DATA AND DOMAIN UNCERTAINTY IN MACHINE LEARNING

by

Artin Majdi

A Dissertation Submitted to the Faculty of the
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of
DOCTOR OF PHILOSOPHY

In the Graduate College
UNIVERSITY OF ARIZONA

2023

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by Artin Majdi, titled *DATA AND DOMAIN UNCERTAINTY IN MACHINE LEARNING*, and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

Jeffrey J. Rodriguez	Date
Carlos Alsua	Date
Ali Bilgin	Date
Greg Ditzler	Date

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Jeffrey J. Rodriguez Dissertation Director	Date

STATEMENT by AUTHOR

This dissertation *DATA AND DOMAIN UNCERTAINTY IN MACHINE LEARNING* prepared by Artin Majdi has been submitted in partial fulfillment of requirements for a doctoral degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under the rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of the source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by

SIGNED: _____

ACKNOWLEDGMENTS

As I reflect upon the completion of this invigorating journey, my heart brims with profound gratitude towards those who journeyed with me, imparting their wisdom, support, and guidance at every step.

First and foremost, my deepest appreciation is directed towards my mentor, Dr. Jeffrey J. Rodriguez, whose steadfast guidance and support throughout the project has been unparalleled. His expertise, dedication, and continual encouragement have been priceless to me, for which I am eternally thankful.

My sincere gratitude goes out to Mr. Nirav Merchant, Ms. Maliaca Oxnam, and Drs. Carlos Alsua, Manoj Saranathan, and Mahesh Keerthivasan, whose unceasing support has been invaluable throughout this endeavor. Additionally, I wish to convey my gratitude to Drs. Ali Bilgin and Abhijit Mahalanobis for their insightful contributions and feedback which significantly enhanced the depth and quality of my work. Your invaluable support and mentorship have played a crucial role in my personal and academic evolution.

An acknowledgment wouldn't be complete without a heartfelt tribute to my family. Your unwavering faith in my capabilities, your patience during the tough times, and your consistent encouragement have served as my strongest pillars of support.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Brief Background on Machine Learning and Uncertainty Management

The world is currently experiencing an era of rapid development in the field of machine learning. This exciting progression has triggered massive transformations across numerous sectors, including healthcare, automotive industries, and crowdsourcing platforms. In healthcare, machine learning algorithms help in diagnosing diseases, predicting patient outcomes, and personalizing treatment plans. In the automotive industry, the emergence of self-driving vehicles is revolutionizing transportation and mobility. In crowdsourcing, machine learning is used to aggregate information sourced from a large number of individuals to solve complex problems or provide insights. However, as we leverage these technologies to drive innovation and efficiency, we are also confronted with a critical challenge: *management of uncertainty and domain fluctuations in machine learning models*. Real-world applications of machine learning are riddled with situations where the provided data can contain missing, incorrect, or uncertain values. This often results in the unpredictable behavior of the machine learning models, leading to fluctuating performance across different domains. The precision of model predictions is crucial in high-stakes decision-making situations, such as medical diagnosis or autonomous driving. Given the inherent variability of input data and the susceptibility of models to error, accuracy alone is no longer sufficient. It is crucial to also account for uncertainty in the model's predictions. With sufficient knowledge of model and data uncertainty, we can make informed decisions regarding whether to trust the model's predictions or seek additional information before making a final decision. This capability is particularly important in critical situations where inaccurate predictions could have severe repercussions.

1.1.1 Uncertainty in Machine Learning Models

Uncertainty is an integral part of any predictive model, and machine learning models are no exception. Recognizing, quantifying, and managing this uncertainty is a critical aspect of developing robust and reliable machine learning systems. Uncertainties in the field of machine learning are typically divided into two categories: aleatoric and epistemic. Aleatoric uncertainty, also referred to as statistical uncertainty, is linked to the data's inherent noise or variability. This type of uncertainty is often irreducible as it stems from factors such as measurement errors or inherent randomness in the system being modeled. Epistemic uncertainty, on the other hand, is related to our lack of knowledge about the system. This form of uncertainty, also known as model uncertainty, arises from our inability to perfectly specify the parameters of our model or from using a model that does not perfectly capture the true underlying process. The sources of uncertainty in machine learning models are complex and can be attributed to various factors. Measurement errors, missing data, and inherent noise are examples of data-related sources of uncertainty. In contrast, model-related sources of uncertainty include the choice of model structure, or model parameters, and the use of approximations in model computations. Uncertainty can significantly impact the performance and behavior of machine learning models. Unmanaged uncertainty can lead to reduced model performance as the model struggles to make accurate predictions in the face of noisy data or imperfect model specifications. Uncertainty can also lead to biased predictions, as models may overfit to noisy data or become too confident in their predictions, ignoring the inherent uncertainty in the process. Overconfidence is a particularly dangerous effect of unmanaged uncertainty. When a model is overconfident, it may produce very certain predictions even when they are inaccurate. This can lead to poor decision-making, as users of the model may place too much trust in its predictions. In high-stakes domains such as healthcare or autonomous driving, overconfidence can lead to severe negative outcomes. Therefore, understanding and managing uncertainty is a critical task in machine learning. By providing uncertainty estimates along with predictions, machine learning models can become more reliable and trustworthy, facilitating their use in real-world decision-making tasks. Recent methods tackle the problem of missing and uncertain data, but they assume the data is drawn from well-known distributions defined by a few parameters. Some studies approach the issue of lack of accurate labels by pre-training a network on a large dataset with noisy labels and fine-tuning it for a smaller target dataset [**?**]. Semi-supervised techniques have also been proposed where the data with noisy labels is discarded [**?**]. However, they suffer from model complexity and cannot be applied to large-scale datasets. Even loss functions that are perceived as noise-robust are not completely robust to label noise [**?**]. With the rise and influence of ML in medical applications and the need to translate newly developed techniques into clinical practice, questions about the safety and uncertainty of models have gained more importance. Prior research

mostly focused on assessing the correctness of individual decisions and modeling the behavior of individual labelers (human experts or non-experts who assign labels to data) [**?**]. Label bias becomes more crucial when we move from using manual delineation as our gold standard to using existing software (e.g., Free Surfer [**?**] for subcortical segmentation tasks).

1.2 Objectives and Scope

The primary objective of this dissertation is to advance the understanding of uncertainty management and, to a lesser extent, domain fluctuation in machine learning models, with the ultimate aim of enhancing models' performance, reliability, and applicability in a variety of real-world settings. We have developed and evaluated novel methods for estimating and mitigating the impact of uncertainty on machine learning outcomes. By doing so, we hope to facilitate the more widespread adoption of these technologies across various sectors and real-world applications and, in the process, harness their full potential.

A significant part of this objective is devoted to investigating ways to incorporate uncertainty information into existing techniques with the goal of improving the accuracy of the prediction models as well as providing an additional confidence score that can enhance the models' interpretability and applicability in real-life settings. The challenge facing the machine learning community is to establish a comprehensive understanding of these issues, laying the foundation for the design of new techniques and strategies that manage uncertainty effectively in both supervised and unsupervised settings.

Addressing this challenge, we present several approaches to estimate and mitigate the effect of uncertainty in model outcomes. We propose strategies to manage uncertainty, aiming to make machine learning models more robust, reliable, and trustworthy. We focus primarily on healthcare, autonomous driving, and crowdsourcing, as these are areas where either the stakes are high or their applicability is widespread, and thus accurate uncertainty estimates can significantly enhance decision-making and outcomes.

In healthcare, we aim to improve the performance of machine learning models used for tasks such as disease diagnosis and organ segmentation. In the realm of autonomous driving, we introduce a novel transfer learning approach for driver distraction detection. In the field of crowdsourcing and ensemble learning, we have proposed new label aggregation techniques that take into account the consistency and accuracy of annotators (models in the case of ensemble learning). Further, we investigate the impact of domain fluctuation on model accuracy when segmenting thalamic nuclei and provide a novel

technique that facilitates the utilization of low-contrast imaging sequences by developing a model that, albeit trained on advanced imaging technologies, can readily be used on low-contrast yet more widely available imaging sequences without sacrificing too much accuracy. By keeping a broad scope, we have developed various methods that are not only effective in handling uncertainty and domain fluctuations but are also versatile and adaptable across different domains and applications, and thus have enhanced the utility of these techniques across different domains.

1.3 Dissertation's Organization

This dissertation contributes to the ongoing research in uncertainty management and domain fluctuation in machine learning models. The methods proposed in this dissertation take into account the inherent uncertainty and variability in the data, leading to improved model performance. These methods provide a framework that can be extended to other machine learning applications. Chapter 2 presents a novel method, "crowd-certain", that provides a more accurate and reliable label aggregation technique, leading to improved overall performance in both crowdsourcing and ensemble learning scenarios. This method takes into account the consistency and accuracy of the annotators as a measure of their reliability, which allows us to obtain a weight that closely follows the annotator's degree of reliability. Chapter 3 proposes a novel hierarchical multilabel classification technique that utilizes the taxonomic relationship between different classes to improve classification accuracy. Further, to reduce the effect of domain fluctuation and improve the generalizability of the model to images obtained from other sources, the proposed technique provides one model trained on multiple large publicly available chest X-ray datasets (CheXpert [**?**], NIH [**?**], and PADCHEST [**?**]). Chapter 4 presents a fast and accurate convolutional neural network (CNN) for segmentation of thalamic nuclei that is optimized for various diseases, magnetic field strengths, and image modalities. It demonstrates the potential of the proposed method for improving our understanding of the thalamic nuclei's involvement in neurological diseases. Finally, Chapters 5 and 6 provide a transfer learning approach for the detection of driver distraction and primary cilia cells, respectively. The proposed technique uses a combination of a CNN and a random decision forest to improve classification accuracy.

Chapter 2

Crowd-Certain: Towards Robust Label Aggregation in Crowdsourced and Ensemble Learning Classification

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 introduce Crowd-Certain, a novel approach for label aggregation in crowdsourced and ensemble learning classification tasks that offers superior performance, robustness, and computational efficiency. The proposed method uses the consistency of the annotators versus a trained classifier to determine a reliability score for each annotator. Furthermore, Crowd-Certain leverages predicted probabilities, enabling the reuse of trained classifiers on future sample data, thereby eliminating the need for recurrent simulation processes inherent in existing methods. We extensively evaluated our approach against ten existing techniques across ten different datasets, each labeled by varying numbers of annotators. The findings demonstrate that Crowd-Certain consistently outperforms the existing methods (Tao, Sheng, KOS, MACE, MajorityVote, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene), delivering higher average accuracy, F1 score and AUC rates across all tested scenarios, irrespective of the number of annotators involved. Additionally, we explored the performance of different confidence score measurement techniques, Freq and Beta, using two evaluation metrics: Expected Calibration Error (ECE) and Brier Score Loss. Our results show that Crowd-Certain consistently achieves higher Brier Score, indicating better-calibrated predictions, and superior performance in terms of ECE across most datasets, suggesting a higher accuracy of probabilistic predictions.

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

2.1 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. We take this concept a step further by calculating a weight for each annotator based not only on their reliability but also on their agreements with other annotators involved. This consideration of inter-reliability ensures a more comprehensive and dynamic weighting process, adjusting to the overall performance of the entire group of annotators. Thus, the generated weights become a robust measure of both individual and collective trustworthiness, which significantly improves the accuracy and efficacy of our labeling aggregation method.

We introduce a novel approach, termed as Crowd-Certain, that offers a significant improvement in label aggregation for crowdsourced and ensemble learning classification tasks, yielding improved performance across various scenarios. This technique leverages the consistency of annotators versus a trained classifier to ascertain their reliability, resulting in more accurate and efficient label aggregation. Our extensive experimental evaluation, conducted across ten diverse datasets, demonstrates that Crowd-Certain outperforms established techniques (Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene) in terms of aggregated label accuracy compared to ground truth labels. Importantly, Crowd-Certain consistently generates weights that closely follow pre-set ground truth accuracy for each annotator (termed as probability threshold in this study). Moreover, During inference time, Crowd-Certain employs predicted probabilities (obtained from a classifier trained on the worker's label set) rather than worker's labels, which facilitates the reuse of trained classifiers for future data samples, resulting in computational efficiency that surpasses previous methods.

The remainder of this paper is organized as follows. Section **??** examines related work involving label aggregation algorithms. In Section **??**, we provide an in-depth explanation of Crowd-Certain. Section **??**

presents the experiments and findings, and Section **??** encapsulates the results, highlighting key insights. Lastly, Section **??** concludes the paper and highlights the potential directions for future research.

2.2 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 $s_{\alpha}^{(i,k)}$ for the annotator α and instance i and the overall quality τ_{α} 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 α for that instance. To calculate the overall quality τ_{α}, 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 α as true labels and then assigns the average accuracy of the classifiers across all remaining instances as τ_{α}. The final weight for annotator α 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 α to estimate their respective τ_{α} 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.

2.3 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 even if they label instances incorrectly), we extend the proposed technique by penalizing the annotators for instances in which they disagree with the aggregated label obtained using 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 will be demonstrated in Section **??**, the proposed Crowd-Certain method is not only comparable to other techniques in terms of accuracy of the aggregated labels with respect to the ground truth labels 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.

2.3.1 Glossary of Symbols

For convenience, the following list summarizes the major symbols used in the subsequent discussion:

N: Number of instances.

M: Number of annotators.

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

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

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

$\pi_\alpha^{(k)}$: Probability threshold used as pre-set ground truth accuracy, for each annotator α and class k. It is used to generate sample binary labels (fictitious ground truth label set) for annotator α for class k. For example, the threshold values 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.

$Y^{(i)} = \left\{y^{(i,1)}, y^{(i,2)}, \ldots, y^{(i,K)}\right\}$: 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 $Y^{(i)} = \{1, 1, 0\}$.

$Z_\alpha^{(i)} = \left\{z_\alpha^{(i,1)}, z_\alpha^{(i,2)}, \ldots, z_\alpha^{(i,K)}\right\}$: Label set given by the annotator α 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 is 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 probability threshold $\pi_\alpha^{(k)}$ is greater than $\rho^{(i)}$, the true label $y^{(i,k)}$ is assigned; otherwise, an incorrect label $1 - y^{(i,k)}$ is assigned.

$\Pi_\alpha = \left\{\pi_\alpha^{(1)}, \pi_\alpha^{(2)}, \ldots, \pi_\alpha^{(K)}\right\}$: set of K probability thresholds for annotator α.

$\mathbb{X} = \left\{X^{(i)}\right\}_{i=1}^{N}$: 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 α.

$\mathbb{P} = \left\{\rho^{(i)}\right\}_{i=1}^{N}$: 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 α.

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

$P_\alpha^{(i),(g)} = \left\{p_\alpha^{(i,k),(g)}\right\}_{k=1}^{K}$: 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.

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

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

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

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

$\omega_{\alpha}^{(k)}$: Estimated weight for annotator α and class k.

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

2.3.2 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:

$$E = \sum_{i=1}^{N}\sum_{a=1}^{M}\left(\sum_{k=1}^{K}\delta\left(y^{(i,k)}, z_{\alpha}^{(i,k)}\right)\right) \tag{2.3.1}$$

where δ 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 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 (of the aggregated labels with respect to the ground truth labels), 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 defined as follows:

$$R(h) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}\left(Y^{(i)}, h\left(\left\{Z_{\alpha}^{(i)}\right\}_{\alpha=1}^{M}\right)\right) \tag{2.3.2}$$

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 [**?**]:

$$\widehat{h} = \underset{h}{\operatorname{argmin}} R(h) \tag{2.3.3}$$

2.3.3 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 $M \times K$ probability thresholds Π, where K is the number of classes. (Note that an annotator may be skilled at labeling dogs, but not rabbits.) Then we use these probability thresholds to generate the crowd label set $Z_\alpha^{(i)}$ from the ground truth labels for each instance i. For each annotator α, 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 α, 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 α; otherwise, the incorrect label is used. The calculated annotator labels $z_\alpha^{(i,k)}$ for each annotator α, instance i and class k are as follows:

$$z_\alpha^{(i,k)} = \begin{cases} y^{(i,k)} & \text{if} \rho^{(i)} \leq \pi_\alpha^{(k)}, \\ 1 - y^{(i,k)} & \text{if} \rho^{(i)} > \pi_\alpha^{(k)}, \end{cases} \quad \forall i, a, k \tag{2.3.4}$$

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

2.3.4 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 comprises 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 α. 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 $\left\{p_\alpha^{(i,k),(g)}\right\}_{k=1}^{K}$ representing the probability

that class k is present in the sample. Consequently, we obtain a collection of G predicted probability sets $\left\{\left\{p_\alpha^{(i,k),(g)}\right\}_{k=1}^{K}\right\}_{g=1}^{G}$ for each annotator α and instance i. The set $\left\{p_\alpha^{(i,k),(g)}\right\}_{g=1}^{G}$ contains the predicted probabilities for class k, annotator α, and instance i. Disagreements between predicted probabilities $\left\{p_\alpha^{(i,k),(g)}\right\}_{g=1}^{G}$ 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 gth classifier for all annotators. Define $t_\alpha^{(i,k),(g)}$ as the predicted label obtained by binarizing the predicted probabilities $p_\alpha^{(i,k),(g)}$ 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 $u_\alpha^{(i,k)}$ 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 indicates 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:

$$\Delta_\alpha^{(i,k)} = H\left(\left\{p_\alpha^{(i,k),(g)}\right\}_{g=1}^{G}\right) = -\sum_g p_\alpha^{(i,k),(g)} \log\left(p_\alpha^{(i,k),(g)}\right) \tag{2.3.5}$$

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 $\{t_\alpha^{(i,k),(g)}\}_{g=1}^{G}$ with mean value μ, the standard deviation is

defined as.

$$\Delta_{\alpha}^{(i,k)} = \mathrm{SD}\left(\left\{t_{\alpha}^{(i,k),(g)}\right\}_{g=1}^{G}\right) = \sqrt{\frac{1}{G-1}\sum_{g=1}^{G}\left(t_{\alpha}^{(i,k),(g)} - \mu\right)^2}, \quad \mu = \frac{1}{G}\sum_{g=1}^{G} t_{\alpha}^{(i,k),(g)} \tag{2.3.6}$$

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 γ (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:

$$\begin{aligned} P\left(Q_L^k \leq p_{\alpha}^{(i,k),(g)} \leq Q_U^k\right) &= \gamma \\ \Delta_{\alpha}^{(i,k)} &= Q_L^k - Q_U^k \end{aligned} \tag{2.3.7}$$

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 $P_{\alpha}^{(i,k)} = \mathrm{sorted}\left(\left\{p_{\alpha}^{(i,k),(g)}\right\}_{g=1}^{G}\right), \quad \forall \alpha, k, i.$

2. Calculate the lower and upper quantile indices based on the chosen confidence level γ. The lower quantile index is $L = \mathrm{ceil}\left(\frac{G}{2}(1-\gamma)\right)$, and the upper quantile index is $U = \mathrm{floor}\left(\frac{G}{2}(1+\gamma)\right)$, 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 $P\left(Q_L^k <= p_{\alpha}^{(i,k),(g)} <= Q_U^k\right) = \gamma$, where Q_L^k and Q_U^k represent the bounds of the interval containing the α 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

The 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.

$$\Delta_{\alpha}^{(i,k)} = \mathrm{VarCommittee}\left(P_{\alpha}^{(i,k)}\right) = \frac{1}{G-1}\sum_{g=1}^{G}\left(p_{\alpha}^{(i,k),(g)} - \mu\right)^2, \quad \mu = \frac{1}{G}\sum_{g=1}^{G} p_{\alpha}^{(i,k),(g)} \tag{2.3.8}$$

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, $\mathrm{score\_function}$ measures the conformity of the prediction with the true label. In the context of this study, the true label can be replaced by $\eta_{\alpha}^{(i,k)}$. A common choice for $\mathrm{score\_function}$ 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 $\zeta_k^g = \mathrm{score\_function}\left(p_{\alpha}^{(i,k),(g)}, y^{(i,k)}\right)$

2. Calculate the p-value for each class k as the proportion of classifiers with nonconformity scores greater than or equal to a predefined threshold $\mathsf{T}^{(k)}$: $\text{p-values}(k) = \frac{\left|\{g:\, \zeta^{(k),(g)} \geq \mathsf{T}^{(k)}\}\right|}{G}$

3. 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.

2.3.5 Crowd-Certain: Uncertainty-Based Weighted Soft Majority Voting

Consistency Measurement

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

calculate $c_\alpha^{(i,k)}$ from $\Delta_\alpha^{(i,k)}$.

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

$$c_\alpha^{(i,k)} = 1 - \Delta_\alpha^{(i,k)} \ , \ \forall i, \alpha, k \tag{2.3.9}$$

2. In a second approach (shown in Equation (**??**)), we penalize annotators for instances in which their predicted label $\eta_\alpha^{(i,k)}$ (explained in the Glossary of Symbols section) does not match the MV of all annotator labels $\underset{\alpha}{\mathrm{MV}}\left(z_{\alpha,\,k}^{(i,k)}\right)$. As previously discussed, instead of directly working with the annotator's labels $z_\alpha^{(i,k)}$, we use the predicted labels obtained from the ensemble of classifiers $\eta_\alpha^{(i,k)}$. 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.

$$c_\alpha^{(i,k)} = \begin{cases} 1 - \Delta_\alpha^{(i,k)} & \text{if } \eta_\alpha^{(i,k)} = \mathrm{MV}_\alpha(\eta_\alpha^{(i,k)}) \\ 0 & \text{otherwise} \end{cases} \tag{2.3.10}$$

Reliability Measurement

For each annotator, for each class, and for each instance, there is a consistency score, $c_\alpha^{(i,k)}$. By averaging these scores across all instances, we can define a reliability score for each annotator and for each class:

$$\psi_\alpha^{(k)} = \frac{1}{N} \sum_{i=1}^{N} c_\alpha^{(i,k)} \tag{2.3.11}$$

If desired, one may also calculate an overall reliability score for each annotator by averaging across all classes:

$$\psi_\alpha = \frac{1}{K} \sum_{k=1}^{K} \psi_\alpha^{(k)} \tag{2.3.12}$$

Weight Measurement

Furthermore, we calculate the annotators' weights $\omega_\alpha^{(k)}$ for each class k by normalizing the reliability values as follows:

$$\omega_\alpha^{(k)} = \frac{\psi_\alpha^{(k)}}{\sum_{\alpha=1}^{M} \psi_\alpha^{(k)}} \tag{2.3.13}$$

Aggregated Label Calculation

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

$$\nu^{(i,k)} = \begin{cases} 1 & \text{if } \left(\sum_{\alpha=1}^{M} \omega_\alpha^{(k)} \eta_\alpha^{(i,k)}\right) > 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, k \tag{2.3.14}$$

Confidence Score Calculation

In previous section we showed how to calculate the aggregated label $\nu^{(i,k)}$ (shown in Equation (**??**)). Define $F^{(i,k)}$ as the confidence score for instance i and class k. We calculate two confidence scores $F^{(i,k)}$, based on how many different annotators agree on the reported label $\nu^{(i,k)}$. 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 α.

Using Weighted Sum: In a standard voting system, for every instance i and class k, each contributor in the group—whether that be an annotator in a crowd or a model in an ensemble learning context—provides a class label. The label receiving the most votes, meaning it's predicted by the majority of contributors, is selected as the final prediction. This approach is often referred to as majority voting or hard voting.

This method could be improved by taking into account not only the number of votes each label receives, but also the confidence associated with each vote. This factor introduces the notion of a "weighted sum of all votes for a particular class". As part of this study, we propose techniques that assign a weight to each contributor in the group. This calculation is based on their voting consistency and the degree to which they concur with their peers.

To compute the weighted sum of all votes for each class, we can combine the calculated weights with the corresponding labels, whether provided or predicted. This calculation gives greater confidence to votes with greater certainty. This refined approach prioritizes certainty, thereby enhancing the ensemble's overall effectiveness.

The confidence score $F_\Omega^{(i,k)}$ is formulated as follow.

$$F_\Omega^{(i,k)} = \sum_{\alpha=1}^{M} \omega_\alpha^{(k)} \delta\left(\eta_\alpha^{(i,k)} \ , \ \ \nu^{(i,k)}\right) \tag{2.3.15}$$

where δ is the Kronecker delta function.

Using CDF of Beta distribution function: The binomial distribution survival function, also referred to as the complementary cumulative distribution function (CCDF), provides the probability of observing a result as extreme or more extreme than a given value. It provides the probability that a random variable drawn from the binomial distribution is greater than or equal to a given value.

It can be applied in the following ways when calculating confidence scores:

1. **Hypothesis testing:** Suppose you are testing a hypothesis concerning a parameter of a population, and you collect a sample of observations. Given the null hypothesis, the binomial survival function can be used to calculate the probability of observing a result as extreme or more extreme than the one observed. This probability is the p-value, and if it is very small, the null hypothesis may be rejected. In this context, the confidence score could be interpreted as (1 − p-value), a measure of the certainty with which the null hypothesis can be rejected.

2. **Binary classification:** Consider a binary classification problem in which an algorithm sorts objects into two groups, A and B. Given the observed results, the binomial survival function can be used to calculate the probability of misclassification. The confidence score in this instance could be interpreted as (1 − probability of misclassification).

In the following equation, the probability of obtaining k successes or more out of n trials where the probability of success on any given trial is p is calculated. In this context, "success" could refer to the event in question, such as the correct classification of an object or the acceptance of a hypothesis. The CDF of the Beta distribution at the decision threshold of 0.5 is used to calculate a confidence score $F_{\beta}^{(i,k)}$. To calculate the two shape parameters of the Beta distributions $l^{(i,k)}$ and $u^{(i,k)}$, a weighted sum of all correct and incorrect aggregated labels, is used respectively:

$$\begin{aligned} l^{(i,k)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha}^{(k)} \ \delta\left(\eta_{\alpha}^{(i,k)}, \nu_{k}^{(i,k)}\right) \\ u^{(i,k)} &= 1 + \sum_{\alpha=1}^{M} \omega_{\alpha}^{(k)} \ \delta\left(\eta_{\alpha}^{(i,k)}, 1 - \nu_{k}^{(i,k)}\right) \end{aligned} \tag{2.3.16}$$

$$F_{\beta}^{(i,k)} = I_{0.5}\left(l^{(i,k)}, u^{(i,k)}\right) = \sum_{t=[l^{(i,k)}]}^{T-1} \frac{(T-1)!}{t!(T-1-t)!} 0.5^{T-1} \tag{2.3.17}$$

where $T = \left[l^{(i,k)} + u^{(i,k)}\right\rangle]$ and $[\cdot]$ is an integer function.

2.3.6 Metrics

- **Accuracy:** The accuracy of the model is the proportion of true results (both true positive and true negatives) among the total number of cases examined. Mathematically, accuracy can be represented as:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\delta\left(v^{(i,k)}, y^{(i,k)}\right) \tag{2.3.18}$$

  where δ is the Kronecker delta function, N is the total number of instances, and K is the number of classes, $y^{(i,k)}$ and $v^{(i,k)}$ are the ground truth and aggregated label respectively for class k and instance i. Although accuracy is most effective for balanced classes, its interpretation can be skewed in the presence of significant class imbalance.

- **F1 Score:** The F1 score is the harmonic mean of precision and recall and can be used for assessing the quality of aggregated labels, especially in the presence of imbalanced classes. F1 score provides a balanced measure of precision and recall, ranging from 0 to 1, where 1 represents the best possible F1 score. It's computed as:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.3.19}$$

  where Precision = $\frac{\text{TP}}{\text{TP+FP}}$ and Recall = $\frac{\text{TP}}{\text{TP+FN}}$, and TP, FP and FN are the number of true positives, false positives and false negatives, respectively.

- **Area Under the Receiver Operating Characteristic Curve (AUC-ROC):** AUC-ROC measures the trade-off between true positive rate (sensitivity) and false positive rate (1-specificity) for every possible cut-off. Higher AUC-ROC values indicate better classification performance.

- **Brier Score:** Brier score provides a measure of the accuracy of the probabilistic or confidence score predictions. It's calculated as the mean squared difference between the predicted probability and the actual outcome, thereby rewarding predictions that are both well-calibrated and confident. It can be calculated as follows:

$$\text{Brier Score} = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\left(v^{(i,k)} - y^{(i,k)}\right)^2 \tag{2.3.20}$$

- **Expected Calibration Error (ECE):** is used to quantify the calibration of confidence scores produced by a model. It's computed as a weighted average of the absolute differences between the actual

accuracies and the predicted confidences within each bin when predictions are grouped into distinct bins based on their predicted confidence. A lower ECE signifies a model whose predicted probabilities closely match the observed frequencies across all bins. ECE can be formulated as follows:

$$\text{ECE} = \sum_{b=1}^{B} \frac{|B_b|}{N} \left|\text{Accuracy}(B_b) - \text{Confidence-Score}(B_b)\right| \tag{2.3.21}$$

where B is the number of bins, B_b is the set of instances in bin b, N is the total number of instances, Accuracy(B_b) is the accuracy of bin b, and Confidence-Score(B_b) is the average confidence of bin b.

2.4 Results

To evaluate our proposed technique, we conducted a series of experiments comparing the proposed technique with several existing 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 [**?**].

2.4.1 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 [**?**].

- 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.

Table 2.1: 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 **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 attributes (X, O, and blank) corresponding to each of the nine tic-tac-toe squares.
- 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. Attributes include 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. We generated multiple fictitious label sets for each dataset to simulate the crowdsourcing concept of collecting several crowd labels for each instance. We selected random samples in the datasets using a uniform distribution and altered 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 various levels of accuracy, we can evaluate the performance of the proposed method under different conditions of annotator expertise and reliability. This allows 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.

2.4.2 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 [**?**].

- **Worker Agreement with Aggregate (WAWA) [?]:** Wawa, also referred to as "inter-rater agreement", is a metric used in crowdsourcing jobs that do not employ test questions [**?**]. The WAWA algorithm consists of three steps: it calculates the majority vote label, estimates workers' skills as a fraction, and calculates the agreement between workers and the majority vote [**?**].

- **Zero-Based-Skill (ZBS) [?]:** employs a weighted majority vote (WMV). After processing a collection

of instances, it re-evaluates 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.

- **Karger-Oh-Shah (KOS) [?]:** Iterative algorithm that calculates the log-likelihood of the task being positive while modeling the reliabilities of the workers. Let $A_{(i,\alpha)}$ be a matrix of answers of worker α on task i. $A_{(i,\alpha)} = 0$ if worker α didn't answer the task i otherwise $|A_{(i,\alpha)}| = 1$. The algorithm operates on real-valued task messages $x_{i\to\alpha}$ and worker messages $y^{\alpha\to i}$. A task message $x_{i\to\alpha}$ represents the log-likelihood of task i being a positive task, and a worker message $y_{\alpha\to i}$ represents how reliable worker α is. On iteration k the values are updated as follows [**?**]:

$$x_{i\to\alpha}^{(k)} = \sum_{\alpha'\in\partial i\setminus\alpha} A_{(i,\alpha')} y_{\alpha'\to i}^{(k-1)} \, y_{\alpha\to i}^{(k)} = \sum_{i'\in\partial\alpha\setminus i} A_{(i',\alpha)} x_{i'\to\alpha}^{(k-1)} \tag{2.4.1}$$

- **Multi-Annotator Competence Estimation (MACE) [?, ?]:** Probabilistic model that associates each worker with a probability distribution over the labels. For each task, a worker might be in a spamming or not spamming state. If the worker is not spamming, they yield a correct label. If the worker is spamming, they answer according to their probability distribution. Let's assume that the correct label $y^{(i)}$ comes from a discrete uniform distribution. When a worker annotates the task, they are in the spamming state with probability $\mathrm{Bernoulli}(1-\theta_\alpha)$. So, if their state $s_\alpha = 0$, their response $z_\alpha^{(i)} = y^{(i)}$. Otherwise, their response $z^{(i,\alpha)}$ is drawn from a multi-nomial distribution with parameters ξ_α.

- **Matrix Mean-Subsequence-Reduced Algorithm (MMSR) [?, ?]:** The MMSR assumes that workers have different level of expertise and associated with a vector of "skills" $\boldsymbol{s}$ which entries s_α show the probability of the worker α to answer correctly to the given task. Having that, we can show that.

$$\mathbb{E}\left[\frac{K}{K-1}\widetilde{C} - \frac{1}{K-1}\mathbf{1}\mathbf{1}^T\right] = \boldsymbol{s}\boldsymbol{s}^T, \tag{2.4.2}$$

  where K is the total number of classes, $\widetilde{C}$ is a covariation matrix between workers, and $\mathbf{1}\mathbf{1}^T$ is the all-ones matrix which has the same size as $\widetilde{C}$.

  So, the problem of recovering the skills vector $\boldsymbol{s}$ becomes equivalent to the rank-one matrix completion problem. The MMSR algorithm is an iterative algorithm for *rubust* rank-one matrix completion, so its result is an estimator of the vector $\boldsymbol{s}$. Then, the aggregation is the weighted majority vote with weights equal to $\log\frac{(K-1)s_\alpha}{1-s_\alpha}$.

- **Generative model of Labels, Abilities, and Difficulties (GLAD) [?, ?]:** A probabilistic model that parametrizes workers' abilities and tasks' difficulties. Let's consider a case of K class classification. Let p be a vector of prior class probabilities, $\omega_\alpha \in (-\infty, +\infty)$ be a worker's ability parameter, $\beta^{(k)} \in (0, +\infty)$ be an inverse task's difficulty, $y^{(k)}$ be a latent variable representing the true task's label, and $z_\alpha^{(k)}$ be a worker's response that we observe. The relationships between these variables and parameters according to GLAD are represented by the following latent label model. The prior probability of $y^{(k)}$ being equal to c is $\Pr(y^{(k)} = c) = p[c]$ the probability distribution of the worker's responses conditioned by the true label value c follows the single coin Dawid-Skene model where the true label probability is a sigmoid function of the product of worker's ability and inverse task's difficulty:

$$\Pr(z_\alpha^{(k)} = j | y^{(k)} = c) = \begin{cases} f(\alpha, k), & j = c \\ \frac{1 - f(\alpha,k)}{K-1}, & j \neq c \end{cases}, \tag{2.4.3}$$

  where $f(\alpha, k) = \frac{1}{1+e^{-\omega_\alpha \beta^{(k)}}}$.

  Parameters p, ω, β and latent variables y are optimized through the Expectation-Minimization algorithm.

- **Dawid-Skene [?, ?]:** Probabilistic model that parametrizes workers' level of expertise through confusion matrices. Let e^α be a worker's confusion (error) matrix of size $K \times K$ in case of K class classification, p be a vector of prior classes probabilities, $y^{(i)}$ be a true task's label, and $z_\alpha^{(i)}$ be a worker's answer for the task i. The relationships between these parameters are represented by the following latent label model. Here the prior true label probability is $\Pr(y^{(i)} = c) = p[c]$ and the distribution on the worker's responses given the true label c is represented by the corresponding column of the error matrix: $\Pr(z_\alpha^{(i)} = k | y^{(i)} = c) = e^\alpha[k, c]$ Parameters p and e^α and latent variables z are optimized through the Expectation-Maximization algorithm.

2.4.3 Weight Measurement Evaluation

Following the generation of multi-label sets, the aggregate labels were determined using the proposed Crowd-Certain as well as various established methods. We examined two strategies for classifier selection, as detailed in Section **??**. Give there was no substantial variations in the final outcomes observed, the second strategy was adopted for its utilization of the random forest classification technique. This choice not only conserved processing time but also decreased the need for numerous Python package dependencies. For each annotator α, we trained ten distinct random forests, each comprising

four trees with a maximum depth of four, under various random states, as outlined in Section **??**.

Figure **??** depicts the relationship between the randomly assigned annotators' probability threshold ($\pi_\alpha^{(k)}$) and their corresponding estimated weights ($\omega_\alpha^{(k)}$). In Tao's method scenario, the figure presents the average weights over all instances. Notably, as the reliability (probability threshold) of an annotator exceeds a particular threshold, the weight computed by Tao's method reaches a saturation point, while the proposed technique exhibits a considerably stronger correlation. The individual data points symbolize the actual calculated weights, and the curve illustrates the regression line.

2.4.4 Label Aggregation Evaluation

The Figure **??** portrays a thorough accuracy comparison of our novel label aggregation technique, termed Crowd-Certain, against ten existing methods, evaluated over ten distinct datasets. Each dataset was labeled by three different workers, with labels generated based on a uniform distribution and specific probability thresholds Π_α as explained in Section **??**.

For a comprehensive evaluation, all experiments were repeated three times using different random seed numbers to account for randomness. The accuracy scores presented in the figure represent the average of these three runs and illustrate the degree of concordance between the aggregated label $\nu^{(i,k)}$ from each technique and the actual ground truth $y^{(i,k)}$.

It is important to note that, in the execution of our proposed technique, Crowd-Certain, the aggregated labels were derived through the application of the predicted probabilities, denoted as $\eta_\alpha^{(i,k)}$. This approach is significant as it enables the reuse of trained classifiers on future sample data, eliminating the need for recurrent simulation processes - a substantial advantage in terms of computational efficiency. Conversely, the methodologies of existing techniques necessitated the use of actual crowd labels $z_\alpha^{(i,k)}$ to determine the aggregated labels. For example, in case of Tao [**?**] the aggregated labels were obtained using the equation **??**. These methods inherently involve re-running simulations for every new dataset, which could be computationally expensive and time-consuming.

$$\nu^{(i,k)} = \begin{cases} 1 & \text{if } \left(\sum_{\alpha=1}^{M} \omega_\alpha^{(k)}\, z_\alpha^{(i,k)}\right) > 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, k \tag{2.4.4}$$

Across all ten datasets, it is clear that the Crowd-Certain method consistently outperforms the existing methods, yielding higher average accuracy rates. For example, in the 'kr-vs-kp' dataset, our proposed Crowd-Certain method achieved an average accuracy of approximately 0.923, significantly exceeding

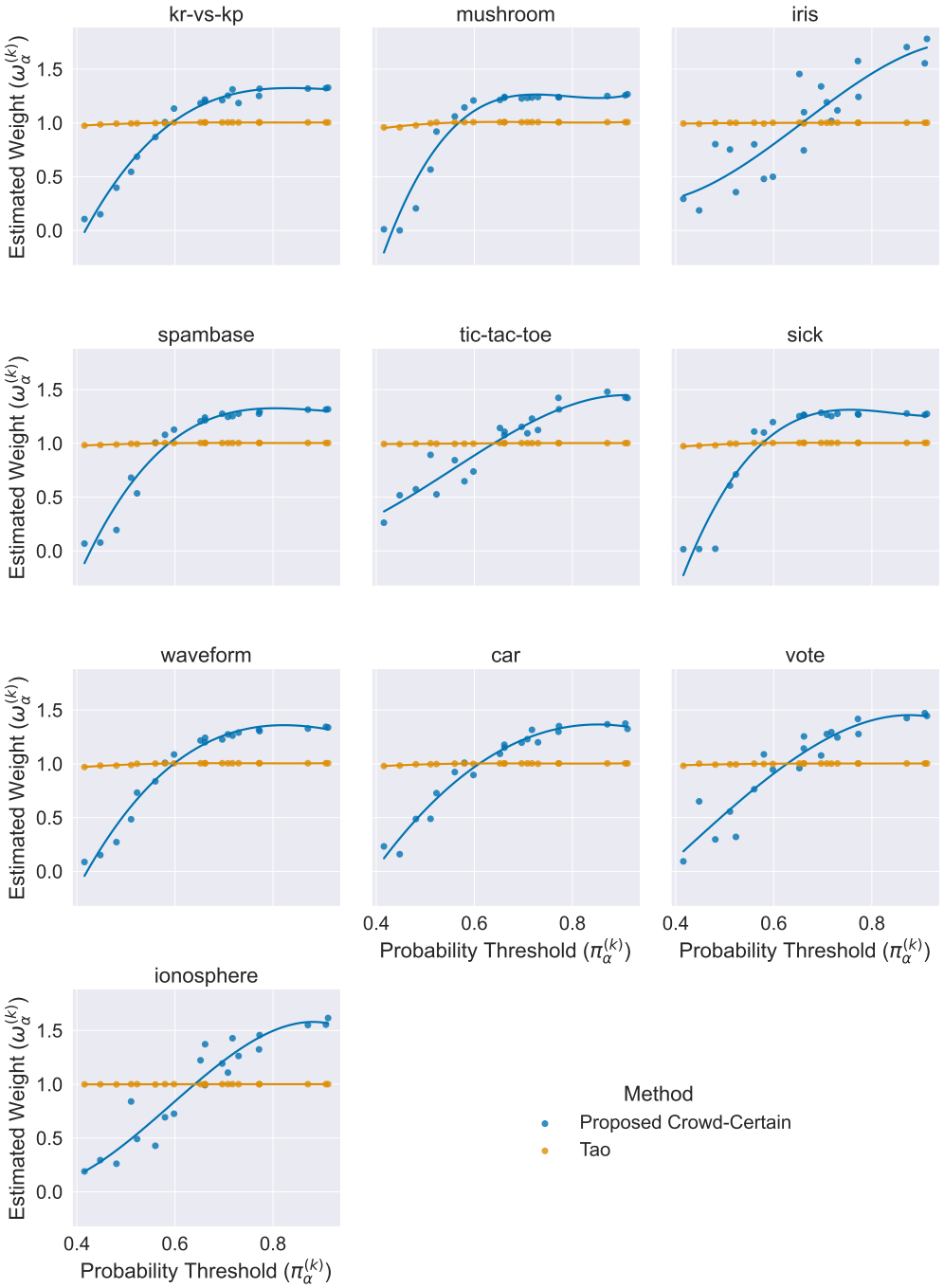

Figure 2.1: Comparison of weight computation techniques across ten different datasets. Each subplot corresponds to a unique dataset, illustrating the relationship between the randomly assigned annotator's probability threshold ($\pi_\alpha^{(k)}$) (horizontal axis) and the computed weights ($\omega_\alpha^{(k)}$) (vertical axis) for the proposed aggregation technique with penalization Crowd-Certain and Tao [**?**]. The individual data points represent actual measured weights, while the curve stands for the regression line.

the highest-performing existing method that reached an accuracy of about 0.784. This trend holds true across other datasets as well, such as 'mushroom', 'spambase', and 'tic-tac-toe', where the Crowd-Certain method achieves superior average accuracies of around 0.980, 0.900, and 0.741, respectively. In contrast, the competing methods struggled to exceed an average accuracy of 0.771 in these datasets.

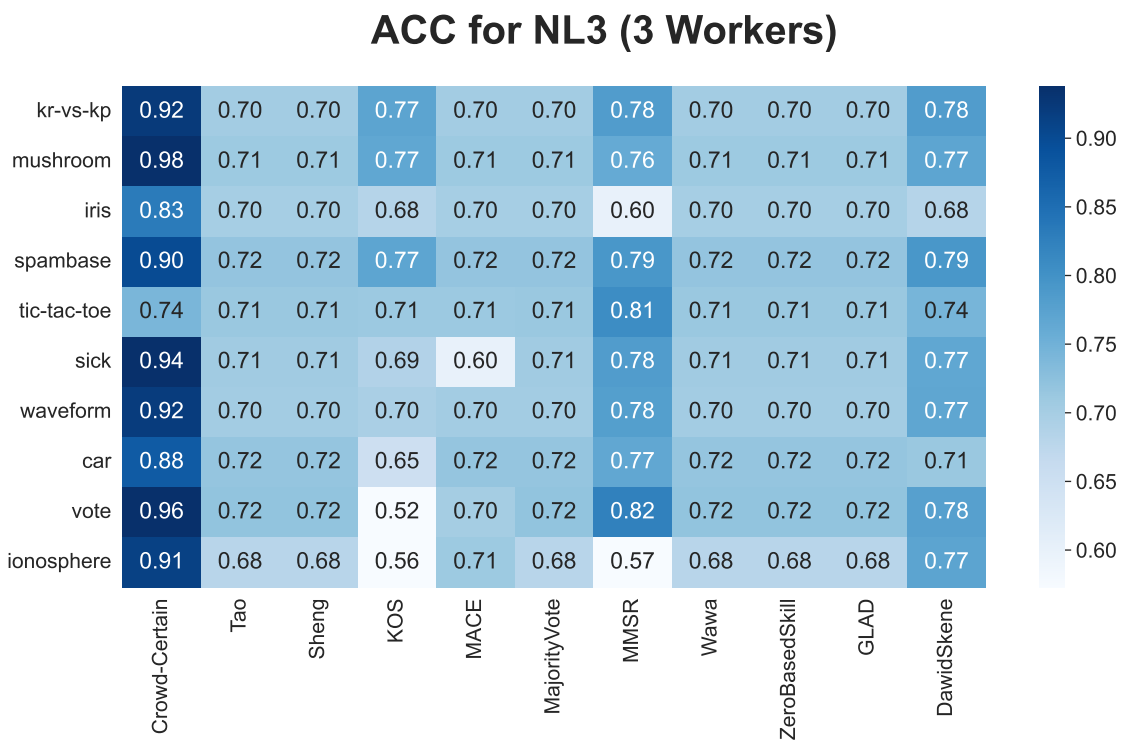

Figure 2.2: Comparison of Accuracy scores for multiple label aggregation techniques on various datasets. The figure displays the mean accuracy score obtained across three independent trials for the proposed method (Crowd-Certain) and ten existing label aggregation techniques. The trials were conducted using three annotators (workers) per dataset. The aggregated labels for Crowd-Certain were derived using predicted probabilities, allowing for reuse of trained classifiers. In contrast, existing techniques used actual crowd labels, necessitating repeated simulations.

We further extended our experiment to explore the effects of varying the number of annotators, ranging from 3 up to 7. The results shown in Figure **??**, are presented as a series of box plots, each illustrating the distribution of accuracy (1st column), F1 (2nd column), and AUC (3rd column) scores across the 10 datasets for a given number of annotators. These plots provide a clear visual summary of our technique's robust performance across various settings, including the median, quartiles, and potential outliers in the distribution of accuracies. Notably, our proposed Crowd-Certain technique consistently shows improvements over the 10 benchmark methods across all scenarios. This enhancement is evident

irrespective of the number of annotators involved, further highlighting the robustness and adaptability of the Crowd-Certain approach.

2.4.5 Confidence Score Evaluation

The table presents the evaluation of the two confidence score measurement techniques, namely Freq and Beta, using two performance metrics: Expected Calibration Error (ECE) and Brier Score Loss. The evaluations were conducted across a variety of datasets and three techniques: Crowd-Certain, Tao, and Sheng, when using 3 workers.

Figure **??** depicts the performance of three different strategies: Crowd-Certain, Tao, and Sheng, compared across two metrics - Expected Calibration Error (ECE) and Brier Score Loss. These results are obtained using two different confidence score calculation techniques Freq and Beta, applied over ten different datasets when having three annotators. The ECE score offers an aggregated measure of the reliability of probabilistic predictions. In this case, it is used to assess the calibration of the aggregated labels across different techniques and strategies. A lower ECE score indicates better-calibrated predictions, i.e., the predicted probabilities are closer to the true probabilities. Brier Score Loss is a metric that quantifies the accuracy of probabilistic predictions. It calculates the mean squared difference between the predicted probabilities and the actual outcome. Hence, lower Brier Score Loss values correspond to better model performance. In the Figure **??**, it can be observed that for the ECE metric, across all datasets, the proposed Crowd-Certain strategy consistently achieves lower scores when compared to Tao and Sheng, for both Freq and Beta techniques. This indicates that the Crowd-Certain strategy offers better-calibrated predictions, providing a higher level of confidence in the aggregated labels. For the Brier Score Loss metric, the Crowd-Certain strategy also appears to outperform Tao and Sheng across most datasets, for both techniques. The Figure **??** showcases the results for two metrics, Expected Calibration Error (ECE) and Brier Score Loss, for two confidence measurement techniques - Freq and Beta strategies, applied using three different techniques: Crowd-Certain, Tao, and Sheng. These results are obtained for the kr-vs-kp dataset under different numbers of annotators from 3 (denoted with NL3) up to (denoted with NL7).

In general, the ECE and Brier Score Loss both increase as the number of annotators increases, which suggests that increasing the number of annotators does not necessarily improve the performance. The performance varies depending on the confidence measurement technique and the strategy used.

For the Freq strategy, the Crowd-Certain technique consistently yields lower ECE and Brier Score across different numbers of annotators compared to the Tao and Sheng techniques, indicating better calibration

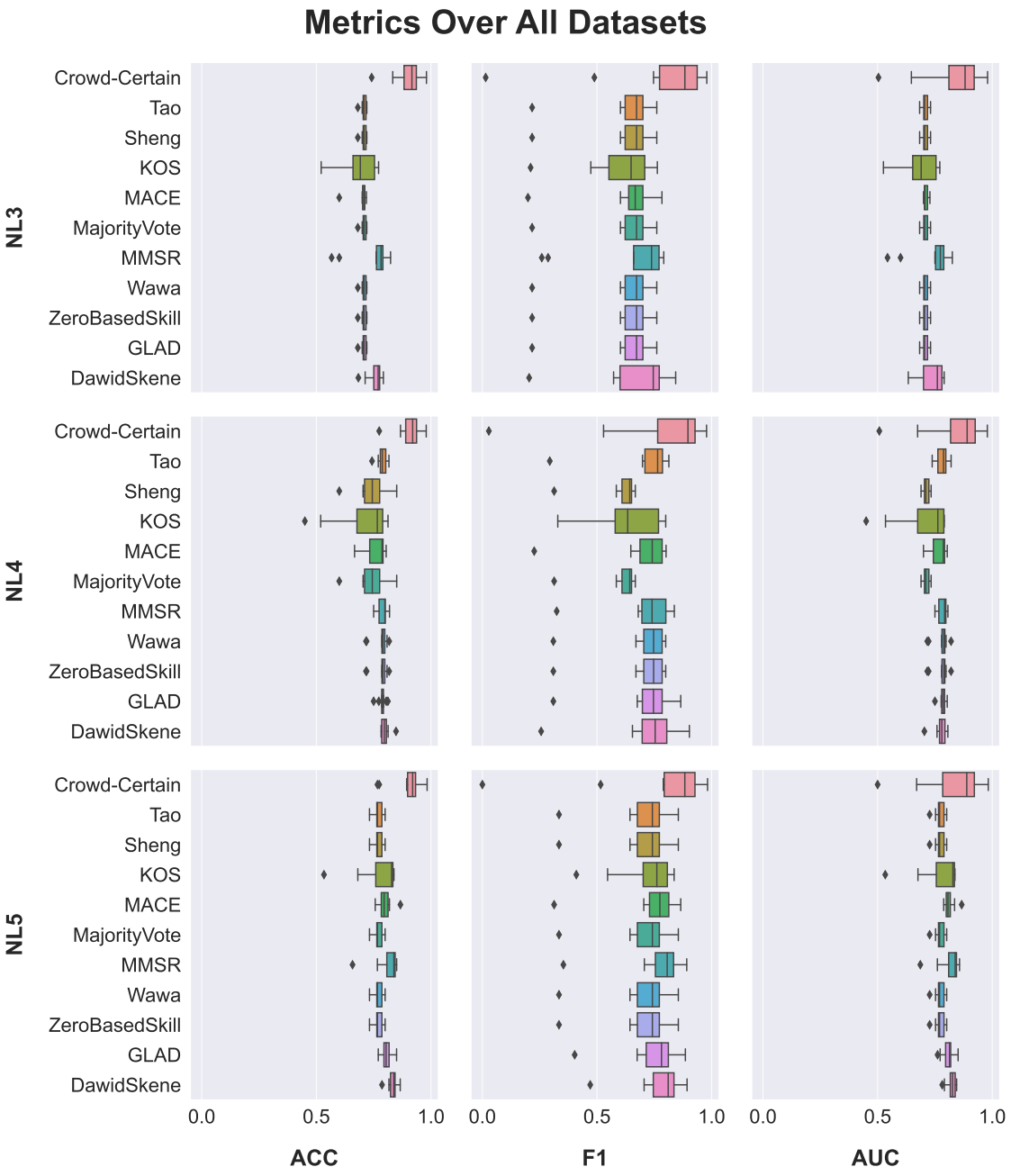

Figure 2.3: Comparison of Accuracy, F1, and AUC scores for multiple label aggregation techniques for various number of annotators. The mean Accuracy, F1, and AUC scores obtained for each of the 10 datasets (each experiment repeated 3 times each with a distinct random seed and then averaged). The 3 figures on the top row shows the boxplots of these 10 values for each technique in the presence of 3 annotators (NL3). Similarly the two lower rows displays the results for the presence of 4 (NL4) and 5 (NL5) annotators respectively. Prior to calculating the metrics, the aggregated labels were obtained using the predicted probabilities $\eta_\alpha^{(i,k)}$ for the Crowd-Certain and the original annotator's labels $z_\alpha^{(i,k)}$ for other techniques.

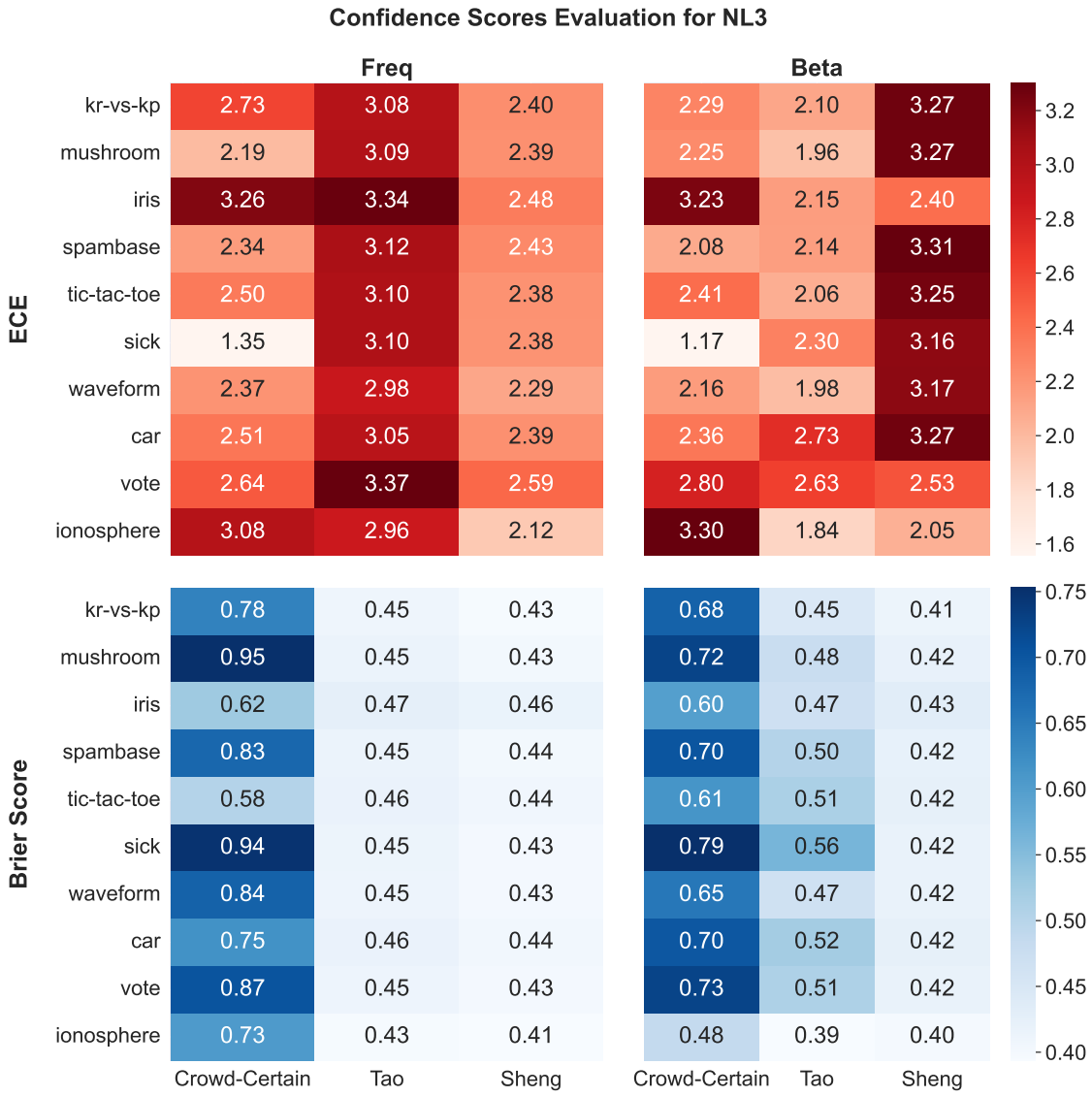

Figure 2.4: Comparison of Expected Calibration Error (ECE) and Brier Score Loss for two confidence score measurement strategies (Freq and Beta) across three different techniques (Crowd-Certain, Tao, and Sheng). Results are shown for ten different datasets for 3 workers (NL3). The metrics reflect the calibration and sharpness of the predictions under different configurations.

and sharper predictions. For the Beta strategy, the performance varies between techniques. The Tao technique generally results in higher ECE and Brier Score Loss, indicating worse calibration and sharper predictions, whereas the Crowd-Certain and Sheng techniques show varying performance depending on the number of annotators.

For the Brier Score Loss, the Freq strategy combined with the Crowd-Certain technique tends to perform better across all numbers of annotators compared to other combinations of techniques and strategies. For the ECE, the Beta strategy combined with the Crowd-Certain technique yields the lowest values for three and four annotators, indicating a good match between predicted confidences and observed frequencies. However, the ECE tends to increase as the number of annotators increases, indicating a decline in calibration.

Overall, these results suggest that the choice of the confidence measurement technique and the strategy has significant impacts on the calibration and sharpness of the predictions. Further investigations could be beneficial to understand the specific conditions under which certain techniques and strategies yield superior performance.

2.5 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 this work, we introduced a new method for crowd labeling aggregation termed as Crowd-Certain. This technique effectively leverages uncertainty measurements to refine the aggregation of labels obtained from multiple annotators. Through an extensive comparative analysis, it was shown to yield higher accuracy in label aggregation against ground truth, particularly in settings where only a limited number of annotators are available. This advantage over established methods such as Gold Majority Vote, MV, MMSR, Wawa, Zero-Based Skill, GLAD, and Dawid Skene demonstrates the potential of the proposed method in enhancing the reliability of label aggregation in crowdsourcing and ensemble learning applications.

Our approach is distinguished by its application of a weighted soft majority voting scheme, where the weights are determined based on the level of uncertainty associated with each annotator's labels. Importantly, the proposed technique takes into account the possibility of consistently inaccurate annotators and includes measures to penalize them (shown in Eq. **??**), thus ensuring the credibility of the computed weights $\omega_{\alpha}^{(k)}$. The calculated weights follow a pre-set ground-truth accuracy closely,

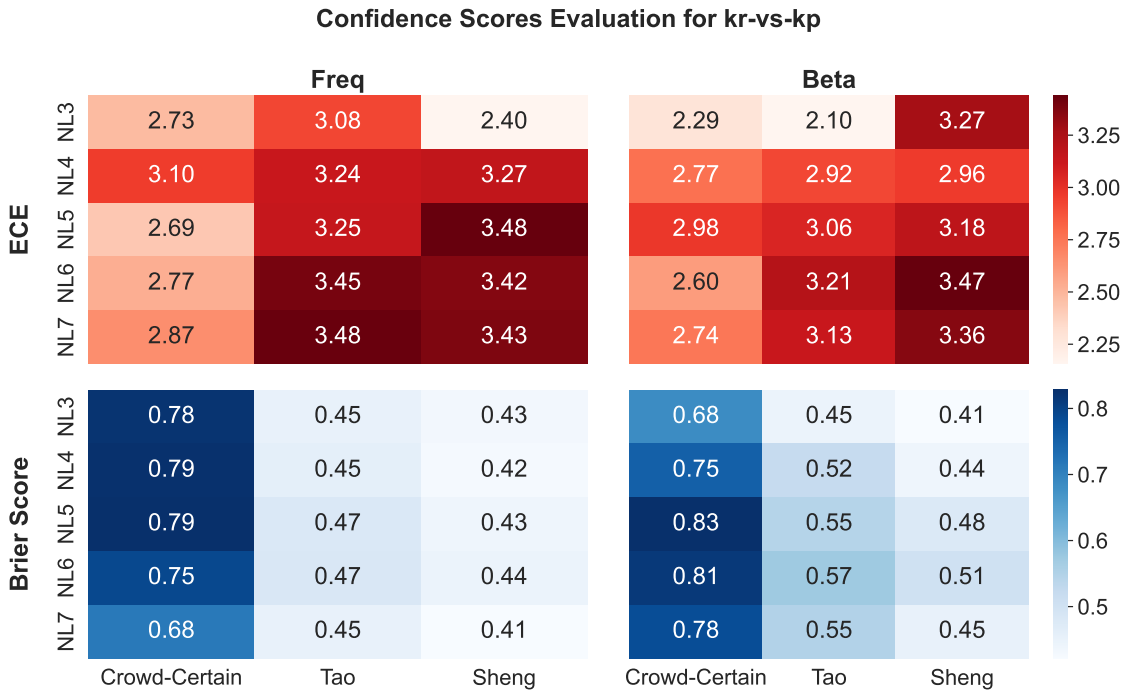

Figure 2.5: Comparison of Expected Calibration Error (ECE) and Brier Score Loss for two confidence score measurement strategies (Freq and Beta) across three different techniques (Crowd-Certain, Tao, and Sheng). Results are shown for varying numbers of annotators (NL3 to NL7) on the kr-vs-kp dataset. The metrics reflect the calibration and sharpness of the predictions under different configurations.

highlighting the effectiveness of the technique in capturing the quality of annotators' labels. Moreover, the Crowd-Certain technique demonstrates an appreciable capability to generate confidence scores that accompany each aggregated label, offering an extended context that can be invaluable in practical applications.

In this study, we evaluated various techniques for aggregating crowd-sourced labels and measuring the confidence scores associated with these labels. This evaluation involved two key metrics: Expected Calibration Error (ECE) and Brier Score Loss for the evaluation of confidence scores, as well as three metrics: accuracy, AUC, and F1 score used to evaluate the aggregated labels. These metrics assessed different facets of model performance: calibration and sharpness of the confidence scores, and the performance of the aggregated labels against the ground truth. By comparison to existing methodologies, which rely on the use of actual crowd labels, our method demonstrates superior performance across a variety of datasets, yielding higher average accuracy rates. Furthermore, our experiments, which involved varying the number of annotators, demonstrated that Crowd-Certain consistently outperforms the benchmark methods in all scenarios, irrespective of the number of annotators involved. This indicates that our approach is not only accurate and efficient, but also highly robust and adaptable to a range of settings, showing consistent improvements across different numbers of annotators.

Significantly, our technique introduces an advantageous property by assigning a single weight to each annotator for all instances in the dataset. Moreover, the application of predicted probabilities in our method, denoted as $\eta_{\alpha}^{(i,k)}$, allows for the reuse of trained classifiers on future sample data, which eliminates the need for recurrent simulation processes. This presents a distinct advantage over conventional techniques, which require computationally expensive and time-consuming repeated simulations for every new dataset [**?**]. It's worth noting that the novel Crowd-Certain method is not only superior in terms of average accuracy, but also shows a consistent outperforming across all tested datasets. This consistency is evident even when considering variance in dataset characteristics, such as 'kr-vs-kp', 'mushroom', 'spambase', and 'tic-tac-toe', further attesting to the robustness and versatility of our approach.

In addition to label aggregation, the evaluation of confidence score measurements revealed further advantages of the Crowd-Certain method. When analyzing two confidence score measurement techniques, Freq and Beta, we found that our strategy consistently achieved lower Expected Calibration Error (ECE) scores compared to other techniques, such as Tao and Sheng. This implies that Crowd-Certain provides better-calibrated predictions, offering a higher level of confidence in the aggregated labels. Furthermore, Crowd-Certain also outperformed other techniques in terms of Brier Score Loss across

most datasets, indicating a higher accuracy of probabilistic predictions.

Our results indicate that the choice of aggregation and confidence measurement technique can significantly impact the performance. Specifically, the Crowd-Certain technique consistently performed well when combined with the Beta strategy, yielding lower ECE and Brier Score Loss across different numbers of annotators. This suggests that the Crowd-Certain technique can effectively aggregate crowd-sourced labels and produce well-calibrated and sharp confidence scores when applied using the Beta strategy. We also noted that increasing the number of annotators does not necessarily improve the performance, as indicated by the general increase in ECE and Brier Score Loss with a higher number of annotators. This suggests a trade-off between the number of annotators and the performance, and that the optimal number may depend on the specific context and the chosen techniques.

2.6 Conclusion

The proposed Crowd-Certain label aggregation technique offers a promising solution for crowdsourced labeling tasks. It not only provides superior accuracy but also demonstrates robustness and adaptability across various settings. Furthermore, it improves computational efficiency by allowing for the reuse of trained classifiers on future sample data, making it a viable option for large-scale data labeling tasks. While our findings are encouraging, further research and validation across more diverse datasets and real-world scenarios are warranted to further refine and enhance this approach. Future work could delve deeper into understanding why certain techniques and strategies outperform others under specific conditions. Further investigations could explore the effects of other factors such as the complexity of the task and the diversity of the crowd, which may impact the performance of different techniques and strategies. Our findings could guide future research and applications in this domain, with potential implications for various fields that rely on crowd-sourced data, including machine learning, data science, and citizen science.

2.7 Availability of Data and Materials

The code can be found in Crowd-Certain

2.8 Appendices

List of abbreviations

Competing interests

Acknowledgements