This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

Department: Head
Editor: Name, xxxx@email

Machine learning from crowds using candidate set-based labelling

Iker Beñaran-Muñoz
Basque Center for Applied Mathematics, Bilbao, Spain

Jerónimo Hernández-González
Departament de Matemàtiques i Informàtica, Universitat de Barcelona (UB), Barcelona, Spain

Aritz Pérez
Basque Center for Applied Mathematics, Bilbao, Spain

***Abstract*—Crowdsourcing is a popular cheap alternative in machine learning for gathering information from a set of annotators. Learning from crowd-labelled data involves dealing with its inherent uncertainty and inconsistencies. In the classical framework, each annotator provides a single label per example, which fails to capture the complete knowledge of annotators. We propose candidate labelling, that is, to allow annotators to provide a set of candidate labels for each example and thus express their doubts. We propose an appropriate model for the annotators, and present two novel learning methods that deal with the two basic steps (label aggregation and model learning) sequentially or jointly. Our empirical study shows the advantage of candidate labelling and the proposed methods with respect to the classical framework.**

Introduction

Recently, crowdsourcing has spread in machine learning for labelling data [1] as a cheap alternative to expert labelling. As the reliability of the contributors is unknown, several labels, which are usually inconsistent, are gathered per example to reduce the associated uncertainty.

Learning from crowds [2] aims to learn classifiers from crowd-labelled data. There are two basic tasks to solve: to estimate the ground truth labels by aggregating the inconsistent labels provided by the annotators, and to learn the model. These are approached either *sequentially* or *jointly*. Sequential approaches deal with the label uncertainty in the aggregation task. *Majority voting* [3] (the label chosen by most annotators is assigned to each instance) or *weighted voting* [4] (the choice of the annotators is weighted according to their reliability) are basic strategies to do so. Many methods, starting from [5], rely for aggregation on the Expectation-Maximisation (EM) strategy [6] due to its ability to deal with

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

missing data or uncertainty. It estimates both the reliability of the annotators and the ground truth labels, through an iterative procedure where one estimate helps compute the other. Nevertheless, EM-based methods can also follow the joint approach to directly learn a classifier [2].

Traditionally, each annotator provides a single label for each example, a scheme we call *full labelling*. However, a single class label might not capture the doubts of the annotators. For example, if two labels are equally plausible in the mind of an annotator and we force them to select a single one, we lose all the information about the not chosen class. Thus, our working hypothesis is that providing annotators with the flexibility to return all the labels among which they doubt allows to extract more information from the labellers. Previous attempts to soften full labelling rigidity include to provide a "don't know" option to the annotators [7], or ask them to return also how sure they are about their answer [8].

We propose to use *candidate labelling* as an alternative to full labelling, inspired by the partial labels (PL) problem [9]. PL is a weakly supervised problem where each instance is associated to a set of candidate labels, and it is assumed that the ground truth label is in that set. Analogously, in our proposal for crowd annotation, candidate labelling allows annotators to select more than one label for each example. We do not assume that annotators provide always the true label. But the flexibility of candidate labelling raises the probability that the true label is selected, help us to gather more information about annotator doubt, and optimise the label gathering process.

We also present an annotator reliability model for candidate labelling in crowds. We propose two novel learning procedures from this type of data, one that performs sequential learning (SL-C), and another one that performs joint learning (JL-C). They are inspired by the proposals of [5] and [2] respectively, and can be seen as their generalisation from the full labelling to the candidate labelling context.

The paper continues with the review of the related work and the background description. Then, we formulate the framework, we propose our annotator model, and present our learning methods. Next, we analyse an extensive empirical study. Finally, the main conclusions are summarised.

Related work

Many crowd learning methods are based on the EM strategy [6], which iteratively maximises the likelihood of the parameters of an annotator model that usually accounts for their reliability. The key idea is to realise that annotator reliability can be used to improve label aggregation, and, in turn, the aggregated labels can help us to measure annotator reliability. EM is guaranteed to converge to a local maximum through the iteration of two steps: (i) Expectation (E-step), where the expected value of the uncertain ground truth is computed using the current model fit, and (ii) Maximisation (M-step), new maximum likelihood estimates (MLE) of the model parameters are obtained given the previously completed data. The key work by Dawid and Skene [5] uses an EM-based method and models each annotator with a conditional probability distribution over the classes given the real label. It solves the label aggregation task, and classifier learning is left as a subsequent step (sequential scheme). We use this method as a baseline method to compare with, as it is the equivalent to our method SL-C in the full labelling context.

EM-based methods can also implement a joint learning scheme, where both the parameters of the annotator model and those of the classification model are estimated simultaneously. One of them is Raykar et al. [2]'s method, that in binary problems uses an annotator model with only two parameters per annotator which represent the probability that the annotator correctly labels instances of true class 1 (sensitivity) and instances of true class 0 (specificity). In multiclass problems, their annotator model becomes similar to that of Dawid-Skene [5]. This joint learning method is used in this work as a baseline, as it is the equivalent to our method JL-C in the full labelling framework. Sheng et al. [15] presented a set of methods for joint learning that weigh each instance and then use a cost-sensitive classifier to learn from the weighted examples. Rodrigues et al. [16] proposed an EM-based method for learning deep neural networks from crowds. They use of a crowd layer, so that the network can be trained directly from the crowdsourced labels using backpropagation.

All the approaches discussed so far in this

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

section consider full labelling. Preliminary results with the presented candidate labelling [10] show that it leads to enhanced performance with simple voting methods, especially in scenarios with high uncertainty. Besides, this flexibility can also lead to faster and less costly labelling [11].

In social sciences, *approval voting* [12], a labelling system similar to candidate labelling, has been extensively studied. Each user provides a set of labels per instance, but their goal is to aggregate the choices assuming that there exists no ground truth. They usually assign to each instance the label that is selected by most annotators. Thus, annotators who provide large sets have more influence on the outcome. In contrast, we assume that large sets indicate greater doubts about labelling, which usually implies that its impact is reduced.

Background

Learning from crowds deals with a supervised classification problem with data incompleteness: true labels are not provided. Several noisy labels are collected for each example from non-expert contributors to face uncertainty.

Formally, let us define the (multivariate) random variable X as the descriptive feature of the problem, taking values x in the space Ω_X. The class variable C takes values c from the set of labels $\Omega_C = \{1, \ldots, r\}$, where $r \geq 2$. We assume that the random vector (X, C) is distributed according to a probability distribution $p(X, C)$, and that each instance x is related to a single true class label c_x, i.e. $p(c_x|x) = 1$. In supervised classification, we aim to infer from a set of instance-label pairs $\{(x_i, c_i)\}_{i=1}^n$ a mapping (classifier) from Ω_X to Ω_C with a good performance in unseen instances.

In learning from crowds, the real class labels are not available, i.e., instances are given alone: $\mathcal{D} = \{x_i\}_{i=1}^n$. The only information of supervision available is provided by a set of annotators, A. In the standard framework (full labelling), each annotator $a \in A$ provides a label $l_x^a \in \Omega_C$ of their choice. Without loss of generality, we assume that every annotator $a \in A$ annotates every instance $x \in \mathcal{D}$. The labelling for instance x is the set of labels $\mathcal{L}_x = \{l_x^a\}_{a \in A}$, and $\mathcal{L} = \{\mathcal{L}_x\}_{x \in \mathcal{D}}$ is the set of crowdsourced labels for the whole training set. Given $\mathcal{D}$ and $\mathcal{L}$, the goal is the same as in supervised classification. A table summarising all the symbols used in the paper is available in a document with supplementary material at https://github.com/IK3R/EM-candidate-learning.

Framework: Crowd learning with candidate labelling

This work builds on top of *standard* learning from crowds assuming that better access and modelling of the uncertainty of the annotators can lead to enhanced performance.

The main novelty is that annotators $a \in A$ are allowed to provide a set of labels $L_x^a \subseteq \Omega_C$, called *candidate set*, for instances $x \in \mathcal{D}$ (we assume $|\Omega_C| > 2$). The candidate set L_x^a is expected to include any class that annotator a considers plausible. Thus, fine-grained information about their doubts is available. Instance x has associated multiple candidate sets $\mathcal{L}_x = \{L_x^a\}_{a \in A}$, and the whole labelling is $\mathcal{L} = \{\mathcal{L}_x\}_{x \in \mathcal{D}}$. Given $\mathcal{D}$ and $\mathcal{L}$, the goal remains that of supervised classification.

Probably the most basic approach to this learning problem would be, as discussed in our preliminary work [10], to aggregate candidate sets using the *candidate voting* estimate. It generates a probabilistic labelling proportional to the weighted sum of annotators that assign label c to instance x, with weights inversely proportional to their candidate set size, $|L_x^a|$:

$$w_x(c) = \frac{1}{|A|} \sum_{a \in A} \frac{\mathbb{1}(c \in L_x^a)}{|L_x^a|}. \tag{1}$$

We can apply a *winner-takes-all* strategy taking the label that maximises the candidate voting estimate to obtain a deterministic labelling:

$$\omega(\mathcal{L}_x) = arg\, max_c w_x(c), \tag{2}$$

which can be seen as a natural generalisation of majority voting [3] for candidate labelling. Indeed, it reduces to majority voting when all annotators provide a single label.

Learning a classifier from the (probabilistic) labelling given by Equation 1 or 2 is possible. However, these implicitly assume that annotators show homogeneous reliability, which is usually not the case. Modelling their reliability can imply an enhanced classification performance.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

A reliability-aware aggregation of candidate-set annotations

First of all, we present our annotator model that will allow us to aggregate the candidate sets considering annotator reliability.

Annotator model for candidate labelling

We assume that annotators produce candidate sets according to the following unspecified probability model over a set of labels: each annotator deals with an independent binary choice for each class label to decide whether to include it in the candidate set. This corresponds to a latent-scale model [13] that represents a probability distribution over sets of elements.

Formally, let $\alpha^a_{ck} \in [0,1]$ denote the probability that annotator a includes label k in the candidate set for an instance with true class label c. Then, the probability of the candidate set provided by annotator a for instance x, L^a_x, corresponds to:

$$Pr[L^a_x] = \prod_{k \in \Omega_C} (\alpha^a_{c_x k})^{\mathbb{1}(k \in L^a_x)} \cdot (1 - \alpha^a_{c_x k})^{(1-\mathbb{1}(k \in L^a_x))}$$

where Ω_C is the set of possible class labels. The annotator with complete knowledge would show $\alpha^a_{ck} = 1$ when $k = c$, and $\alpha^a_{ck} = 0$ in other cases. We denote by $\boldsymbol{\alpha} = \{\alpha^a_{ck} : c, k \in \Omega_C, a \in A\}$ the set of parameters for all the annotators. Take into account that, unlike the models of Dawid-Skene [5] or Raykar et al. [2], our α^a_{ck} parameters do not form conditional probability distributions given a fixed c ($\sum_k \alpha^a_{ck}$ is not necessarily 1). An annotator who usually includes k and k' when the real label is c could show $\alpha^a_{ck} = \alpha^a_{ck'} \approx 0.9$. This behaviour, derived from our use of candidate labelling, represents the main novelty of our model regarding fully labelling models (e.g., [2], [5]). $\boldsymbol{\alpha}$ parameters can be understood as annotator reliability: an annotator is reliable when the true class has the highest probability ($c = arg\,max_{k \in \Omega_C} \alpha_{c,k}$) and, on average, it tends to appear in the candidate set ($\alpha_{c,c} > 0.5$). We assume that annotators are conditionally independent given the true class label, $Pr[\{L^a_x, L^{a'}_x\}] = Pr[L^a_x] \cdot Pr[L^{a'}_x]$.

This model implicitly assumes that (i) the behaviour of an annotator only depends on the true class, $Pr[L^a_x] = Pr[L^a_{x'}]$ for $L^a_x = L^a_{x'}$ if $c_x = c_{x'}$, and (ii) for each annotator the probability of including two labels in the candidate set is conditionally independent given the actual class.

Now, we can define the likelihood given candidate set $\mathcal{L}_x$ for instance x of real class c:

$$Pr(\mathcal{L}_x | c_x = c, \boldsymbol{\alpha}) = \prod_{a \in A} \prod_{k \in \Omega_C} (\alpha^a_{ck})^{\mathbb{1}(k \in L^a_x)} \cdot (1 - \alpha^a_{ck})^{(1-\mathbb{1}(k \in L^a_x))}. \quad (3)$$

SL-C: An EM method for aggregation of candidate sets

We are interested in learning the annotator model parameters, $\boldsymbol{\alpha}$, and aggregating the candidate sets taking them into account. We do not need the descriptive information x to calculate the MLE of these parameters as we assume that the candidate sets only depend on the real class c_x.

The likelihood given a dataset $(\mathcal{D}, \mathcal{L})$ is:

$$Pr\left(\mathcal{L}; \boldsymbol{\alpha}\right) = \prod_{x \in \mathcal{D}} \sum_{c \in \Omega_C} (Pr(\mathcal{L}_x | c_x = c; \boldsymbol{\alpha}) \cdot Pr(c_x = c)). \quad (4)$$

As the real label c_x of each example x is assumed to be unique, $Pr(c_x|x) = 1$, the marginalisation of C can be re-expressed as a product raised to the indicator function, and combined with Eq. 3:

$$\begin{aligned} Pr\left(\mathcal{L}; \boldsymbol{\alpha}\right) &= \prod_{\substack{x \in \mathcal{D} \\ c \in \Omega_C}} \left[Pr(\mathcal{L}_x | c_x = c; \boldsymbol{\alpha})\right]^{\mathbb{1}(c_x = c)} \\ &= \prod_{\substack{x \in \mathcal{D} \\ c \in \Omega_C}} \Bigg[\prod_{\substack{a \in A \\ k \in \Omega_C}} (\alpha^a_{ck})^{\mathbb{1}(k \in L^a_x)} \cdot \\ &\quad \cdot (1 - \alpha^a_{ck})^{(1-\mathbb{1}(k \in L^a_x))} \Bigg]^{\mathbb{1}(c_x = c)}. \end{aligned} \quad (5)$$

Computing the derivative of the log-likelihood with respect to α^a_{ck}, and setting it to zero, we obtain the MLE of α^a_{ck}:

$$\hat{\alpha}^a_{ck} = \frac{\sum_{x \in \mathcal{D}} \mathbb{1}(c_x = c) \cdot \mathbb{1}(k \in L^a_x)}{\sum_{x \in \mathcal{D}} \mathbb{1}(c_x = c)}, \quad (6)$$

which is the proportion of instances x of real class $c_x = c$ for which annotator a included label k in their candidate set, $k \in L^a_x$.

To compute Equation 6, the real labels c_x are required, precisely the information which is missing in crowd learning. Alternatively, using the

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

Algorithm 1 Method *SL-C*

1: **procedure** EM-SL-C
2: $\hat{\boldsymbol{\alpha}} \leftarrow \boldsymbol{\alpha}^{(0)}$
3: **while** $\hat{\boldsymbol{\alpha}}$ not converged **do**
4: $q \leftarrow Pr\left(\boldsymbol{c}|\mathcal{L}; \hat{\boldsymbol{\alpha}}\right)$
5: $\hat{\boldsymbol{\alpha}} \leftarrow arg\,max_{\boldsymbol{\alpha}} \mathbb{E}_{\boldsymbol{c}\sim q} \log Pr(\mathcal{L}|\boldsymbol{c}; \boldsymbol{\alpha})$
6: **end while**
7: **return** $q, \hat{\boldsymbol{\alpha}}$
8: **end procedure**

Bayes rule and Eq. 3, we estimate the probability of the true class as:

$$\begin{aligned} & Pr\left(c_x = c|\mathcal{L}_x; \boldsymbol{\alpha}\right) \propto \\ & \propto Pr(c_x = c) \cdot Pr\left(\mathcal{L}_x|c_x = c; \boldsymbol{\alpha}\right) \\ & \propto Pr(c_x = c) \cdot \prod_{\substack{a\in A \\ k\in\Omega_C}} \left(\alpha_{ck}^a\right)^{\mathbb{1}(k\in L_x^a)} \cdot \\ & \cdot \left(1 - \alpha_{ck}^a\right)^{(1-\mathbb{1}(k\in L_x^a))}, \end{aligned} \tag{7}$$

where $Pr(c_x = c)$ is calculated as the relative frequency of label c. Note that the rest of the class labels ($k \in \Omega_C : k \neq c$) intervene through the use of the $\boldsymbol{\alpha}$ parameters: it accounts for the usual confusions of the annotators; i.e., the probability that an annotator introduces a wrong label k when the real one is c.

To compute Equation 7, the model parameters $\boldsymbol{\alpha}$ are required. Note the mutual requirements of Equations 6 and 7. This naturally leads to an EM method that iterates over two steps: (i) E-step, where the expected value of the ground truth labels is obtained with Equation 7 for every instance x (given the current $\hat{\boldsymbol{\alpha}}$ estimation), and (ii) M-step, where the annotator reliability parameters $\boldsymbol{\alpha}$ are updated with Equation 6 (given the ground truth estimations of the previous E-step). If we define the computation of E-Step as:

$$q_{\hat{\boldsymbol{\alpha}}}(c|x) = Pr\left(c_x = c|\mathcal{L}_x; \hat{\boldsymbol{\alpha}}\right),$$

these $q_{\hat{\boldsymbol{\alpha}}}(c|x)$ estimates can substitute the indicator function $\mathbb{1}(c_x = c)$ in Equation 6, accounting for all the possible values of c_x probabilistically, since the real class label is unknown.

Algorithm 1 describes our EM-based method named **SL-C**, which stands for sequential learning with candidate labelling. The complexity of the initialisation (line 2 in Algorithm 1) is $\mathcal{O}(nmr^2)$ with respect to the numbers of instances (n), annotators (m) and classes (r). For each iteration, the complexity of both the E-step (line 4 in Algorithm 1) and the M-step (line 5 in Algorithm 1) is also $\mathcal{O}(nmr^2)$. Thus, the overall complexity of each iteration is $\mathcal{O}(nmr^2)$.

The E and M-step are iteratively interleaved until convergence. In our implementation, convergence is reached when the difference between the $\hat{\alpha}_{ck}^a$ in two consecutive iterations falls below a threshold. The likelihood is enhanced in each EM iteration until a local maximum is reached at convergence [6]. Thus, our algorithm stops when the MLE $\hat{\boldsymbol{\alpha}}$ cannot be further improved. The result of SL-C is an estimate of the ground truth. To complete the goal of learning from crowds, a classifier can be learned using standard supervised classification techniques (sequential approach).

Reliability-aware joint aggregation and learning from candidate sets

In crowd learning, the final objective is to learn a classifier. The previous method, SL-C, only performs label aggregation, and classifier learning is performed in a subsequent step. Our second method includes the classifier learning step into the loop.

JL-C: An EM method for jointly aggregating candidate sets and learning

We assume the same annotator reliability model described above. Let us also consider a probabilistic classifier that models the probability that instance x belongs to class c, expressed as $h(c|x;\theta)$, where θ represents the classification model parameters.

The likelihood is now:

$$\begin{aligned} & Pr\left(\mathcal{D}, \mathcal{L}|\boldsymbol{\alpha}, \theta\right) = \\ & = \prod_{x\in\mathcal{D}} \sum_{c\in\Omega_C} Pr\left(\mathcal{L}_x|c_x = c, \boldsymbol{\alpha}\right) Pr\left(c_x = c|x, \theta\right) \\ & = \prod_{x\in\mathcal{D}} \sum_{c\in\Omega_C} Pr\left(\mathcal{L}_x|c_x = c, \boldsymbol{\alpha}\right) h(c|x;\theta) \\ & = \prod_{x\in\mathcal{D}} \sum_{c\in\Omega_C} \left[h(c|x;\theta) \prod_{\substack{a\in A \\ k\in\Omega_C}} \left(\alpha_{ck}^a\right)^{\mathbb{1}(k\in L_x^a)} \cdot \right. \\ & \left. \cdot \left(1 - \alpha_{ck}^a\right)^{(1-\mathbb{1}(k\in L_x^a))} \right]. \end{aligned} \tag{8}$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

Using the same trick as for Equation 5, we re-express Equation 8 as a product of factors to the power of the indicator function:

$$Pr\left(\mathcal{D},\mathcal{L}|\boldsymbol{\alpha},\theta\right)=\prod_{\substack{x\in\mathcal{D}\\c\in\Omega_C}}\Bigg[h(c|x;\theta)\cdot \cdot\prod_{\substack{a\in A\\k\in\Omega_C}}(\alpha_{ck}^{a})^{\mathbb{1}(k\in L_x^a)}(1-\alpha_{ck}^{a})^{(1-\mathbb{1}(k\in L_x^a))}\Bigg]^{\mathbb{1}(c_x=c)}, \tag{9}$$

From this, we derive the MLE $\hat{\alpha}_{ck}^{a}$, which turns out to have exactly the same expression of Equation 6. Thus, we face again the need of the real class labels c_x for estimating the $\boldsymbol{\alpha}$ parameters, and also to learn the classifier h. However, as aforementioned, this piece of information is missing in crowd learning. Thus, we resort again to an EM method to obtain the MLE for our model parameters $(\hat{\boldsymbol{\alpha}},\hat{\theta})$.

In the E-step, we estimate the probability of label c for instance x given the parameter estimates $\hat{\boldsymbol{\alpha}}$ and $\hat{\theta}$ as:

$$Pr(c_x=c|\mathcal{L}_x,x;\hat{\boldsymbol{\alpha}},\hat{\theta})= =\frac{Pr(\mathcal{L}_x|c_x=c;\hat{\boldsymbol{\alpha}},\hat{\theta})\cdot Pr(c_x=c|x;\hat{\boldsymbol{\alpha}},\hat{\theta})}{Pr(\mathcal{L}_x;\hat{\boldsymbol{\alpha}},\hat{\theta})}, \tag{10}$$

making use of the Bayes rule, where $Pr(c_x=c|x;\hat{\boldsymbol{\alpha}},\hat{\theta})=h(c|x;\theta)$ is given by the classifier. We denote:

$$q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)=Pr\left(c_x=c|\mathcal{L}_x,x;\hat{\boldsymbol{\alpha}},\hat{\theta}\right),$$

the probabilistic estimate of the ground truth of x, which is obtained as:

$$q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)\propto h(c|x;\hat{\theta})\prod_{\substack{a\in A\\k\in\Omega_C}}(\hat{\alpha}_{ck}^{a})^{\mathbb{1}(k\in L_x^a)}\cdot(1-\hat{\alpha}_{ck}^{a})^{(1-\mathbb{1}(k\in L_x^a))}, \tag{11}$$

where $\hat{\boldsymbol{\alpha}}$ and $\hat{\theta}$ are the parameter estimates found in the previous EM iteration. As before (in Eq. 7, for SL-C), the probability estimate $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)$ depends on all the labels other than c through the $\hat{\boldsymbol{\alpha}}$ parameters, and it is also proportional to the probability predicted by classifier h. Note that, in this case, x is taken into account through the classifier to soften the assumption of the annotator model that the behaviour of the annotators only depends on the real label.

Algorithm 2 Method *JL-C*

1: **procedure** EM-JL-C
2: $\hat{\theta},\hat{\boldsymbol{\alpha}}\leftarrow\theta^{(0)},\boldsymbol{\alpha}^{(0)}$
3: **while** $\hat{\boldsymbol{\alpha}}$ not converged **do**
4: $q\leftarrow Pr\left(\boldsymbol{c}|\mathcal{L},\mathcal{D};\hat{\boldsymbol{\alpha}},\hat{\theta}\right)$
5: $\hat{\theta}\leftarrow arg\,max_\theta\mathbb{E}_{\boldsymbol{c}\sim q}\log Pr(\boldsymbol{c}|\mathcal{D};\theta)$
6: $\hat{\boldsymbol{\alpha}}\leftarrow arg\,max_{\boldsymbol{\alpha}}\mathbb{E}_{\boldsymbol{c}\sim q}\log Pr(\mathcal{L}|\boldsymbol{c};\boldsymbol{\alpha})$
7: **end while**
8: **return** $q,\hat{\theta},\hat{\boldsymbol{\alpha}}$
9: **end procedure**

The M-step uses the distributions $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)$ to fit the model parameters $\hat{\boldsymbol{\alpha}}$ and $\hat{\theta}$. As before, since the real labels are missing, we find the $\hat{\boldsymbol{\alpha}}$ estimates that maximise the expectation $\mathbb{E}_{\boldsymbol{c}\sim q}\log Pr(\mathcal{L}|\boldsymbol{c};\boldsymbol{\alpha})$, which implies substituting the indicator functions in Eq. 6 with $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)$.

The $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)$ estimates are also used for learning the classification model parameters $\hat{\theta}$ using a training dataset with probabilistic labelling: the pair (x,c) has weight $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)$.

Algorithm 2 describes the method named **JL-C**, which stands for joint learning with candidate labelling. The computational complexity of the initialisation (line 2 in Algorithm 2) is $\mathcal{O}(f+nmr^2)$, where f represents the complexity of fitting the chosen classifier h. For each iteration, the complexity of the E-step (line 4 in Algorithm 2) is $\mathcal{O}(gnmr^2)$, where g represents the complexity of the prediction using the chosen classifier h. The complexity of the M-step (lines 5 and 6 in Algorithm 2) is $\mathcal{O}(f+nmr^2)$. Thus, the overall complexity of each iteration is $\mathcal{O}(f+gnmr^2)$.

The rest of details of the JL-C method are implemented in the same way as for SL-C.

Model selection and initialisation

As aforementioned, EM methods are guaranteed to converge to a local optimum of the likelihood that depends on the initialisation. To better explore the space of solutions, performing multiple runs with different initialisation is usually advised to try to reach different local maxima (and keeping the model with the highest likelihood).

For SL-C, we need to set initial values for

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

$\hat{\boldsymbol{\alpha}}$ (line 2 at Alg. 1), and for JL-C we need to fill in $\hat{\theta}$ too (line 2 at Alg. 2). We use random initialisations as follows: First, (i) obtain the candidate voting estimate as in Equation 1 for every $(x, c) \in (\mathcal{D}, \Omega_C)$. Then, (ii) sample the probability distributions provided by the candidate voting estimates for each $x \in \mathcal{D}$ to obtain initial deterministic guesses. In the last step (iii), we use these initial guesses as the ground truth to compute $\hat{\boldsymbol{\alpha}}$ using Eq. 6. For JL-C, we also use the initial guesses as labelling for training the classification model parameters $\hat{\theta}$.

Regarding the evaluation of the models obtained with different runs, we keep the model that maximises the expected log-likelihood. For SL-C, the expected log-likelihood is:

$$\begin{aligned} E_{q_{\hat{\boldsymbol{\alpha}}}}\left[\log\left(Pr\left(\mathcal{L}|\hat{\boldsymbol{\alpha}}\right)\right)\right] = \sum_{\substack{x\in\mathcal{D}\\ c\in\Omega_C}} q_{\hat{\boldsymbol{\alpha}}}(c|x)\cdot \\ \cdot \sum_{\substack{a\in A\\ k\in\Omega_C}} \Big(\mathbb{1}(k\in L_x^a)\cdot\log(\hat{\alpha}_{ck}^a) + \\ + (1-\mathbb{1}(k\in L_x^a))\cdot\log(1-\hat{\alpha}_{ck}^a)\Big), \end{aligned} \tag{12}$$

using the $q_{\hat{\boldsymbol{\alpha}}}(c|x)$ estimates computed in the last E-step.

Analogously, for JL-C, we use the corresponding $q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}$ estimates for calculating the expected log-likelihood:

$$\begin{aligned} &E_{q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}}\left[\log\left(Pr\left(\mathcal{D},\mathcal{L}|\hat{\boldsymbol{\alpha}},\hat{\theta}\right)\right)\right] \\ &= \sum_{\substack{x\in\mathcal{D}\\ c\in\Omega_C}} q_{\hat{\boldsymbol{\alpha}},\hat{\theta}}(c|x)\cdot\Big[\log\Big(h(c|x;\hat{\theta})\Big) \\ &+ \sum_{\substack{a\in A\\ k\in\Omega_C}} \Big(\mathbb{1}(k\in L_x^a)\log(\hat{\alpha}_{ck}^a) \\ &+ (1-\mathbb{1}(k\in L_x^a))\log(1-\hat{\alpha}_{ck}^a)\Big)\Big]. \end{aligned} \tag{13}$$

Experiments

We have carried out an empirical analysis of both presented methods: SL-C and JL-C. The main hypothesis of this work is that candidate labelling provides more information about the true classes than the classical full labelling, which can lead to classifiers with better performance. To check this hypothesis, we test the performance of our methods against that of Dawid-Skene [5] (DS), as the sequential approach analogous to SL-C in the full labelling context, and that of Raykar et al. [2] (RAY), as the joint learning method analogous to JL-C in full labelling.

Besides, the experiments are designed to analyse relative differences of behaviour between SL-C and JL-C, as a way to compare the sequential and joint learning approaches. Unfortunately, there are not real crowdsourced datasets that make use of candidate labelling, so we have resorted to generating synthetic data, which allows us to explore a wider range of experimental scenarios.

Synthetic label generation

Crowdsourced data is simulated departing from standard supervised data and weakly supervised data with partial labels. A general procedure that allows generating both full and candidate crowdsourced labels synthetically is used.

The synthetic label generation procedure for standard supervised datasets is as follows. We have a set of m annotators A, and each annotator $a \in A$ is simulated by means of a set of probability distributions with support in Ω_C, $\{g_a(\cdot|c)\}_{c\in\Omega_C}$. The annotators are sampled from a Dirichlet distribution with r hyperparameters, all equal to 1 except for the c-th one, which is equal to $\beta \geq 1$. This experimental parameter allows us to control the expertise of the annotator: the greater the β value is, the higher tends to be the probability of the c-th class. Note that when $\beta = 1$ annotators are generated such that, on average, their labels are uniformly selected, and thus they not provide useful information about the true class.

Given an instance $x \in \mathcal{D}$ with an associated class label $c_x \in \Omega_C$, the labelling is generated by sampling the probability distribution $g_a(\cdot|c_x)$. That probability distribution is sampled once to perform **full labelling** (annotators express no doubt), or $\lceil prop \cdot r \rceil$ **times with replacement** to perform **candidate labelling** (annotators provide multiple labels to express their doubts). In the case that $prop \leq 1/r$, there will be only one label in the candidate set and it will be equivalent to full labelling, and when $prop > 1/r$, the size of the candidate set is in $[1, \lceil prop \cdot r \rceil]$.

When we use weakly supervised data with partial labels, where each instance $x \in \mathcal{D}$ is associated with a partial label set $C_x \subseteq \Omega_C$, the

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

procedure is the same as above with a single exception. Instead of $g_a(\cdot|c_x)$, we sample the probability distribution $\overline{g_a}(\cdot|x) = \sum_{c\in\mathcal{C}_x}(g_a(\cdot|c)/|\mathcal{C}_x|$ to generate the labelling for instance $x \in \mathcal{D}$.

We would like to highlight that this generative model is more complex than the models underlying SL-C, JL-C, DS and RAY. None of them is in an advantageous position in the following experimental design with that respect.

Experimental design

We use 6 fully labelled datasets from UCI repository (http://archive.ics.uci.edu/ml): *Dermatology* $(366, 6)$, *Glass* $(214, 6)$, *Segment* $(2310, 7)$, *Svmguide4* $(612, 6)$, *Vehicle* $(846, 4)$, and *Vowel* $(990, 11)$, and 3 partially labelled datasets [14]: *Birdac* $(3718, 13)$, *Lost* $(1122, 14)$ and *MSRCv2* $(1758, 23)$, with numbers meaning number of instances n, and number of classes r. We simulate different numbers of annotators $m \in \{3, 5, 7, 9\}$, and different degrees of expertise for them $\beta \in \{1, 3, 5, 7\}$. For candidate labelling generation, the proportion of sampled labels takes values $prop \in \{0.1, 0.3, 0.5, 0.7\}$. We have used two classifiers from very different families from *sklearn 0.22.1* with default parameters: 5-Nearest Neighbour (5NN) and Random Forest (RF).

The models are evaluated using the area under the ROC curve (AUC). It is estimated using stratified 5-fold cross-validation, where the test sets are fully supervised.

Results

Figure 1 shows the impact of the expertise (β parameter) of the annotators on the performance of the methods.

As expected, all methods consistently show better performance as the expertise of the annotators increases. As the annotator expertise decreases and gets closer to $\beta = 3$ (the scenario that is expected to be closest to reality, as annotators are non-expert), the performance difference of our methods relative to that of RAY and DS tends to become larger. When $\beta = 1$, the AUC scores are always near 0.5 (virtually, random classifiers). This is coherent with the fact that, with $\beta = 1$, annotators provide random labels without any information about the true class. Note that this is not the usual case, but gives us a reference to compare with. Overall, JL-C and SL-C outperform RAY and DS. There are cases where RAY or DS are competitive regarding our methods, usually when there is little growth in the AUC score from $\beta = 3$ values on. This might be due to limited problem difficulty, as little information of supervision leads to the best performance that the specific classifier type can reach.

Figure 2 shows the impact of the number of annotators (m) on the performance of the methods. As expected, the performance tends to improve as the number of annotators increases. The steepest performance increases are most commonly observed between $m = 3$ and $m = 5$ (clearly with JL-C and 5NN), and less commonly between $m = 5$ and $m = 7$. Usually, the degree of improvement of RAY and DS as the number of annotators increases is smaller than that of SL-C and JL-C. With fewer annotators, a realistic setup, our methods have advantage in most scenarios, except for JL-C with 5NN.

Figure 3 shows the effect of the maximum candidate set size ($prop$) in the performance. Although RAY and DS are not affected by this parameter, they are included as a reference. Overall, as $prop$ increases the results get better, except in the *glass* dataset with RF classifier (Fig. 3b).

When $prop = 0.1$, most of the times annotators provide a single label for each instance, as in full labelling. In that case, our models for SL-C and JL-C become virtually equivalent to those of DS and RAY, respectively. Thus, the results of SL-C and DS are similar, as well as those of JL-C and RAY (random labelling generation might explain occasional small divergences). In general, as the value of $prop$ increase, the performance of our methods improves with respect to the baselines. These results indicate that annotators should be encouraged to provide candidate sets large enough to ensure that they contain the real class label. In some cases, providing too many labels ($prop = 0.7$) could also lead to poorer results, although they would still perform better than the baselines.

To assess significant differences for each data set and each parameter configuration, we have performed a two-sample t-test with $\alpha = 0.05$ to compare the four methods pairwise. SL-C significantly outperforms DS in 58.54% of the configurations, while the opposite never happens

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

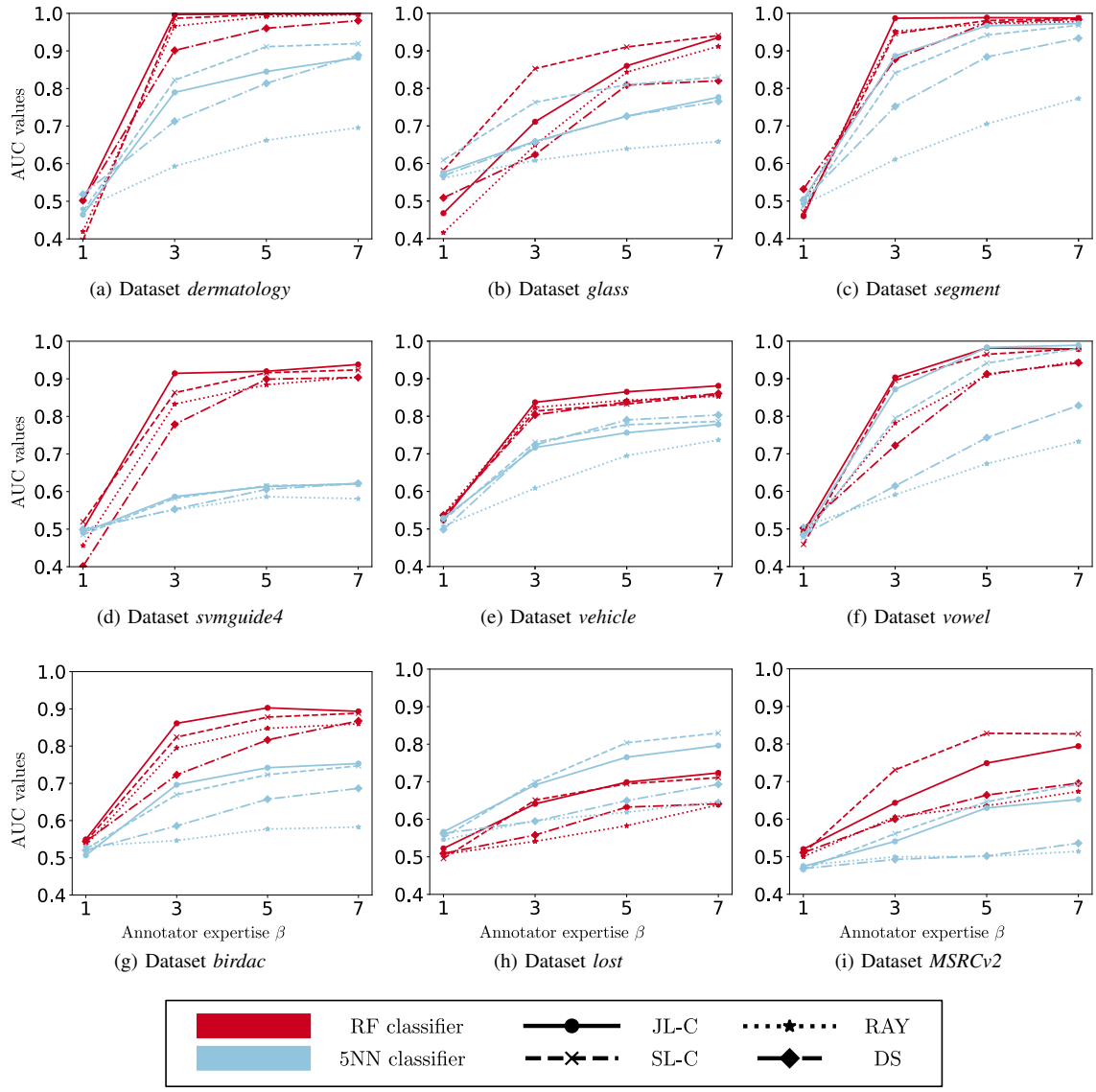

Figure 1. Experimental results throughout different values of the parameter β (annotator expertise), in terms of AUC metric, within different datasets (subplots). Results with classifiers RF and 5NN are displayed in dark blue and light blue colour, respectively. A different line style and marker is used for each method (SL-C, JL-C, RAY, DS). The rest of generative parameters are fixed to $m = 5$ and $prop = 0.5$.

(when comparing against RAY, SL-C has a better performance in 67.08% of the configurations, and the opposite occurs in only 0.83%). JL-C performs significantly better than RAY in 40% of configurations while RAY never obtains a significant advantage (it significantly outperforms DS in 50.41% of configurations, and the opposite never happens). JL-C outperforms SL-C in 17.45% of configurations, and SL-C outperforms JL-C in 20.57%.

Additional figures with alternative datasets and configurations are available in the supplementary material. Similar behaviours to those displayed here are observed.

To sum up, our methods (SL-C and JL-C) outperform the baselines (RAY and DS), in most of the configurations. Their performance is enhanced as the number of annotators and level of expertise are increased (differently depending on the classifier used). In general, by allowing annotators to

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

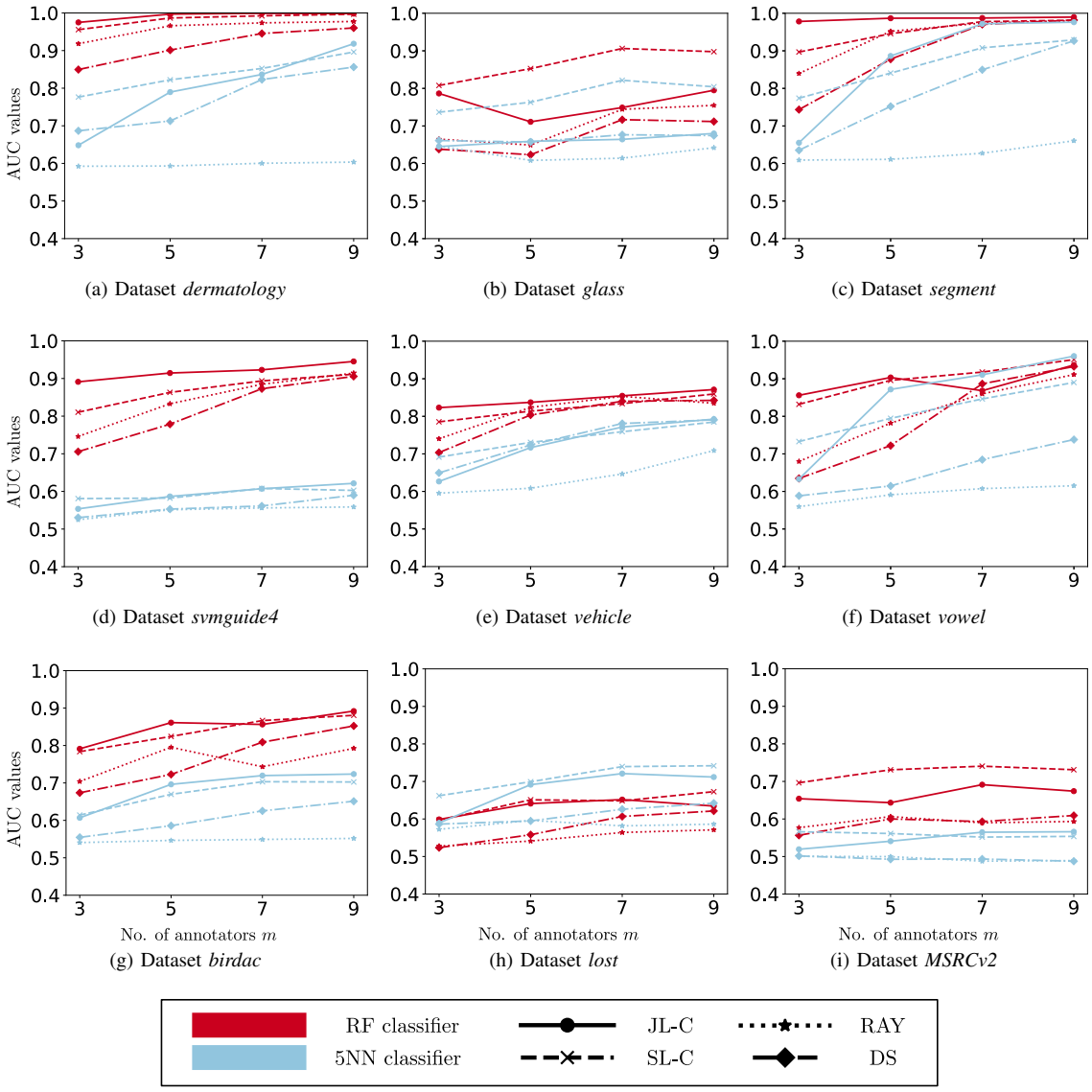

Figure 2. Experimental results throughout different values of the parameter m (number of annotators), in terms of AUC metric, within different datasets (subplots). Results with classifiers RF and 5NN are displayed in dark blue and light blue colour, respectively. A different line style and marker is used for each method (SL-C, JL-C, RAY, DS). The rest of generative parameters are fixed to $\beta = 3$ and $prop = 0.5$.

provide more classes ($prop$), both SL-C and JL-C show a performance improvement. Between SL-C and JL-C, it seems they mutually outperform each other depending heavily on the classifier and the dataset, with virtually no preference among them.

Discussion

Based on this empirical study, we can put forward several ideas.

Candidate labels seem to gather more discriminative information than the classic full labelling: SL-C and JL-C outperform RAY and DS in a vast majority of experimental scenarios. Using candidate labels we can produce classifiers with at least equal performance than using full labelling, with fewer annotators or with lower-expertise annotators. Using candidate labelling would be a way to further reduce the cost of labelling data.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

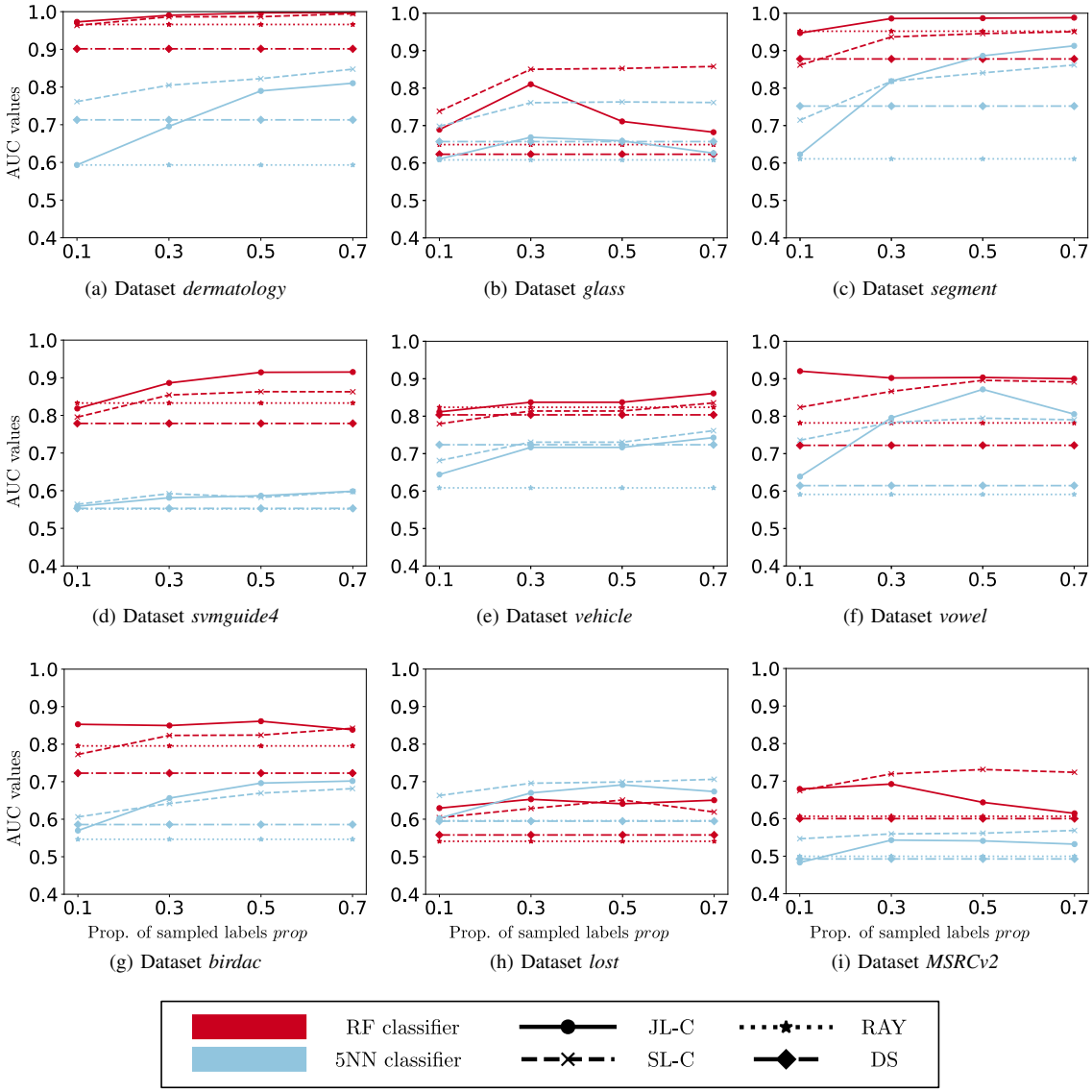

Figure 3. Experimental results throughout different values of the parameter $prop$ (flexibility of the annotators), in terms of AUC metric, within different datasets (subplots). Results with classifiers RF and 5NN are displayed in dark blue and light blue colour, respectively. A different line style and marker is used for each method (SL-C, JL-C, RAY, DS). The rest of generative parameters are fixed to $\beta = 3$ and $m = 5$.

The ability to express doubts about the labelling provides extra information about the true class. The two presented methods consistently improve with annotators that on average provide a larger number of labels (Figure 3). This evidence should at first motivate practitioners to allow annotators to provide sets of labels and encourage them to be as flexible as needed. A fair instruction would be to indicate to annotators that including the correct answer in the candidate set is preferred rather than filtering incorrect answers out. However, we need to be careful with these instructions: a set with too many labels might become uninformative and reduce the performance of the methods.

Other features used in the design of the empirical study do not show any light on the comparison between methods or labelling approaches. Sometimes, the same performance is reached by our candidate labelling-based methods

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

with smaller features values. However, all of them show the already-known trend of enhancement.

The computational complexity of our methods is similar to that of the baselines they were inspired by. We tested the **scalability** of our methods and the results suggest that, with an increasing number of instances, the running time of SL-C is always similar to that of the employed classifier, meanwhile for JL-C it seems to grow exponentially when employing RF and linearly in the case of 5NN. The variable that affects most the running time seems to be the number of classes, while increasing the number of annotators seems to cause just a small increase. The figures that graphically summarise the scalability test are available in the supplementary material.

Sequential or joint. Arguably any crowd learning method could be categorised as: (i) methods that first estimate the ground truth and then use standard machine learning to learn from it, and (ii) methods that learn a model as the ground truth labels are estimated. We presented, for candidate labelling, a method from each category. Our empirical study does not show relevant performance differences between them. This suggests that practitioners should test both approaches and empirically select the most appropriate one for their problems.

Finally, the **annotator model** for candidate labelling is one of our contributions. Both proposed methods use it, and their enhanced performance regarding that of DS and RAY validates it. Nevertheless, these methods could be easily adapted to work with other annotator models. Similarly, this empirical study is influenced by the type of classifier learned (5NN and RF). Nevertheless, our methods are completely abstracted from the classifier type and could work with any probabilistic classifier.

Conclusions

In crowd learning, our proposal to allow annotators to provide sets of candidate labels for each instance instead of a single label facilitates extracting more discriminative information from the crowd. We propose an annotator model and two methods which can be seen as extensions of two state-of-the-art works to the candidate labelling framework. They learn classifiers more robustly (or at least equal) than state-of-the-art methods [2], [5] using fewer annotators and/or lower quality annotators. This might involve a cost reduction of the labelling process.

It remains an open question whether sequential or joint learning approaches can be proved to be consistently better, or at least to describe the experimental scenarios where one of them is preferred. An answer to this question would be useful for practitioners. The promising experimental results support the development of labelling platforms that allow annotators to provide more than a label per instance. Practitioners could benefit from substantial savings in the cost of annotating large databases.

Acknowledgment

This work is partially supported by Spanish Ministry of Science, Innovation and Universities through BCAM Severo Ochoa accreditation SEV-2017-0718; by Basque Government through BERC 2022-2025 and ELKARTEK programs. IBM held a grant no. BES-2016-078095. JHG is a Serra Húnter fellow. We thank Jesús Cerquides (IIIA-CSIC) for his helpful discussion.

REFERENCES

1. 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.
2. V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, "Learning from crowds," *J. Mach. Learn. Res.*, vol. 11, pp. 1297–1322, 2010.
3. B. Parhami, "Voting algorithms," *IEEE Trans. Reliab.*, vol. 43, no. 4, pp. 617–629, 1994.
4. R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng, "Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks," in *EMNLP*, 2008, pp. 254–263.
5. A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *J. R. Stat. Soc. Ser. C*, vol. 28, no. 1, pp. 20–28, 1979.
6. A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Stat. Soc. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.
7. M. Torre, S. Nakayama, T. J. Tolbert, and M. Porfiri, "Producing knowledge by admitting ignorance: Enhancing data quality through an "I don't know" option in citizen science," *PloS one*, vol. 14, no. 2, 2019.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Intelligent Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MIS.2022.3205053

8. P. Smyth, U. M. Fayyad, M. C. Burl, P. Perona, and P. Baldi, "Inferring ground truth from subjective labelling of venus images," in *NeurIPS*, 1994, pp. 1085–1092.
9. T. Cour, B. Sapp, and B. Taskar, "Learning from Partial Labels," *J. Mach. Learn. Res.*, vol. 12, pp. 1501–1536, 2011.
10. I. Benaran-Munoz, J. Hernández-González, and A. Pérez, "Crowd learning with candidate labelling: An EM-based solution," in *CAEPIA*, 2018, pp. 13–23.
11. S. O. A. Banerjee and D. Gurari, "Let's agree to disagree: A meta-analysis of disagreement among crowdworkers during visual question answering," in *GroupSight at HCOMP*, 2017.
12. S. J. Brams and P. C. Fishburn, "Approval voting," *Am. Polit. Sci. Rev.*, vol. 72, no. 3, pp. 831–847, 1978.
13. A. Marley, "Aggregation theorems and the combination of probabilistic rank orders," in *Probability models and statistical analyses for ranking data*, 1993, pp. 216–240.
14. L.-P. Liu and T. G. Dietterich, "A conditional multinomial mixture model for superset label learning," in *NeurIPS*, 2012, pp. 557–565.
15. V. S. Sheng, "Simple multiple noisy label utilization strategies," in *ICDM*, 2011, pp. 635-644.
16. F. Rodrigues, and F. Pereira. "Deep learning from crowds," in *AAAI* 2018, pp. 1611-1618.

Iker Beñaran-Muñoz is currently a Ph.D. Student at University of the Basque Country (UPV/EHU). His research interests include learning form crowdsourced data. He finished a Master's Degree in Computational Engineering and Intelligent Systems at the UPV/EHU in September 2016. Contact him at ibm1993@hotmail.com.

Jerónimo Hernández-González is currently a Serra Húnter (tenure-eligible) lecturer in the department of Mathematics and Computer Science at University of Barcelona, Spain. His major research interests include weak supervision, learning and inference with probabilistic graphical models, and their applications to biomedical and educational domains. He received his Ph.D.6 in computer science from the University of the Basque Country, Spain, in 2015. Contact him at jeronimo.hernandez@ub.edu.

Aritz Pérez is currently a postdoctoral researcher at the Basque Center for Applied Mathematics. His main research lines include supervised, unsupervised and weak classification, probabilistic graphical models, and time series data mining, with applications to industry, energy management and health care. He received his Ph.D. degree in 2010 from the University of Basque Country. Contact him at aperez@bcamath.org.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/