2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Multi-annotator Probabilistic Active Learning

Marek Herde, Daniel Kottke, Denis Huseljic, Bernhard Sick
Intelligent Embedded Systems
University of Kassel, Germany
{marek.herde | daniel.kottke | dhuseljic | bsick}@uni-kassel.de

***Abstract*—Classifiers require annotations of instances, i.e., class labels, for training. An annotation process is often costly due to its manual execution through human annotators. *Active learning* (AL) aims at reducing the annotation costs by selecting instances from which the classifier is expected to learn the most. Many AL strategies assume the availability of a single omniscient annotator. In this article, we overcome this limitation by considering multiple error-prone annotators. We propose the novel AL strategy *multi-annotator probabilistic active learning* (MaPAL). Due to the nature of learning with error-prone annotators, it must not only select instances but annotators, too. MaPAL builds on a decision-theoretic framework and selects instance-annotator pairs maximizing the classifier's expected performance. Experiments on a variety of data sets demonstrate MaPAL's superior performance compared to five related AL strategies.**

I. Introduction

The ease of collecting data and improving computational power promote the development of novel machine learning models [1]. These models learn from data sets containing instances. In many applications, such as the training of classifiers, annotations, i.e., class labels, of the instances are required. The corresponding annotation process is often costly and time-consuming due to its manual execution through human annotators. *Active learning* (AL) aims at reducing these annotation costs. Therefor, an AL strategy selects instances from which the classifier is expected to learn the most [2]. AL strategies have been already employed in several applications, e.g., malware detection [3] and training of robots [4]. Many of these AL strategies assume the availability of a single omniscient annotator [5] providing the correct annotation for each instance. This assumption conflicts with the available sources of annotations. In particular, crowd-sourcing is a common way to obtain annotations [6]. However, on crowd-sourcing platforms, e.g., Amazon's MTurk [7], multiple error-prone annotators must be considered [8]. In this article, we remove the limiting requirement for the availability of a single omniscient annotator. Instead, we allow multiple independent, error-prone, but benevolent annotators. Our contributions are:

- We propose the AL strategy *multi-annotator probabilistic active learning* (MaPAL). It estimates the annotation performance of error-prone annotators as a function of instances. Based on these annotation performance estimates, MaPAL jointly selects an instance-annotator pair maximizing the classifier's expected performance.
- Comparisons of our AL strategy MaPAL to five related AL strategies in an experimental evaluation over a variety of data sets show MaPAL's superior and robust performance.

In Section II, we formalize the problem setting and in Section III, we describe the general structure of AL strategies approaching the problem. Based on this structure, we discuss AL strategies being most related to our problem setting in Section IV. We introduce MaPAL and a model for estimating annotation performances to improve the joint selection of instance-annotator pairs in Section V. In Section VI, we compare MaPAL to several related AL strategies in an experimental evaluation. We conclude the article and formulate open research tasks in Section VII.

II. Problem Setting

We assume that each instance is described by a feature vector $\mathbf{x} = (x_1, \ldots, x_K)^{\mathrm{T}}, K \in \mathbb{N}$, in a K-dimensional feature space Ω_X (e.g., $\Omega_X = \mathbb{R}^K$). The observed instances $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subseteq \Omega_X, N \in \mathbb{N}$ are independently drawn from an unknown distribution $P(X)$. Each instance $\mathbf{x}_n$ belongs to a true class $y_n \in \Omega_Y$ with $|\Omega_Y| = C \in \mathbb{N}_{\geq 2}$ being the number of classes. The true class label y_n is drawn from an unknown categorical distribution $P(Y \mid X = \mathbf{x}_n)$. The set of annotators is denoted by $\mathcal{A} = \{a_1, \ldots, a_M\}$. Each annotator can be queried to provide class labels as annotations at any time. We denote $z_{n,m} \in \Omega_Y \cup \{\odot\}$ as the annotation of annotator a_m for instance $\mathbf{x}_n$. Thereby, $z_{n,m} = \odot$ means that the annotation has not yet been acquired. We summarize the annotations for an instance $\mathbf{x}_n$ in a vector $\mathbf{z}_n = (z_{n,1}, \ldots, z_{n,M})^{\mathrm{T}}$. Additionally, we introduce weights for annotations $\mathbf{w}_n = (w_{n,1}, \ldots, w_{n,M})^{\mathrm{T}} \in [0,1]^M$. A weight allows to control the importance of an annotation, e.g., the probability that an annotation is correct. Together, the observed instances, the annotations, and the weights build a data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{z}_1, \mathbf{w}_1), \ldots, (\mathbf{x}_N, \mathbf{z}_N, \mathbf{w}_N)\} \in \Omega_D$. The data set $\mathcal{D}$ is dynamic because the annotation process leads to updates of the annotations and weights. At the start of an annotation process, there are often no annotations such that the data set takes the form $\mathcal{D} = \{(\mathbf{x}_1, \vec{\odot}, \mathbf{1}), \ldots, (\mathbf{x}_N, \vec{\odot}, \mathbf{1})\}$ with $\vec{\odot}$ and $\mathbf{1}$ being M-dimensional vectors. The set of all possible data sets is denoted by Ω_D. On a data set $\mathcal{D}$, we can train a classifier with parameters $\boldsymbol{\theta}_{\mathcal{D}}$. The trained classifier gives rise to a function $\hat{y} : \Omega_X \to \Omega_Y$ that predicts a class label $\hat{y}(\mathbf{x} \mid \boldsymbol{\theta}_{\mathcal{D}}) \in \Omega_Y$ for a given instance $\mathbf{x} \in \Omega_X$.

Given these preliminaries, we define the **objective of an AL strategy** as determining the optimal data set $\mathcal{D}^*$, such that the classifier's misclassification risk $R(\boldsymbol{\theta}_{\mathcal{D}^*}) \in \mathbb{R}_{\geq 0}$ is minimal

978-1-7281-8808-9/20/$31.00 ©2020 IEEE

2020 25th International Conference on Pattern Recognition (ICPR) | 978-1-7281-8808-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPR48806.2021.9412298

given a fixed budget $B \in \mathbb{N}$ of annotation acquisitions:

$$\mathcal{D}^* = \underset{\mathcal{D} \in \Omega_D}{\arg\min} \left(R(\boldsymbol{\theta}_{\mathcal{D}}) \right)$$
$$\text{subject to } \left(\sum_{(\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}} \sum_{m=1}^{M} I(z_{n,m} \neq \odot) \right) = B, \tag{1}$$

where $I(c)$ returns one if the condition c is met and zero otherwise. The misclassification risk $R(\boldsymbol{\theta}_{\mathcal{D}})$ is defined based on the zero-one loss [9]. By taking the expectation with respect to the true joint distribution $P(X, Y)$, we obtain:

$$R(\boldsymbol{\theta}_{\mathcal{D}}) = \mathop{\mathbb{E}}_{P(X=\mathbf{x}, Y=y)} \left[I(y = \hat{y}(\mathbf{x} \mid \boldsymbol{\theta}_{\mathcal{D}})) \right]. \tag{2}$$

III. Active Learning Cycle with Multiple Error-prone Annotators

It is hardly possible to specify the annotation process for obtaining the optimal data set $\mathcal{D}^*$ given by Eq. 1 in advance [10]. Therefore, an AL strategy aims at approximating the optimal solution through a greedy approach. Thus, the data set $\mathcal{D}$ is updated iteratively by executing a cycle. In each cycle iteration, an AL strategy selects one or multiple instance-annotator pairs by specifying a query set $\mathcal{S} \subseteq \{(\mathbf{x}_n, a_m) \in \mathcal{X} \times \mathcal{A} \mid z_{n,m} = \odot\}$. For each selected instance-annotator pair $(\mathbf{x}_n, a_m) \in \mathcal{S}$, the annotator a_m is queried to provide a class label $z_{n,m} \in \Omega_Y$ as annotation for the instance $\mathbf{x}_n$. We assume that an annotator a_m cannot be repeatedly selected to annotate the instance $\mathbf{x}_n$.

According to Fig. 1, which visualizes the selection procedure for a single cycle iteration, an AL strategy consists of three main components, i.e., an annotator model represented by the parameters $\boldsymbol{\omega}_{\mathcal{D}}$, a classifier represented by the parameters $\boldsymbol{\theta}_{\mathcal{D}}$, and a selection algorithm. The parameters of the classifier and the annotator model are determined on the current data set $\mathcal{D}$. The **annotator model** makes assumptions regarding the behaviors of annotators [11]. For example, it assumes independence between the annotators. Given the current data set $\mathcal{D}$, its main task is to estimate the actual annotation performance through a function $\psi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$. The output $\psi(\mathbf{x}_n, a_m \mid \boldsymbol{\omega}_{\mathcal{D}})$ represents the estimated performance value of an annotator a_m regarding an instance $\mathbf{x}_n$. A common measure for a performance value is the probability of providing a correct annotation [12], [13]. In dependence on the **classifier**, an instance utility function $\phi : \mathcal{X} \to \mathbb{R}$ is defined, e.g., *uncertainty sampling* (US) [14]. It computes the utility value $\phi(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}})$ of an observed instance $\mathbf{x}_n$ regarding the classifier with its current parameters $\boldsymbol{\theta}_{\mathcal{D}}$. Based on an instance utility function ϕ and an annotation performance function ψ as inputs, the **selection algorithm** selects instance-annotator pairs in each cycle iteration by specifying the query set $\mathcal{S}$.

IV. Related Work

There exist many AL strategies assuming a single omniscient annotator. In such a case, only an instance utility function and a selection algorithm are to be defined. An overview of these strategies is given in [15]. In this section, we focus on related AL strategies dealing with error-prone annotators.

In 2009, Donmez et al. proposed **IEThresh** [16] as one of the first AL strategies taking error-prone annotators into account. First, it selects the instance with maximum utility based on US. Subsequently, the annotators with performance values above an adaptive threshold are selected for annotating this instance. A performance value is computed as the upper bound of an annotator's estimated mean annotation accuracy. Zheng et al. proposed **IEAdjCost** [17] being closely related to IEThresh. In an initial phase, it explores the performance values of the annotators. Once the estimates of performance values are accurate enough, a subset of the estimated best annotators is determined. The remaining annotators are excluded from the further annotation process. One issue of IEThresh and IEAdjCost is their unrealistic assumption of constant/uniform performance values across the observed instances. Moon et al. aimed at overcoming this issue by proposing **Proactive** [13]. It employs an annotator model estimating class-dependent performance values. This means that the annotators are expected to have constant performance values w.r.t. instances belonging to the same class. For this purpose, it expects a small initial set of instances with their true class labels as annotations. Otherwise, it resorts to approximations, e.g., querying all annotators to annotate an initial set of instances and taking the majority vote annotation for each of these instances. Moreover, instead of selecting an instance in first place and annotators in second place, Proactive takes the product of a US based instance utility and the annotation performance value to ensure that both need to be high. Finally, the instance-annotator pair with the maximum product is selected. The AL strategies **CEAL** [18] (Huang et al.) and **ALIO** [12] (Chakraborty) follow an approach being closely related to Proactive. However, they go one step further and assume that the annotation performance values are instance-dependent. As a result, the actual performance values may vary across the observed instances, e.g., the chance of recognizing a hand-written digit in an image correctly depends on actual values of the image's pixels. Since instance-dependent performance values are the most general type of annotation performance, our AL strategy MaPAL also estimates performance values in dependence on annotator and instance. Compared to CEAL and ALIO, it does not require an initial set of instances with (approximated) true class labels but can start without any acquired annotation.

V. Multi-annotator Probabilistic Active Learning

In this section, we explain the necessary steps toward MaPAL's final selection algorithm. These explanations include its instance utility and annotation performance function.

A. Class Frequency Estimates

A requirement of MaPAL is a classifier estimating so-called *class frequency estimates* (CFE) $\mathbf{f}_{\mathbf{x}}^{\mathcal{D}} \in \mathbb{R}_{\geq 0}^{C}$. They describe the annotation status in the neighborhood of an instance $\mathbf{x}$. Several types of classifiers can provide these CFE [19], in particular generative classifiers [20]. In this article, we obtain them from a *similarity-based classifier* (SbC). It uses a similarity function $s : \Omega_X \times \Omega_X \to [0, 1]$ with $s(\mathbf{x}, \mathbf{x}')$ as the similarity

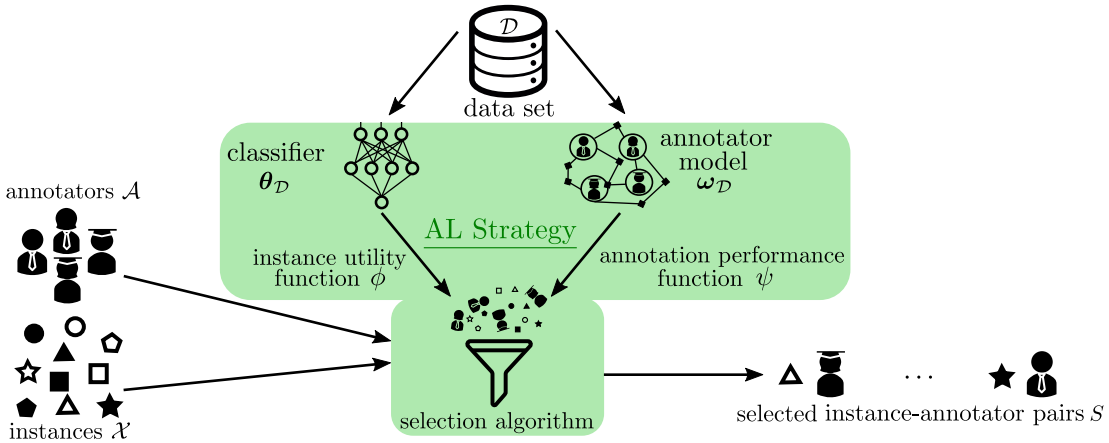

Fig. 1: Selection procedure of an AL strategy (marked in green) dealing with error-prone annotators.

value of two instances $\mathbf{x}$ and $\mathbf{x}'$. This value is to be maximum for two equal instances: $s(\mathbf{x}, \mathbf{x}) = 1$. Based on such a similarity function s and a data set $\mathcal{D}$, we define the y-th element of the vector $\mathbf{f}_{\mathbf{x}}^{\mathcal{D}}$ as

$$f_{\mathbf{x},y}^{\mathcal{D}} = \sum_{(\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}} s(\mathbf{x}, \mathbf{x}_n) \sum_{m=1}^{M} w_{n,m} \cdot I(z_{n,m} = y), \quad (3)$$

which is a proxy of the number of annotations of class y in the neighborhood of instance $\mathbf{x}$. The annotation weights, i.e., $w_{n,m}$, are used to control the influence of acquired annotations. Given the CFE, the SbC predicts the most frequent class:

$$\hat{y}(\mathbf{x} \mid \boldsymbol{\theta}_{\mathcal{D}}) = \hat{y}(\mathbf{x} \mid \mathcal{D}) = \arg\max_{y \in \Omega_Y} \left(f_{\mathbf{x},y}^{\mathcal{D}} \right). \quad (4)$$

We additionally employ the CFE to estimate the true but unknown class membership probabilities of an instance. To reflect the uncertainty of our estimates, we rely on a Bayesian approach by introducing a prior $\boldsymbol{\alpha} \in \mathbb{R}_{>0}^{C}$. An entry α_y of this vector represents the prior number of observed annotations of class y. Finally, the class membership probability of instance $\mathbf{x}$ for class y is estimated according to

$$P(Y = y \mid X = \mathbf{x}) \approx P(Y = y \mid \mathbf{f}_{\mathbf{x}}^{\mathcal{D}}, \boldsymbol{\alpha}) = \frac{f_{\mathbf{x}_n,y}^{\mathcal{D}} + \alpha_y}{||\mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}} + \boldsymbol{\alpha}||_1}. \quad (5)$$

In fact, the approximation in Eq. 5 is the expected value of a Dirichlet distribution whose derivation is given in [21]. Moreover, this kind of probability estimation is a central part of MaPAL and we make use of it throughout this section.

B. Probabilistic Active Learning for Error-Prone Annotations

The instance utility function of our AL strategy MaPAL is based on the framework of *probabilistic active learning* (PAL) [20], [21]. The major difference is that PAL assumes a single omniscient annotator, whereas we have multiple error-prone annotators. The idea of our approach is to estimate the misclassification risk R in Eq. 2 while explicitly considering erroneous annotations. Therefor, we need to approximate the joint distribution $P(X, Y) = P(Y \mid X)P(X)$. The true class membership probabilities are estimated according to Eq. 5 with the prior $\boldsymbol{\alpha} = \mathbf{1}$:

$$P(Y = y \mid X = \mathbf{x}_n) \approx P(Y = y \mid \mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}}, \mathbf{1}). \quad (6)$$

This way, we assume a uniform and unbiased (i.e., uninformative) prior distribution over the unknown class membership probabilities of an instance. Since our observed instances in the data set $\mathcal{D}$ are independently drawn from the distribution $P(X)$, we employ Monte Carlo integration for the estimation of $P(X)$. Then, the final risk estimate is computed as

$$R(\boldsymbol{\theta}_{\mathcal{D}}) \approx R(\boldsymbol{\theta}_{\mathcal{D}} \mid \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}} \sum_{y \in \Omega_Y} P(Y = y \mid \mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}}, \mathbf{1}) \cdot L(y, \hat{y}(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}})). \quad (7)$$

Next, we assume that we acquire new annotations. Updating the corresponding entries in the data set $\mathcal{D}$ leads to a new data set $\mathcal{D}^+$. To determine the classifier's performance change resulting from newly acquired annotations, we define the performance gain as the reduction in the estimated risk:

$$\Delta R(\boldsymbol{\theta}_{\mathcal{D}}, \boldsymbol{\theta}_{\mathcal{D}^+} \mid \mathcal{D}^+) = R(\boldsymbol{\theta}_{\mathcal{D}} \mid \mathcal{D}^+) - R(\boldsymbol{\theta}_{\mathcal{D}^+} \mid \mathcal{D}^+). \quad (8)$$

Both risk estimates are computed with respect to the updated data set $\mathcal{D}^+$ because it contains more annotations (i.e., information) than $\mathcal{D}$ and we assume that these annotations are in average better than random guesses. Since $\mathcal{D}^+$ is unknown in advance, we need to simulate the possible outcomes of the annotation acquisitions. Therefore, we define a new data set separately for each annotator a_m:

$$\mathcal{D}_m = \{(\mathbf{x}_n, \mathbf{z}_n, \mathbf{e}_m) \mid (\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D} \wedge z_{n,m} \neq \odot\}, \quad (9)$$

with $\mathbf{e}_m \in [0, 1]^M$ as M-dimensional unit vector with a 1 at index m. In $\mathcal{D}_m$, each annotation of the annotator a_m has a maximum weight of 1, while the annotations of the other annotators $\mathcal{A} \setminus \{a_m\}$ are ignored by assigning a minimum weight of 0. Following Eq. 5, we estimate the probability that a_m provides class label y as annotation for $\mathbf{x}_n$ according to

$$P(Z_m = y \mid X = \mathbf{x}_n) \approx P(Z_m = y \mid \mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}_m}, \mathbf{1}), \quad (10)$$

where Z_m denotes the random variable of the annotations of annotator a_m. By relying on Eq. 10, we assume that an annotator provides similar annotations for similar instances. Next, we simulate the case when multiple annotators are queried to annotate the same instance. In particular, we are interested in the probability of observing different realizations of the annotations. Without loss of generality, let us assume we query the annotators $\mathcal{A}' = \{a_1, \ldots, a_{M'}\} \subseteq \mathcal{A}$. Then, we estimate the probability of observing the class labels $\mathbf{y} = (y^{(1)}, \ldots, y^{(M')})^{\mathrm{T}} \in \Omega_Y^{|\mathcal{A}'|}$ as annotations for $\mathbf{x}_n$ as:

$$P(Z_1 = y^{(1)}, \ldots, Z_{M'} = y^{(M')} \mid X = \mathbf{x}_n) \tag{11}$$

$$= \prod_{a_m \in \mathcal{A}'} P(Z_m = y^{(m)} \mid X = \mathbf{x}_n) \tag{12}$$

$$\approx \prod_{a_m \in \mathcal{A}'} P(Z_m = y^{(m)} \mid \mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}_m}, \mathbf{1}) = P(\mathbf{y} \mid \mathbf{x}_n, \mathcal{A}'). \tag{13}$$

The factorization in Eq. 12 is reasonable because of our assumption of independent annotators. Now, we compute the average expected performance gain for querying the annotators $\mathcal{A}'$ to annotate $\mathbf{x}_n$. To obtain the exact expectation, we would need to iterate over all $|\Omega_Y^{|\mathcal{A}'|}| = C^{M'}$ possible combinations of class labels. Hence, the complexity increases strongly with the number of classes and annotators. As a more tractable alternative, we propose the following gain approximation:

$$G(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}}, \mathcal{D}, \mathcal{A}') = \frac{1}{\mid \mathcal{A}' \mid} \sum_{y \in \Omega_Y} P(y \cdot \mathbf{1} \mid \mathbf{x}_n, \mathcal{A}') \cdot \Delta R(\boldsymbol{\theta}_{\mathcal{D}}, \boldsymbol{\theta}_{\mathcal{D}^+} \mid \mathcal{D}^+), \tag{14}$$

where $\mathcal{D}^+$ denotes the data set after updating the triple $(\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}$ by setting $z_{n,1} = y, \ldots, z_{n,M'} = y$. The vector $y \cdot \mathbf{1} = (y, \ldots, y)^{\mathrm{T}} \in \Omega_Y^{|\mathcal{A}'|}$ represents the case, when all annotators $\mathcal{A}'$ provide the same class label y as annotation. We consider only those updates as they lead to the highest risk differences ΔR. Moreover, this way, the number of summations in Eq. 14 equals C instead of $C^{M'}$.

C. Beta Annotator Model

So far, we did not explicitly define the annotation weights $\mathbf{w}_n \in [0,1]^M$ of a triple $(\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}$. Since we deal with error-prone annotators, we define them through instance-dependent annotation performances: $w_{n,m} = \psi(\mathbf{x}_n, a_m \mid \boldsymbol{\omega}_{\mathcal{D}})$. For estimating them, we propose the *Beta annotator model* (BAM), which outputs the expected probability that an annotator provides a correct annotation for a given instance. For an annotator a_m, we solve a binary classification problem with the classes $\Omega'_Y = \{1, 2\}$. A class label $y'_{n,m} = 2$ indicates a correct annotation of annotator a_m for instance $\mathbf{x}_n$, whereas $y'_{n,m} = 1$ indicates a false annotation. These class labels are unknown and have to be estimated. Therefor, we determine CFE on the data set

$$\mathcal{D}_{\overline{m}} = \{(\mathbf{x}_n, \mathbf{z}_n, \mathbf{1} - \mathbf{e}_m) \mid (\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}\}. \tag{15}$$

In $\mathcal{D}_{\overline{m}}$, the annotations of a_m have minimum weights of 0, i.e., they are ignored, and the annotations of the remaining annotators $\mathcal{A} \setminus \{a_m\}$ have maximum weights of 1, i.e., they have equal importance. The CFE $\mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}_{\overline{m}}}$ are to estimate the true class label y_n of instance $\mathbf{x}_n$. The estimated true class labels are used to evaluate the performance of annotator a_m and by considering only the annotations of $\mathcal{A} \setminus \{a_m\}$, annotator a_m cannot bias her/his own performance estimation [22]. Based on the CFE for data set $\mathcal{D}_{\overline{m}}$, we define the training set $\mathcal{D}'_m$ for the above described binary classification problem:

$$\mathcal{D}'_m = \{(\mathbf{x}_n, y'_{n,m}, w'_{n,m}) \mid (\mathbf{x}_n, \mathbf{z}_n, \mathbf{w}_n) \in \mathcal{D}_m\}, \tag{16}$$

$$y'_{n,m} = I(\hat{y}(\mathbf{x}_n \mid \mathcal{D}_{\overline{m}}) = z_{n,m}) + 1, \tag{17}$$

$$w'_{n,m} = 1 - \frac{H[Y|X = \mathbf{x}_n, \mathcal{D}_{\overline{m}}]}{\ln(C)}, \tag{18}$$

where H denotes the entropy. The weight $w'_{n,m}$ is to reflect the entropy-based confidence in our inferred class label $y'_{n,m}$. If the annotators $\mathcal{A} \setminus \{a_m\}$ do not clearly agree on one class, this weight takes a rather low value, as we cannot be certain whether a_m is right or wrong. The CFE $\mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}'_m}$ summarize the estimated numbers of false and true annotations provided by annotator a_m in the neighborhood of instance $\mathbf{x}_n$. Following the Bayesian approach in Eq. 5, we estimate the performance value of annotator a_m regarding instance $\mathbf{x}_n$ according to:

$$\psi(\mathbf{x}_n, a_m \mid \mathcal{D}'_m, \boldsymbol{\beta}) = P(Y_m = 2 \mid \mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}'_m}, \boldsymbol{\beta}) = \frac{f_{\mathbf{x}_n,2}^{\mathcal{D}'_m} + \beta_2}{||\mathbf{f}_{\mathbf{x}_n}^{\mathcal{D}'_m} + \boldsymbol{\beta}||_1}, \tag{19}$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2)^{\mathrm{T}} \in \mathbb{R}^2_{\geq 0}$ is a prior parameter and the random variable Y_m indicates whether annotator a_m provides a false or correct annotation. The performance value in Eq. 19 equals the expectation of a Beta distribution leading to the name of our annotator model. The prior $\boldsymbol{\beta}$ is a hyperparameter, where β_1 can be interpreted as the prior number of false and β_2 as the prior number of true annotations. By parameterizing $\boldsymbol{\beta}$, we integrate prior knowledge about annotators into the estimation of their performance values. It supports us regarding the exploration-exploitation trade-off in annotator selection. In favor for exploring the performance values of the annotators, we define $\boldsymbol{\beta} = \beta_0 \cdot (0,1)^{\mathrm{T}}$ with $\beta_0 \in \mathbb{R}_{>0}$. In fact, this is not a proper prior of a Beta distribution. However, by using it, we are optimistic regarding the performance values of the annotators and assume that each annotator a_m has a maximum prior performance value of $\psi(\mathbf{x}_n, a_m \mid \mathcal{D}'_m, \boldsymbol{\beta}) = 1$. This assumption is important for exploring the performance values across the observed instances of each annotator in an initial learning phase. Increasing the value of hyperparameter β_0 leads to a stronger influence of prior knowledge on the performance estimates. If there are no prior information about the annotators available, we suggest to use a small value, e.g., $\beta_0 = 10^{-4}$.

D. Joint Selection with Ranking of Annotators

The gain function G proposed in Eq. 14 already computes an instance utility value in dependence of the available annotators. However, it does not define a clear selection criterion for the annotators. Moreover, given an instance without any assigned annotations, we would need to iterate over all

$2^{|\mathcal{A}|} - 1$ non-empty subsets $\mathcal{A}' \subseteq \mathcal{A}$ of annotators to compute the estimated performance gain. To reduce computational complexity, we rank the annotators according to their currently estimated performance values per instance. The resulting pairs of instances and rankings of annotators are summarized as

$$\mathcal{R} = \{(\mathbf{x}_1, \mathbf{a}_1), \ldots, (\mathbf{x}_N, \mathbf{a}_N) \mid \mathbf{a}_n = (a_{n_1}, \ldots, a_{n_{M_n}}), \\ w_{n,n_1} \geq \cdots \geq \cdots \geq w_{n,n_{M_n}}\}, \tag{20}$$

where $n_1, \ldots, n_{M_n} \in \{1, \ldots, M\}$ represent the indices of the M_n annotators who have not annotated instance $\mathbf{x}_n$ yet. In a next step, we quantify the utility values of instances in dependence on their respective rankings of annotators. Therefor, we define the final instance utility function

$$\phi(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}}, \mathcal{D}, \mathbf{a}_n, M_{\text{max}}) = \\ \max_{m \in \{1, \ldots, \min(M_{\text{max}}, M_n)\}} \left(G\left(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}}, \mathcal{D}, \{a_{n_1}, \ldots, a_{n_m}\}\right)\right). \tag{21}$$

The instance utility is obtained by taking the maximum of the above computed average performance gains. The hyperparameter $M_{\text{max}} \in \{1, \ldots, M\}$ is to restrict the maximum number of modeled annotation acquisitions per instance. For example, if there are $M_n = 10$ annotators available for annotating instance $\mathbf{x}_n$, we can look ahead for 10 annotation acquisitions. However, modeling many future annotation acquisitions linearly increases the computational complexity. For most problems, $M_{\text{max}} = 2$ being a rather low value is a good compromise between performance and computational complexity. Once the utility values for all instances with their respective rankings of annotators have been computed, MaPAL selects the pair of instance and annotator ranking with maximum utility:

$$(\mathbf{x}_{n^*}, \mathbf{a}_{n^*}) = \underset{(\mathbf{x}_n, \mathbf{a}_n) \in \mathcal{R}}{\arg\max} \left(\phi(\mathbf{x}_n \mid \boldsymbol{\theta}_{\mathcal{D}}, \mathcal{D}, \mathbf{a}_n, M_{\text{max}})\right). \tag{22}$$

Instead of asking all annotators of the vector $\mathbf{a}_{n^*}$ to annotate instance $\mathbf{x}_{n^*}$, MaPAL selects only annotator $a_{n_1^*}$ having the highest rank: $\mathcal{S} = \{(\mathbf{x}_{n^*}, a_{n_1^*})\}$. This way, MaPAL always makes use of the most recent annotations before selecting a new instance-annotator pair. Pseudo-code of the selection procedure is given in Fig. 2 with complexity $\mathcal{O}(N^2 \cdot C^2 \cdot M_{\text{max}})$.

Input: data set $\mathcal{D}$, annotators $\mathcal{A}$, similarity function s, prior number of correct annotations β_0, maximum number of modeled annotation acquisitions M_{max}
Output: query set $\mathcal{S}$

1: for each instance-annotator pair $(\mathbf{x}_n, a_m) \in \mathcal{X} \times \mathcal{A}$ compute the annotation performance value and update the weight of the data set $\mathcal{D}$ according to Eq. 19
2: specify the set $\mathcal{R}$ consisting of pairs of instances and annotator rankings according to Eq. 20
3: compute the utility value for each pair of instance and annotator ranking $(\mathbf{x}_n, \mathbf{a}_n) \in \mathcal{R}$ according to Eq. 21
4: specify the best pair of instance and annotator ranking according to Eq. 22
5: determine the query set $\mathcal{S} = \{(\mathbf{x}_{n^*}, a_{n_1^*})\}$
6: **return** $\mathcal{S}$

Fig. 2: Pseudo-code of MaPAL's selection algorithm: The selection of the query set $\mathcal{S}$ corresponds to one cycle iteration shown in Fig. 1.

VI. Evaluation

In this section, we experimentally evaluate the performance of MaPAL on data sets with error-prone annotators. First, we analyze MaPAL's annotation acquisition behavior on a two-dimensional toy data set. Second, we compare it quantitatively to five related AL strategies (cf. Section IV) and test its robustness in settings with different assumptions regarding the performances of annotators. Code to reproduce results is available at our repository https://github.com/mherde/mapal.

A. Annotation Acquisition Behavior of MaPAL

To give an intuitive understanding of MaPAL, we visualize its application on a two-dimensional toy data set in Fig. 4. The annotation performance estimates of the BAM are low (i.e., light green) in regions where the respective annotator a_m provided annotations disagreeing with the predictions of the SbC using the data set $\mathcal{D}_{\overline{m}}$. In regions, where an annotator provided no annotations, the performance estimates are high (i.e., dark green) because our optimistic prior knowledge is the only information we have. Looking at the distribution of the actively acquired annotations, we observe that MaPAL queried annotators with high performance estimates more frequently. For example, annotator a_2 was queried the most since she/he made no annotation mistakes. In contrast, annotator a_1 was queried only four times because half of her/his annotations were wrong. Nevertheless, this annotator would be not excluded from an ongoing annotation process as the annotation performance is estimated locally. For example, annotator a_1 could be still queried to annotate instances in the upper left or lower right region of the feature space. This example is supported by the plots visualizing instance utility values as functions of the annotators, where we observe rather high utility values in the aforementioned regions for annotator a_1. In general, instance utility is high in regions close to the decision boundary of the SbC using data set $\mathcal{D}$, in regions with only a few acquired annotations, and in regions with high density.

We gain further insights into MaPAL by analyzing its hyperparameters β_0 and M_{max}. Therefor, we applied MaPAL with different parametrizations on a larger version (i.e., 1000 instances) of the toy data set in Fig. 4. The obtained results were obtained following a similar setup as in Section VI-C and are shown in Fig. 3. The learning curves confirm the benefit of increasing M_{max}. Comparing $M_{\text{max}} = 1$ and $M_{\text{max}} = 3$, we notice that $M_{\text{max}} = 3$ leads to lower misclassification rates being also more robust indicated by smaller standard errors. Especially after 80 annotations, $M_{\text{max}} = 1$ may be not sufficient since acquiring one additional annotation does not affect the decision boundary of the classifier, i.e., the performance gain G is zero. However, the selection of a larger M_{max} comes at the expense of increased computational complexity.

The bar chart in Fig. 3 indicates that increasing the value of β_0 leads to a more uniform distribution w.r.t. the number

of annotations queried from the different annotators. This behavior corresponds to a rather strong exploration regarding the performances of the annotators. By decreasing the value of β_0, MaPAL tends to exploit the performance estimates of the annotators more strongly. For example, MaPAL with $\beta_0 = 10^{-4}$ queried annotator a_2 about 60 times, whereas annotator a_4 was queried only about 40 times. Hence, by adjusting the value of β_0 we can balance the trade-off between exploration and exploitation regarding the annotation performances.

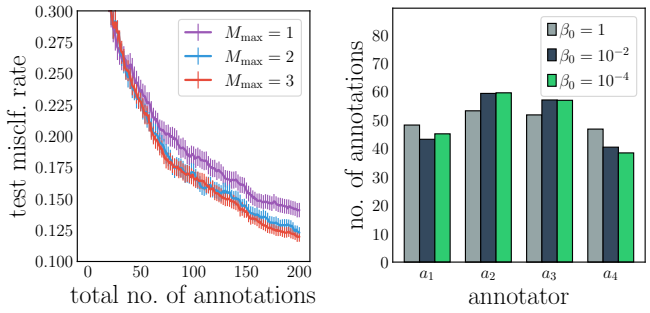

Fig. 3: Study of hyperparameters on a toy data set (cf. Fig. 4): The left plot shows MaPAL's test misclassification rate for three different values of $M_{\max}$ and $\beta_0 = 10^{-4}$ as learning curves. The error bars indicate the standard error. We notice that increasing the value of $M_{\max}$ leads to lower misclassification rates. The right plot shows how often MaPAL has queried an annotator for three different values of β_0 and $M_{\max} = 2$. We observe that increasing the value of β_0 leads to a more uniform distribution of the provided annotations among the different annotators.

B. Data sets and Baselines

We conducted experiments on 29 data sets. Four of these data sets were annotated by real-world annotators. For each of the remaining 25 data sets, we employed three techniques to simulate annotators with different assumptions regarding their annotation performances. (1) Simulation of uniform performances: Given an instance, the probability of providing a correct annotation depended only on the annotator. (2) Simulation of class-dependent performances: Given an instance, the probability of providing a correct annotation depended on both, the annotator and the instance's true class. (3) Simulation of instance-dependent performances: We performed a k-means clustering [23] ($k = M$) on the instances of a data set. Given an instance, the probability of providing a correct annotation depended on both, the annotator and the cluster to which this instance belonged. The number of annotators varied between four to six across all data sets. After simulation, there were in fact 75 data sets with simulated annotators (i.e., 25 for each of the three simulation techniques). In our repository, more details on the simulation procedures are given. Moreover, we list all used data sets including their references and provide specific characteristics such as the number of instances, features, instances per class, and the actual annotation accuracies of the annotators. We compared MaPAL with $M_{\max} = 2$ and $\beta_0 = 10^{-4}$ to five related AL strategies, namely IEThresh [24], IEAdjCost [17], Proactive [13], CEAL [18], and ALIO [12]. Their hyperparameters were set according to recommendations in the respective articles and their exact values are also given in our repository. As a baseline AL strategy, we implemented *Random*, which randomly selects instance-annotator pairs.

C. Experimental Setup

Each experiment, i.e., testing an AL strategy on a data set, was ran 100 times. In each run, we randomly split the data set into a training set consisting of 60% of the instances and a test set containing the remaining 40%. The training set did not contain any annotations at the start of each run. The annotation process was stopped, when either 40% of the available annotations were acquired or the limit of 1000 annotation acquisitions was reached. For classification, we employed the proposed SbC. For two data sets, namely compendium and mozilla, the *cosine similarity* kernel was employed because both data sets deal with text classification. For the remaining data sets, the SbC used a *radial basis function* kernel as similarity function, whose bandwidth parameter was set according to the mean bandwidth criterion proposed in [25].

D. Results

Since we conducted experiments on many data sets, it is difficult to compare the quantitative performances of the AL strategies based on learning curves, which indicate the test misclassification error after each annotation acquisition. In favor of a clear presentation of the results, we followed the evaluation procedure in [21] and computed the *area under the learning curve* (AULC) for each run of an experiment. As a result, there are 100 AULC values for each AL strategy per data set. We ranked these values between the same runs of the different AL strategies and averaged the ranks for each AL strategy. We visualize the resulting mean ranks for each combination of a data set and an AL strategy in Fig. 5: dark green means good rank, whereas light green means bad rank.

Starting with the results for *simulated annotators with uniform performance values* in Fig. 5a, we observe that MaPAL performed best on 17 of 25 data sets. In particular, outperforming IEThresh and IEAdjCost is remarkable because their assumption of uniform performance values is met on these data sets, whereas MaPAL makes the more general assumption of instance-dependent performance values. Regarding the data sets with *simulated annotators having class-dependent performance values*, MaPAL's superiority is confirmed for 20 of 25 data sets. Comparing these results to the ranking statistics for *simulated annotators with instance-dependent performance values* in Fig. 5a, the performance of MaPAL strongly increases in comparison to the other AL strategies. Here, MaPAL performed best on 22 of 25 data sets. This improvement is an indicator that the BAM provides reliable estimates of the instance-dependent annotation performance values. Regarding the results with *real-world annotators* in Fig. 5b, MaPAL performed best on three of four data sets. Inspecting Fig. 5c, the mean ranks of MaPAL confirm its superior and robust performance across all four tested data set collections with different types of annotators, i.e., uniform,

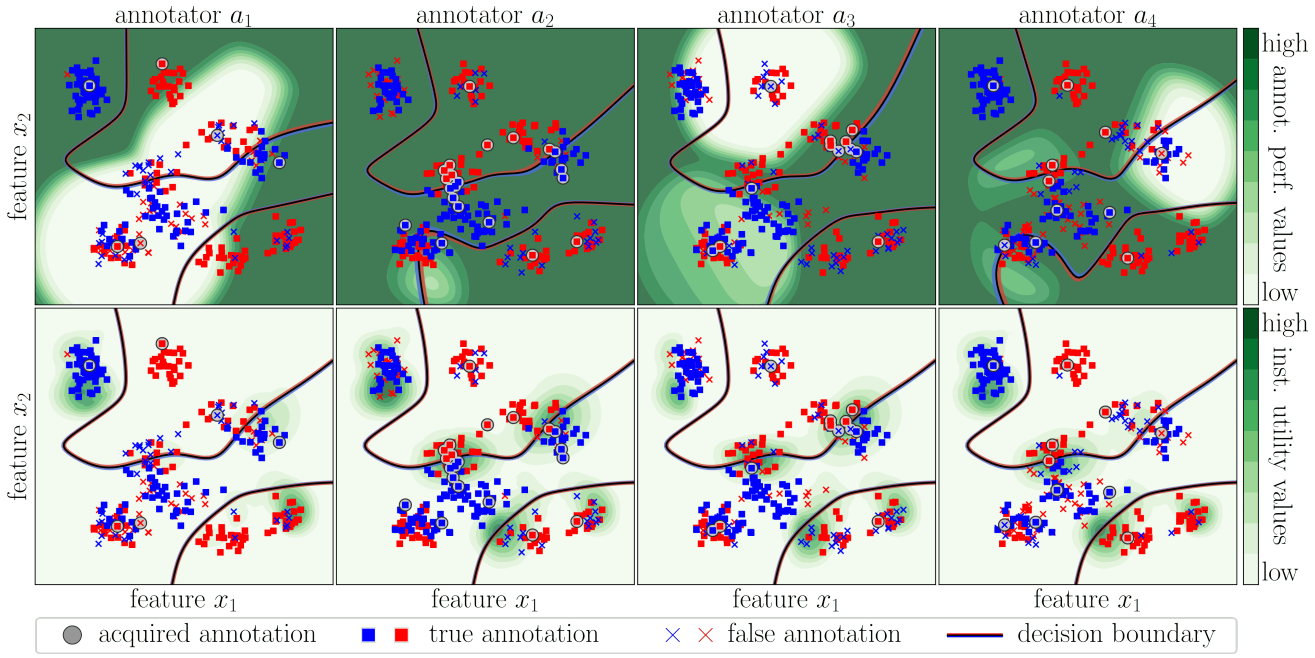

Fig. 4: Visualization of MaPAL's annotation acquisition behavior with $\beta_0 = 10^{-4}$ and $M_{\max} = 2$ after $B = 50$ actively acquired annotations from $M = 4$ annotators: The eight plots visualize the same two-dimensional toy data set with instances of two classes (blue vs. red). There are four simulated annotators with instance-dependent performances (see Section VI-B for more details). The four upper plots show the performance values estimated by the BAM (cf. Eq. 19). The black line of such a plot represents the decision boundary of the SbC using the data set $\mathcal{D}_{\overline{m}}$ (cf. Eq. 15). We notice rather low performance estimates in regions where an annotator provided false annotations indicated by an encircled cross. The four lower plots show the instance utility values as a function of each annotator a_m. Therefor, we redefined the instance utility in Eq. 21 by assuming we would query the respective annotator first, i.e., $a_{n_1} = a_m$. The black line of such a plot represents the decision boundary of the SbC using the data set $\mathcal{D}$. We observe high utility values in dense regions near the SbC's decision boundary, provided that the performance values of the corresponding annotators are high. This way, MaPAL queries annotators in regions with sufficient performance estimates. Further explanations are given in Section VI-A.

class-dep., instance-dep., and real-world. This robustness is not much affected by varying the hyperparameter β_0. Concrete results for five other parameterizations of β_0 are given in our repository. There, we also provide the execution times, learning curves, and the mean AULC values together with their standard deviations for all pairs of AL strategy and data set.

VII. Conclusion

In this article, we proposed the AL strategy MaPAL considering multiple error-prone annotators. Building on a decision-theoretic framework, it estimates the performance gain of querying a class label from a specific annotator for a specific instance. The results of an extensive evaluation, including different types of annotators, indicated the superior and robust performance of MaPAL. Future work will focus on more sophisticated models of the annotation costs [27], the employment of further loss functions next to the zero-one loss, and the incorporation of additional annotator feedback, such as confidence scores [28]. Moreover, we aim to transfer our approach to other types of classifiers, e.g., by computing CFE from the predictive distributions of Gaussian process classifiers. Regarding deep learning methods, we may combine our strategy with evidential deep learning [29], which already estimates a kind of CFE. In both cases, the main challenge will be the computational complexity of MaPAL. Hence, we need to develop efficient incremental training procedures for different classifiers.

References

[1] P. Ongsulee, "Artificial Intelligence, Machine Learning and Deep Learning," in *Int. Conf. on ICT and Knowledge Engineering*, 2017, pp. 1–6.

[2] B. Settles, "Active Learning Literature Survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2010.

[3] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, "Novel active learning methods for enhanced PC malware detection in windows OS," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5843–5857, 2014.

[4] M. Herde, D. Kottke, A. Calma, M. Bieshaar, S. Deist, and B. Sick, "Active Sorting – An Efficient Training of a Sorting Robot with Active Learning Techniques," in *Int. Joint Conf. on Neural Networks*, 2018, pp. 1–8.

[5] B. Settles, "From Theories to Queries: Active Learning in Practice," in *Workshop on Active Learning and Experimental Design*, 2011, pp. 1–18.

[6] J. Howe, "The Rise of Crowdsourcing," *Wired Magazine*, vol. 14, no. 6, pp. 1–5, 2006.

[7] M. Buhrmester, T. Kwang, and S. D. Gosling, "Amazon's Mechanical Turk," *Perspectives on Psychological Science*, vol. 6, no. 1, pp. 3–5, 2011.

[8] J. Zhang, X. Wu, and V. S. Sheng, "Learning from crowdsourced labeled data: a survey," *Artificial Intelligence Review*, vol. 46, no. 4, pp. 543–576, 2016.

[9] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.

[10] P. Donmez and J. G. Carbonell, "Proactive Learning: Cost-Sensitive Active Learning with Multiple Imperfect Oracles," in *ACM Conf. on Information and Knowledge Management*, 2008, pp. 619–628.

[11] R. Gilyazev and D. Turdakov, "Active Learning and Crowdsourcing: A Survey of Optimization Methods for Data Labeling," *Programming and Computer Software*, vol. 44, no. 11, pp. 476–491, 2018.

[12] S. Chakraborty, "Asking the Right Questions to the Right Users: Active Learning with Imperfect Oracles," in *AAAI Conf. on Artificial Intelligence*, 2020.

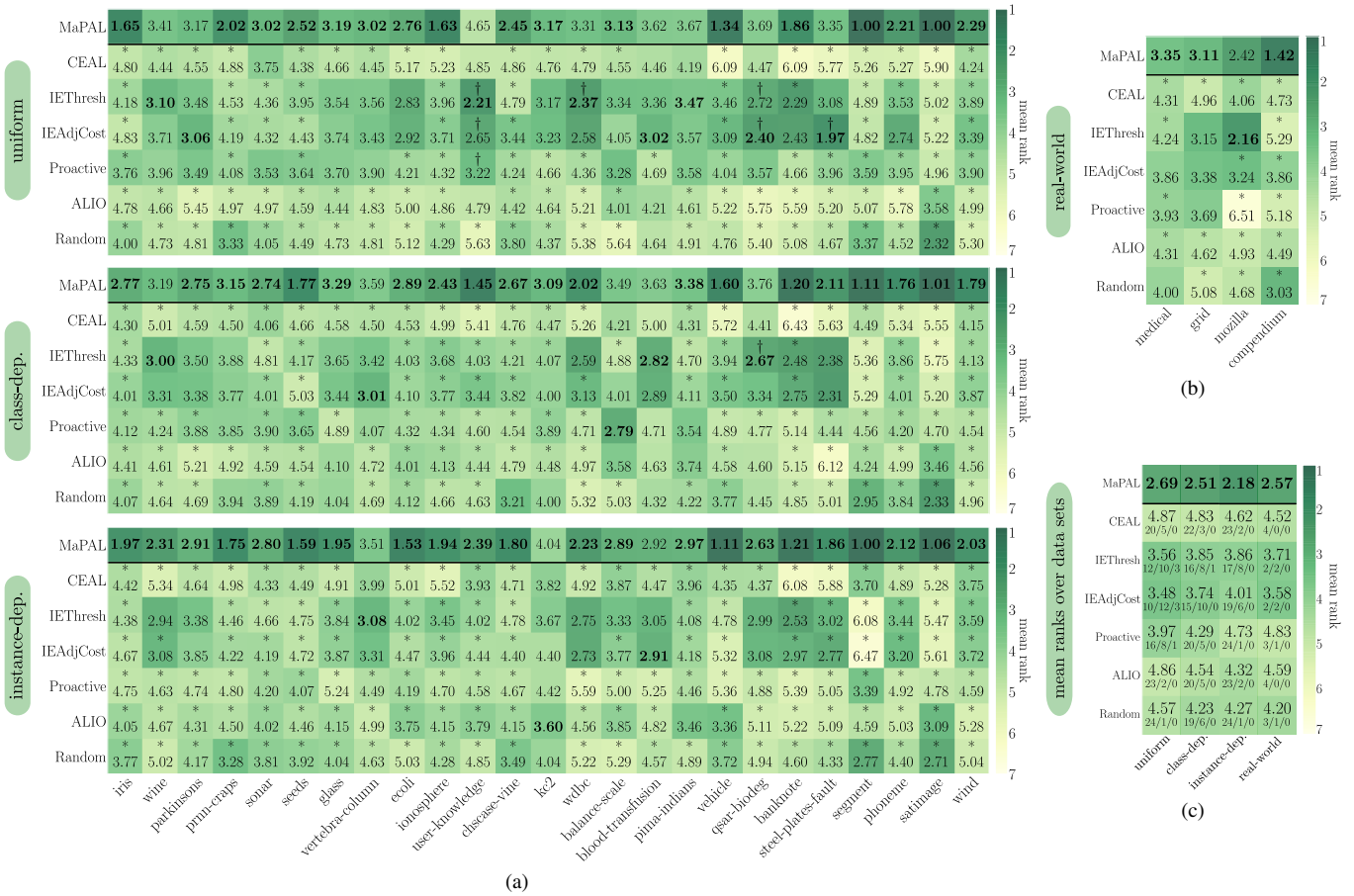

Fig. 5: Ranking statistics: Figs. 5a and 5b show the mean ranks based on the AULC values for all combinations of AL strategies and data sets across 100 runs. The best AL strategy per data set is printed in bold. The Wilcoxon-signed-rank test [26] shows pairwise significance (p-value: 0.001) between MaPAL and its competitor ($*$ MaPAL better, † competitor better). Fig. 5c gives the mean rank of each AL strategy taken over each of the four collections of data sets (i.e., uniform, class-dep., instance-dep., and real-world) in Figs. 5a and 5b. Additionally, for each of these data set collections, it indicates the total number of wins (i.e., number of $*$) / ties / losses (i.e., number of †) of MaPAL in comparison to its respective competitor.

[13] S. Moon and J. G. Carbonell, "Proactive Learning with Multiple Class-sensitive Labelers," in *Int. Conf. on Data Science and Advanced Analytics*, 2014, pp. 32–38.

[14] D. D. Lewis and C. Catlett, "Heterogeneous Uncertainty Sampling for Supervised Learning," in *Machine Learning Proceedings*, 1994, pp. 148–156.

[15] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249–283, 2013.

[16] P. Donmez, J. G. Carbonell, and J. Schneider, "Efficiently Learning the Accuracy of Labeling Sources for Selective Sampling," in *Int. Conf. on Knowledge Discovery and Data Mining*, 2009, pp. 259–268.

[17] Y. Zheng, S. Scott, and K. Deng, "Active Learning from Multiple Noisy Labelers with Varied Costs," in *IEEE Int. Conf. on Data Mining*, 2010, pp. 639–648.

[18] S. J. Huang, J. L. Chen, X. Mu, and Z. H. Zhou, "Cost-effective Active Learning from Diverse Labelers," in *Int. Joint Conf. on Artificial Intelligence*, 2017, pp. 1879–1885.

[19] C. Beyer, G. Krempl, and V. Lemaire, "How to Select Information That Matters: A Comparative Study on Active Learning Strategies for Classification," in *Int. Conf. on Knowledge Technologies and Data-Driven Business*, 2015.

[20] G. Krempl, D. Kottke, and M. Spiliopoulou, "Probabilistic Active Learning: Towards Combining Versatility, Optimality and Efficiency," in *Int. Conf. on Discovery Science*, 2014, pp. 168–179.

[21] D. Kottke, M. Herde, C. Sandrock, D. Huseljic, G. Krempl, and B. Sick, "Toward Optimal Probabilistic Active Learning Using a Bayesian Approach," *arXiv:2006.01732*, 2020.

[22] F. Rodrigues, F. Pereira, and B. Ribeiro, "Gaussian Process Classification and Active Learning with Multiple Annotators," in *Int. Conf. on Machine Learning*, 2014, pp. 433–441.

[23] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[24] P. Donmez, J. Carbonell, and J. Schneider, "A Probabilistic Framework to Learn from Multiple Annotators with Time-Varying Accuracy," in *SIAM Int. Conf. on Data Mining*, 2010, pp. 826–837.

[25] A. Chaudhuri, D. Kakde, C. Sadek, L. Gonzalez, and S. Kong, "The Mean and Median Criteria for Kernel Bandwidth Selection for Support Vector Data Description," in *IEEE Int. Conf. on Data Mining Workshop*, 2017, pp. 842–849.

[26] D. Rey and M. Neuhäuser, "Wilcoxon-Signed-Rank Test," in *International Encyclopedia of Statistical Science*. Springer Berlin Heidelberg, 2011, pp. 1658–1659.

[27] A. Calma, J. M. Leimeister, P. Lukowicz, S. Oeste-Reiß, T. Reitmaier, A. Schmidt, B. Sick, G. Stumme, and K. A. Zweig, "From Active Learning to Dedicated Collaborative Interactive Learning," in *Int. Conf. on Architecture of Computing Systems*, 2016, pp. 1–8.

[28] C. Sandrock, M. Herde, A. Calma, D. Kottke, and B. Sick, "Combining Self-reported Confidences from Uncertain Annotators to Improve Label Quality," in *Int. Joint Conf. on Neural Networks*, 2019, pp. 1–8.

[29] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential Deep Learning to Quantify Classification Uncertainty," in *Neural Information Processing Systems 31*, 2018, pp. 3179–3189.