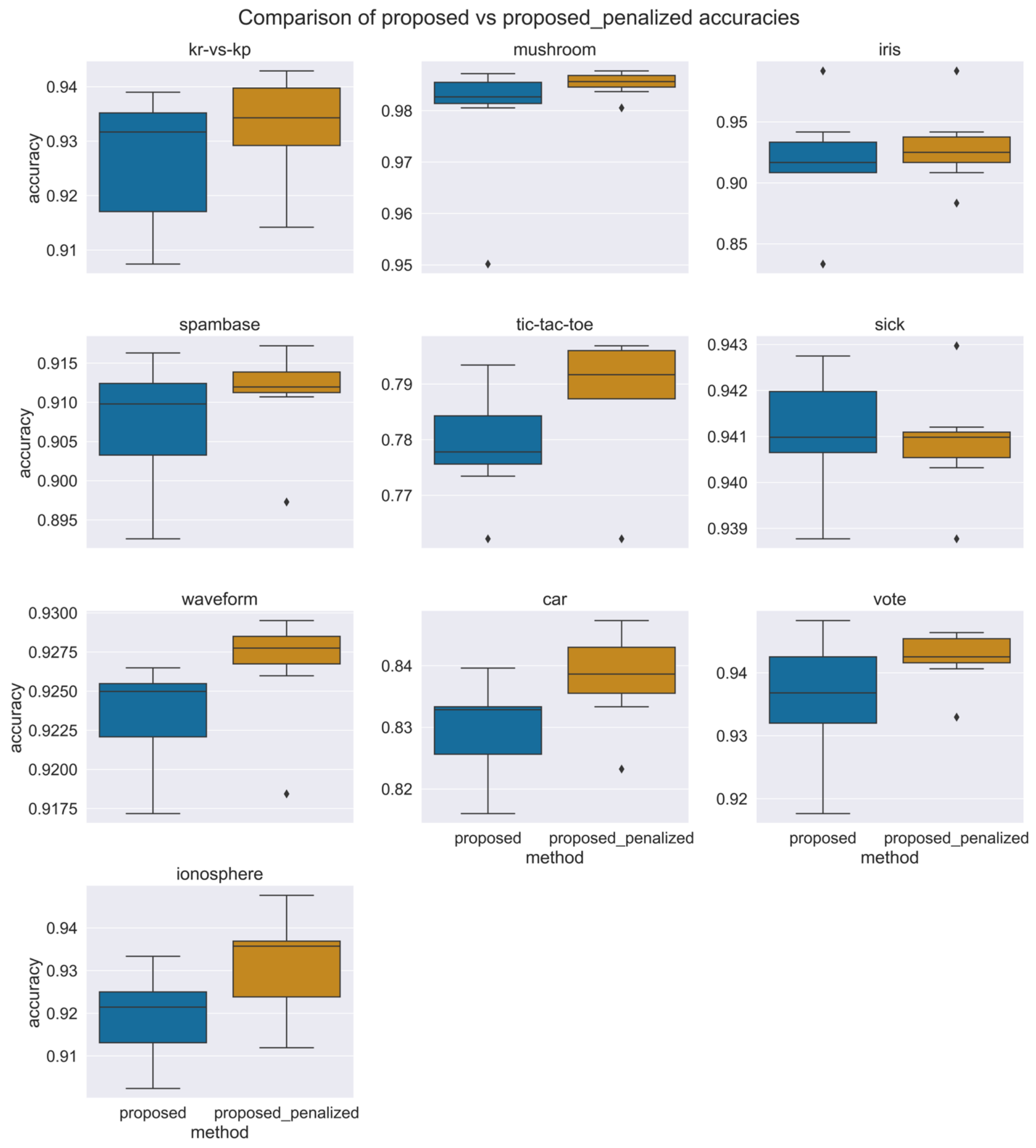

Figure 10. Comparison of the measured average accuracy for the proposed crowd-certain technique with and without penalization and using the 'freq' confidence-score strategy on ten different datasets with different numbers of annotators (starting from 3 to 10).

Table 5. Statistical tests between the proposed-penalized technique and Tao [?] for the 'freq' confidence measurement strategy.

| Independent t-test | Diff | Degrees of freedom | t | 2-sided p-value | Diff<0 p-value | Diff>0 p-value | Cohen d | Hedge's g | Glass's delta | Pearson r |
|---|---|---|---|---|---|---|---|---|---|---|
| Kr-vs-kp | -0.044 | 12 | -6.171 | 0 | 0 | 1 | -3.299 | -3.088 | -2.743 | 0.872 |
| mushroom | -0.012 | 12 | -2.781 | 0.017 | 0.008 | 0.992 | -1.487 | -1.392 | -1.076 | 0.626 |
| iris | -0.012 | 12 | -0.506 | 0.622 | 0.311 | 0.689 | -0.271 | -0.253 | -0.226 | 0.145 |
| spambase | -0.031 | 12 | -3.691 | 0.003 | 0.002 | 0.998 | -1.973 | -1.847 | -1.458 | 0.729 |
| tic-tac-toe | -0.036 | 12 | -4.612 | 0.001 | 0 | 1 | -2.466 | -2.308 | -2.156 | 0.8 |
| sick | 0.002 | 12 | 0.294 | 0.774 | 0.613 | 0.387 | 0.157 | 0.147 | 0.112 | 0.085 |
| waveform | -0.025 | 12 | -5.118 | 0 | 0 | 1 | -2.736 | -2.561 | -2.022 | 0.828 |
| car | -0.008 | 12 | -0.779 | 0.451 | 0.226 | 0.774 | -0.416 | -0.39 | -0.309 | 0.219 |
| vote | -0.033 | 12 | -5.352 | 0 | 0 | 1 | -2.861 | -2.678 | -2.112 | 0.84 |
| ionosphere | -0.061 | 12 | -6.047 | 0 | 0 | 1 | -3.232 | -3.026 | -2.586 | 0.868 |

Table 6. Statistical tests between the proposed-penalized and Tao [?] technique for the "Beta" confidence measurement strategy.

| Independent t-test | Diff | Degrees of freedom | t | 2-sided p-value | Diff $<$ 0 p-value | Diff $>$ 0 p-value | Cohen d | Hedge's g | Glass's delta | Pearson r |
|---|---|---|---|---|---|---|---|---|---|---|
| kr-vs-kp | -0.04 | 12 | -5.702 | 0 | 0 | 1 | -3.048 | -2.854 | -2.921 | 0.855 |
| mushroom | -0.009 | 12 | -0.793 | 0.443 | 0.222 | 0.778 | -0.424 | -0.397 | -0.477 | 0.223 |
| iris | -0.002 | 12 | -0.091 | 0.929 | 0.465 | 0.535 | -0.048 | -0.045 | -0.048 | 0.026 |
| spambase | -0.03 | 12 | -3.547 | 0.004 | 0.002 | 0.998 | -1.896 | -1.775 | -1.421 | 0.715 |
| tic-tac-toe | -0.033 | 12 | -4.326 | 0.001 | 0 | 1 | -2.312 | -2.165 | -1.781 | 0.781 |
| sick | 0.001 | 12 | 0.14 | 0.891 | 0.554 | 0.446 | 0.074 | 0.07 | 0.053 | 0.04 |
| waveform | -0.023 | 12 | -2.672 | 0.02 | 0.01 | 0.99 | -1.428 | -1.337 | -1.442 | 0.611 |
| car | -0.008 | 12 | -0.871 | 0.401 | 0.2 | 0.8 | -0.465 | -0.436 | -0.332 | 0.244 |
| vote | -0.034 | 12 | -5.198 | 0 | 0 | 1 | -2.779 | -2.601 | -2.081 | 0.832 |
| ionosphere | -0.052 | 12 | -3.875 | 0.002 | 0.001 | 0.999 | -2.071 | -1.939 | -1.742 | 0.746 |

data measured between the proposed penalized technique and Tao's method. As can be seen from these results, the proposed technique has a significant improvement over the seven datasets, while delivering similar results for the other three datasets (p-value $< 0.05$).

Figure **??** shows the average accuracy distribution ("Beta" strategy) of the proposed penalized versus Tao and Sheng strategies for different numbers of annotators using kernel density estimation. This demonstrates that the proposed technique outperforms both Tao and Sheng on seven out of ten datasets, with higher average accuracy and less fluctuation over different annotator counts. Furthermore, Table **??** shows the statistical data measured between the proposed penalized technique and Tao, showing a significant improvement in six datasets, while performing similarly in the remaining datasets (p-value $< 0.05$).

Figure **??** shows the average accuracy for the "ionosphere" dataset for various

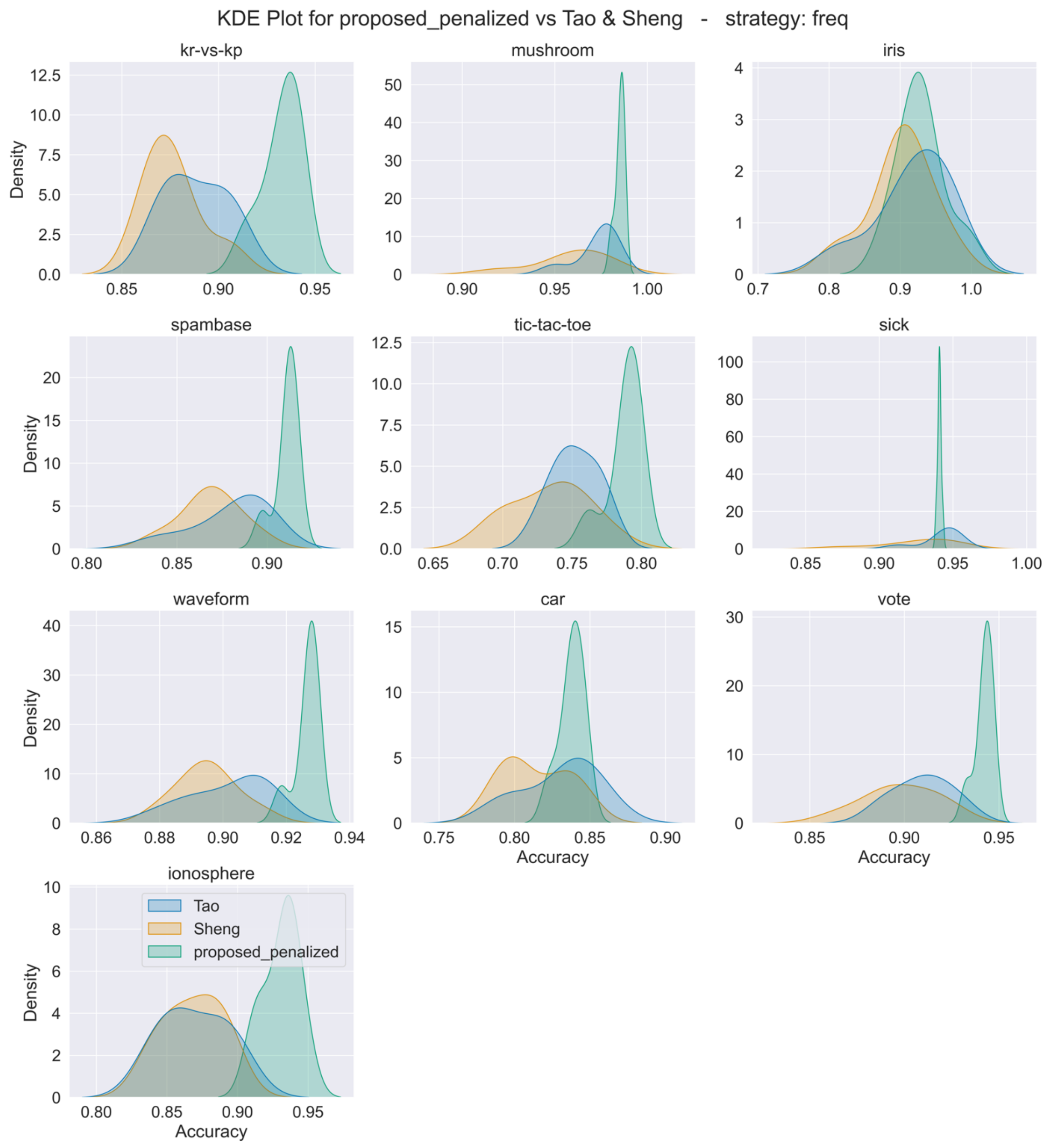

FIGURE 11. Measured accuracy distribution of the proposed-penalized aggregation technique uwMV-Freq, compared to wMV-Freq (Tao [?]), and MV-Freq (Sheng [?]) for different numbers of annotators, using the kernel density estimation technique

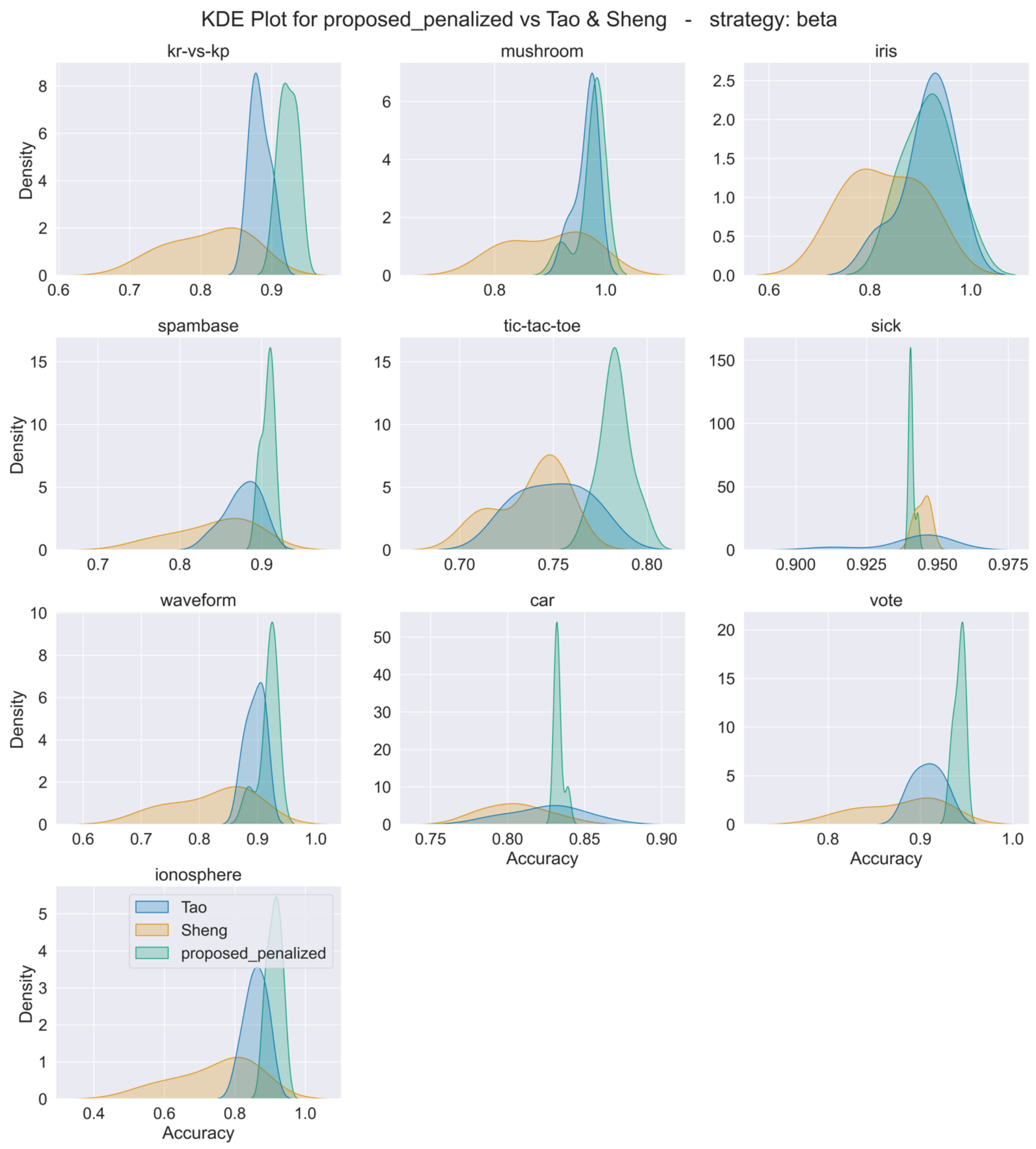

FIGURE 12. Measured accuracy distribution of the proposed-penalized aggregation technique uwMV-Beta, compared to wMV-Beta (Tao [?]), and MV- Beta (Sheng [?]) for different numbers of annotators, using the kernel density estimation technique.

annotator counts (horizontal axis). As can be seen, the proposed strategies considerably improve accuracy while utilizing a small number of annotators. Furthermore, Figure **??** shows the average accuracy of the three annotators (the smallest number of annotators that could perform consensus voting) on all ten datasets. Similarly, for the ionosphere dataset, we observed a similar trend in achieving higher accuracy on nine of the ten datasets compared to all benchmarks. It is important to note that the "freq" confidence measurement strategy is used to report the proposed techniques, as well as the Tao and Sheng results. Furthermore, the measured p-value calculated for the measured average accuracy (using three annotators) over different datasets showed a significant improvement for the proposed technique with and without penalization over all remaining benchmarks (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, Dawid Skene).

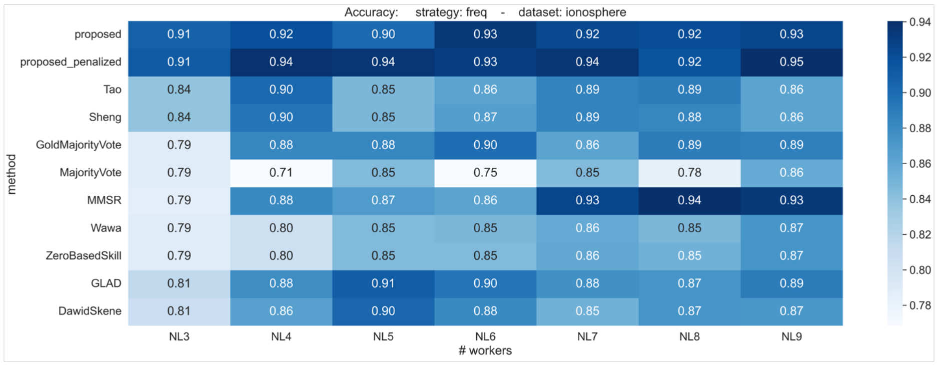

Figure 13. Average accuracy for the proposed aggregation techniques compared to the benchmarks for different numbers of annotators (horizontal axis) in the ionosphere dataset.

## 0.12 Discussion

Label aggregation is a critical component of crowdsourcing and ensemble learning strategies. Many generic label aggregation algorithms fall short because they do not

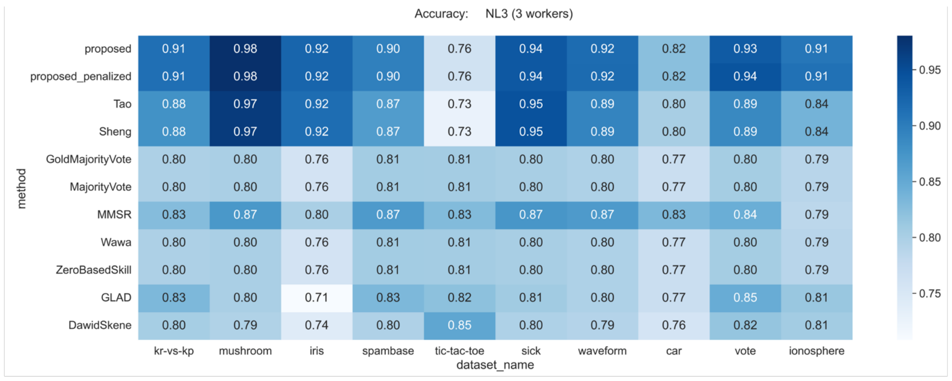

FIGURE 14. Average accuracy of the proposed aggregation techniques compared to the benchmarks for different datasets using three annotators.

account for the varying reliability of the annotators. In response to this, we have developed a novel label aggregation method that measures annotator reliability based on their consistency and accuracy, in relation to other annotators. We utilized uncertainty estimates to assign each annotator a more accurate weight, which correlates with their agreement with others and their consistency during labeling. In the second approach, we improved our initial strategy by penalizing annotator reliability estimates based on their inconsistencies in labeling. The first part of the proposed algorithm (calculating weights based on consistency) is essential because non-expert annotators often exhibit more irregular consistency during labeling than experts, as they are not trained to identify specific features. This measure helps to differentiate skilled and unskilled annotators. The goal of the second part of the algorithm (penalty for voting against the majority) is to prevent the algorithm from assigning disproportionately high weights to annotators who are consistently incorrect. For example, if annotators consistently mislabel a specific bird species, the second condition penalizes them for their error, despite their consistency. Furthermore, our method reports a single weight for the entire dataset instead of individual weights for each instance. This enables the reuse of calculated weights for future unlabeled test samples without needing to re-acquire labels or retrain classifiers each time new data need labeling. While we have not assessed our method in multi-label scenarios, the proposed techniques are anticipated to perform comparably on multi-label datasets, considering that all steps of the proposed approach involve per-class calculations. Experiments conducted on various crowdsourcing datasets demonstrate that our proposed methods outperform existing techniques in terms of accuracy and variance, especially when there are few annotators available.

## 0.13 Availability of data and materials

The code can be found in crowd-certain

## 0.14 Appendices

### List of abbreviations

### Competing interests

### Acknowledgements