Crowd-Powered Hybrid Classification Services: Calibration is all you need

Burcu Sayin, Evgeny Krivosheev, Jorge Ramírez, Fabio Casati
University of Trento
Trento, Italy

Ekaterina Taran, Veronika Malanina
TPU
Tomsk, Russia

Jie Yang
TU Delft
Delft, The Netherlands

***Abstract*—Hybrid classification services are online services that combine machine learning (ML) and humans - either crowd workers or experts - to achieve a classification objective, from relatively simple ones such as deriving the sentiment of a text to more complex ones such as medical diagnoses. This paper takes the first steps toward a science for hybrid classification services, discussing key concepts, challenges, and architectures, and then focusing on a central aspect, that of ML calibration and how it can be achieved with crowdsourced labels.**

1. Introduction and Motivations

Hybrid classification refers to the process of solving classification problems by leveraging both humans and ML [1]. Hybrid classification services have received little attention in the literature but, in our experience are by far the most common form of classification, especially in enterprise context. Almost invariably, when ML is adopted in the enterprise as well as in domains from medical to automotive, ML attempts an inference and a decision on whether to accept the inference or ask a human ensues, based on the *prediction confidence*.

Model calibration refers to the process of adjusting model parameters so that the prediction confidence is an accurate estimation of the probability of the prediction to be correct [2]. Calibration is extremely important in all cases where failures are costly and a fallback option to delegate decisions to human exists, from self-driving cars to automated medical diagnosis and even in work automation. For example, in enterprise workflow companies, a well-calibrated model is central to the adoption of AI as it gives customers the peace of mind to know that they have a model that knows when it doesn't know. Thus, the platform can decide, for each inference, when to trust it (because it has high confidence) and when instead to route the decision problem to a person. More generally, we argue that any time asking humans is an option, that possibility needs to be factored in as a first-class citizen and design variable, when training and using ML services, along with its cost (effort, money) and benefits (possibly, a more accurate classification, especially in cases where ML is undecided).

This paper proposes a *crowd-powered hybrid classification service*. Our goal is to take a step in the direction of developing a science for services that, automatically or semi-automatically, combine ML, crowd and experts in a cost-efficient and effective way to solve batch (finite item pool) or online (infinite pool) classification problems.

Specifically, the classes of problems we tackle are those with the following characteristics[1]:

- We have the option of asking humans for each classification, though at a cost. "Humans" can be experts and/or can be "crowd worker", accessed through a crowdsourcing platform, such as Mechanical Turk or Toloka.
- We have access to ML classification services, at a cost per classification that we assume negligible with respect to the cost of asking humans.
- The cost of asking humans is significantly less than the "cost" of either a false positive or a false negative.

In this very common scenario, given a budget, a crowd of labelers available at a price, a cost function for errors, and a finite or infinite pool of items to classify, we aim at defining a service and a policy to efficiently classify items. Again for ease of expositions (especially for where ML and validation are involved), we focus on tasks that require understanding of text documents, but the concepts are generally applicable.

In this paper, we show that such a service has a few key characteristics.

1) The service must combine the crowdsourcing aspects (with the well known challenges around it) and the efficient use of ML. As we will see there are several ways to do this, and in general the decision on how to process each item to classify depends on many factors, such as the quality of ML for that item, the expected accuracy and cost of the crowd, and the error cost structure [3]. What is common to different strategies is that the ML inference on an item, coupled with the estimated crowd accuracy in the task, may dynamically determine if we ask for crowd votes on an item, and how many (redundancy). None of this is supported by crowd platforms today.

1. For simplicity of exposition we limit here to binary classification

2) At the start of a task, we neither know how well ML performs on that task, nor how well crowd workers perform. We also do not know how "fast" (how cheaply, how many labeled examples) ML can learn, something that is particularly important in finite pool context, where there is a trade-off between what the best strategy (ordering) for selecting items to label from the pool if our goal is to train a model versus if our goal is to classify as many items as possible given a fixed budget. The ability to train a model efficiently as part of hybrid classification services, be it via the recent "few shots learning" approaches or the more "traditional" active learning.
3) The key metric for the ML service component is not the accuracy but calibration [4]. Calibration has been largely neglected in ML literature compared to accuracy, but in hybrid classification where the cost for one type of error is high (and also high with respect to the cost of asking humans), knowing if we can trust the ML service is key, just like it is important that experts tell us, whenever they make a statement, if they are sure or or not. As a consequence, the ability to *estimate* what crowd and ML can do and which kind of items they can or cannot classify accurately becomes central to the effectiveness of the service.

In this paper, we present the concepts and architecture of an hybrid classification service and then dig deeper on understanding how the ML service component, when used in a crowdsourcing context, can be trained to achieve model calibration and reduce the calibration error.

2. Problem Statement, Approach and Service Architecture

2.1. Problem Statement

We formulate the problem as follows. Given a set I of items to classify[2], a cost function λ (that includes cost of false positives, false negatives, and cost of asking humans), an ML algorithm m and a crowd of human labelers $h \in HL$, we aim at devising a hybrid classification policy (and a supporting system architecture) that classify the items with minimal loss.

While the problem statement is general, the interesting cases - which corresponds to many real-life situations - are those where the cost of machine classification is low, the cost of errors (at least one among false positive and false negative) is high or very high, and the cost of asking humans sits in between. In other words, if C_h is the cost for a human label, C_m is the cost for a machine inference, C_{fp} is the cost for a false positive and C_{fn} the cost for a false negative, we expect that:

$$C_m << C_h << max(C_{fp}, C_{fn}) \quad (1)$$

2. Again, for simplicity we assume binary classification but the concepts are generally applicable

And in fact in most cases it is safe to assume that C_m is very small so that for practical purposes we can consider it negligible. Assuming that to classify an item i we ask for n crowd votes on that item, then our loss function is:

$$\lambda = C_m + n \cdot C_h + \begin{cases} 0 & correct\ classification \\ C_{fp} & false\ positive \\ C_{fn} & false\ negative \end{cases} \quad (2)$$

Notice that this is a more general form of the problem we stated in the introduction, where we do not assume that humans are perfect oracles, which is why we may need multiple opinions. Instead, for simplicity, we assume all persons have the same costs, while in real settings different levels of competences may command different prices.

Given the loss function above, then if we are able to compute, for each item i, the probability $P(i \in pos)$ of an item belonging to the positive class, then we can estimate the expected loss $E[\lambda]$ if we classify the item as positive (and similarly do the same if we classify the item as negative). This is simply the cost independent of our classification decision ($C_m + n \cdot C_h$, which is not a random variable) plus the loss due to a possible classification error $E[\lambda_e]$ which is $C_{fp} \cdot (1 - P(i \in pos))$. For each item, we then choose the classification that minimizes such expected loss. As $P(i \in pos)$ gets close to either 0 or 1, the part of the expected loss due to classification errors gets closer to zero. In a hybrid classification service, the way we get such probability closer to the extremes is to either train "better" models or ask for more crowd votes.

As an example, consider Figure 1, that shows (center of the figure) a crowd task aiming at classifying if papers satisfy a search criterion - and in fact, in this case, if they satisfy all of a set of search criteria. Here we can assume a cost for asking a scientist or set of crowd workers to do the filtering, a cost for a false positive (including the paper in our search results) or false negative, that is, missing the paper while it is, instead, relevant.

2.2. Approach and Architecture

Figure 2 shows how we envision a hybrid classification service. A crowdsourcing platform (CP) provides basic access to users and manages payment. There are several such platforms[3], with fairly similar features, and we do not discuss them further here. While in most cases the CP is also responsible for "serving" the specific items to label to users (workers), here we need to take control and override that logic because we want hybrid classification to come into play. Specifically, we want to be able to decide for which items we ask crowd labels for, and to how many persons. We also want to dynamically stop asking if the combination of ML and the votes obtained so far gives us enough confidence to take a decision. Therefore, the items are loaded externally to the CP, into an item pool (on the right in the figure).

3. E.g., https://www.mturk.com or https://toloka.ai/

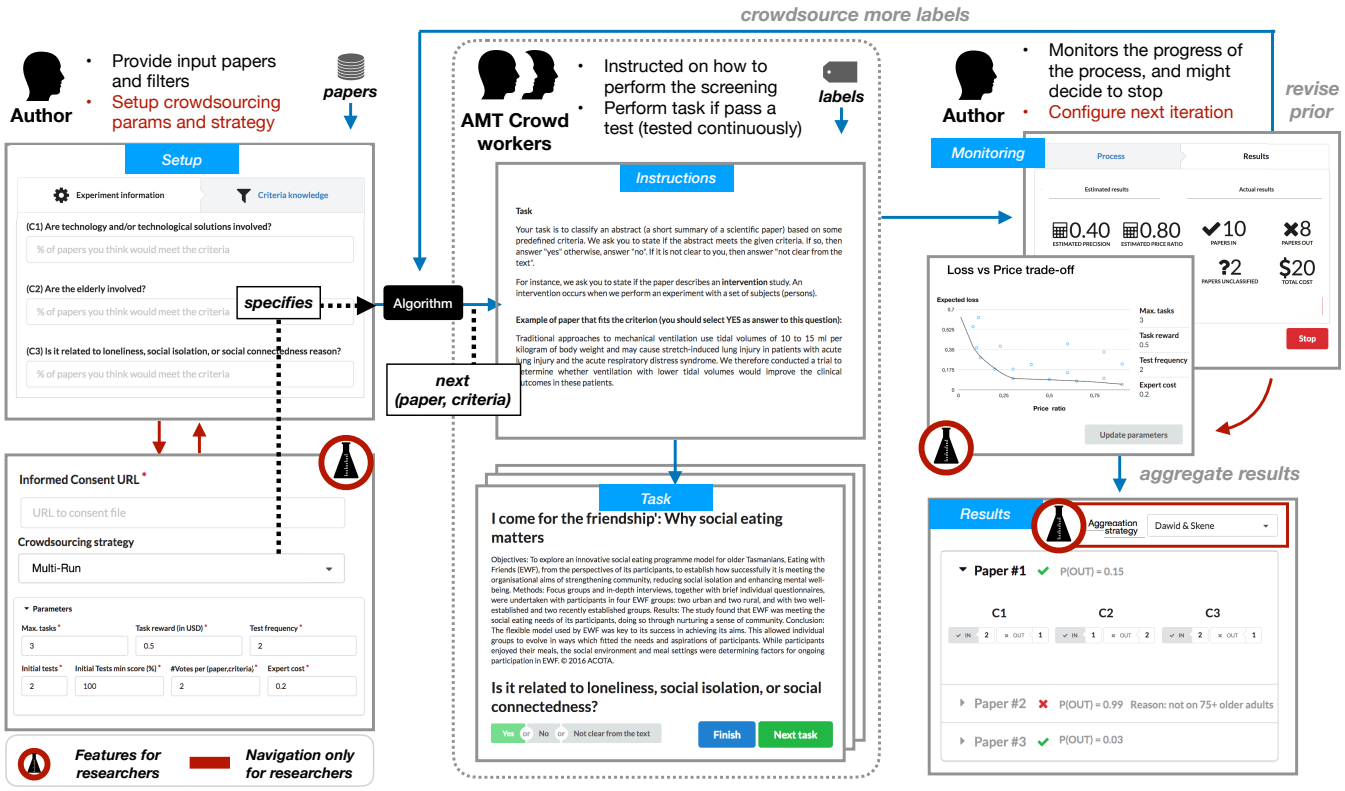

Figure 1: Example task workflow, modeled for a crowdsourced systematic literature review task. On the left column, the task designer writes the classification questions they want workers to answer, and point to the informed consent URL, along with other crowd task parameters. The center column shows what the worker would see in a task and the instructions given to them. The right side shows cost estimates that a platform can provide to the task designers after an initial crowdsourcing run. Details are provided in the technical report [5]

.

Processing proceeds in batch (also because we typically need to show batches of items to workers who label them in quick succession - making crowd workers wait is not really a viable option). The optimal batch can be selected via an "active learning" component which is peculiar in this case because we have two sometimes conflicting optimization objectives, that of asking votes with the goal to classify items in the pool vs asking votes based on what is best for training an ML model that can then help us. This is somewhat more nuanced than traditional active learning approaches and we discuss details in the report [3].

Once we do have a batch of votes, we then classify items, through a combination of humans and ML. As widely done in crowdsourcing, we can leverage the many existing techniques that compute workers' accuracy and aggregate votes on an item, using a variation of Expectation Maximization algorithm, such as DawidSkene [6]. However, the approach we take is to consider the classifier as a "voter", much like human workers, that is, characterized by its own "accuracy", where the accuracy (unlike workers) varies per item and corresponds to the prediction confidence on that item. Once we have the votes and accuracy we predict the class probability for each item, by first computing the probability that an item is positive (or negative) via simple application of Bayes rule:

$$P_{crowd}(i \in pos|V_i) = \frac{P(V_i|i \in pos) \cdot P(i \in pos)}{P(V_i)} \quad (3)$$

Where V_i is the set of labels given by crowd workers on item i. Given the workers accuracy (and therefore the probability that each worker is correct given that the item is positive or negative), these quantities can be easily computed as shown for example in equation 4 of [7]. The same paper also estimates how probability changes if we ask for more crowd labels on i. A hybrid classification service also has the benefit of machine prediction confidence (or, analogously, of having the classification service return the probability of the item being positive). We denote this with $P_{ml}(i \in pos)$. At this point, the probability of interest $P(i \in pos)$ can be computed as the average of P_{crowd} and P_{ml}[4]

Notice that in some cases the classifier may be so confident that its vote is sufficient and we do not need to resort to any crowd vote. Notice also that while workers'

4. Possibly weighted, though this requires determining the appropriate weights which is part of our future work.

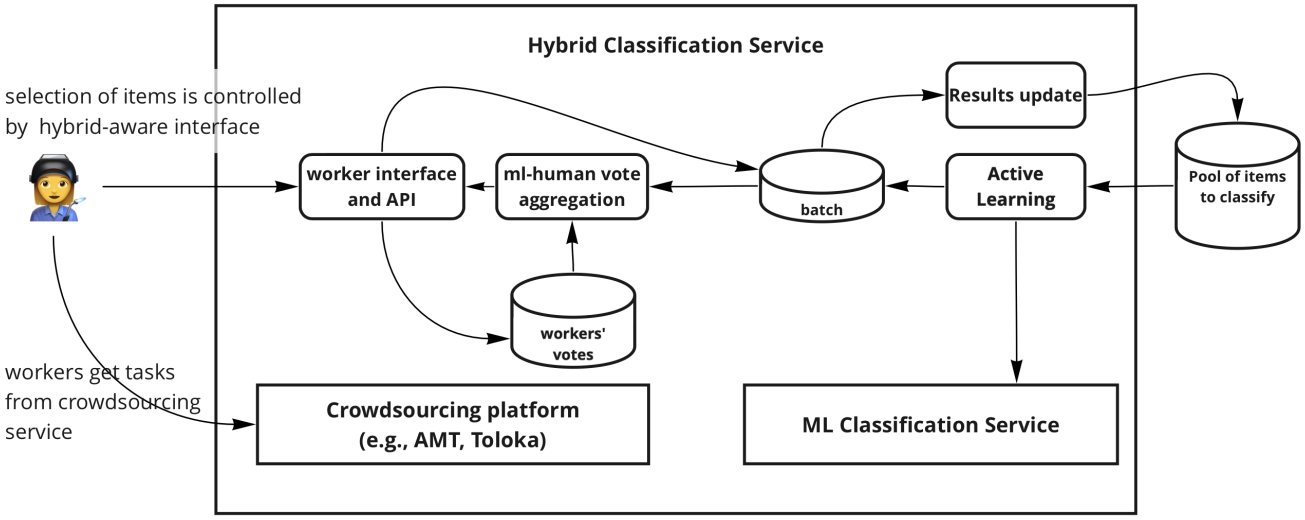

Figure 2: Hybrid classification service.

accuracy is independent of the items (we do not assume the crowdsourcing tasks ask for probabilities, as that is not commonly done and it would raise a whole set of challenges), ML classifiers typically do output a per-item probability and we make use of that here.

Figure 1 shows an example task workflow. This example shows a crowd task that performs a systematic literature review and where the strategy for selecting items to offer crowd workers is part of the task definition and controlled by the hybrid service rather than by the CP, because whether we need more labels for an item depends on whether the confidence we have thus far based on ML and current votes and on the consequent expected loss. The interface also shows a component that aims at estimating the cost for classifying items in the pool.

As it is now apparent, the correct estimation of the classifier confidence for each prediction is central to minimizing classification loss. Doing so requires going into the how ML algorithms can train well-calibrated classifiers in the presence of crowdsourced label. What is interesting - and difficult - here is that in such a service we have a complex interplay between the information we obtain for crowd (which is the only source of "truth", although imperfect - we do not assume crowds are oracles), the training of the ML model, and the calibration error. This interplay in hybrid classification services is the focus of the reminder of the paper. We analyze and discuss this topic in the next sections.

3. A Recap of Calibration

Hybrid classification is progressively gaining traction with more and more papers now starting to focus on combining human and ML inputs in classification problems. Initial work focused on very interesting ways to do this, from learning crowd vote aggregation models from "features" of the crowd task [8], to leveraging crowd to learn features of ML models, as in the brilliant paper by Bernstein and colleagues as well as others [9], [10]. In other works, automation is used to support crowd in a variety of task-specific ways. For example, Lasecki and colleagues show an effective approach to support the crowd in speech captioning by determining speech segment lengths optimal for each worker and merging partial input of each worker [11]. These efforts are complementary to our work since we do not aim at finding features or assisting workers in performing a task. Both tasks and ML algorithms are black boxes.

More recently, proposals have emerged based on training an ML model for a task and then first using that model to classify, then ask humans if that model's confidence is not high enough. For example, Callaghan et al [12] combine ML and crowd by automatically labeling items for which the ML confidence is above a defined threshold, and by polling the crowd for the remaining ones. Variations of this approach are applied in various fields, even in fashion [5].

This belief informs their crowd classification strategy by progressively adjusting the number of votes requested on each item based on whether the crowd confirms or negates such belief. No prior assumption is made on ML classifiers accuracy, and the hybrid algorithms are designed to be robust to weak classifiers. Finally, Nguyen et al. [13] leverage ML to identify which items to ask votes for, and who to ask to (experts or crowd). Each time their hybrid algorithm needs to pick up an item to classify and a type of human classifier (crowd or expert), it computes the *value* of each (item, classifier type) alternative by estimating the reduction in overall classification loss due to the new votes obtained divided by the cost of obtaining that loss reduction (experts are more expensive). The reduction in loss is due to two reasons: items become classified by crowd or expert (both assumed to be more precise than ML, which is not used to take classification decisions once expert or crowd

5. https://multithreaded.stitchfix.com/blog/2016/03/29/hcomp1

opinions are provided), and the additional information is used to train ML which in turn should ideally reduce the uncertainty and error on the items yet to be classified.

From this prior work we borrow the general idea of hybrid classifiers and the specific mechanisms to use ML output in crowd classification, but our contribution lies in how to leverage the per-item prediction confidence within a service and on how to support model calibration in crowdsourcing contexts.

To tackle model calibration, [14] propose to adopt *label smoothing*, where "hard" (1-0) class labels used in cross-entropy loss are smoothed into a probability distribution across classes. Such an approach has shown to be effective [2], [15], [16], [17], particularly for NLP, speech, and vision tasks [18], [19], [20], [21]. However, label smoothing and its effectiveness is still to be studied and understood, since probability amortization in output targets can bring extra noises [22] and prior art does not provide insights in how to set label smoothing hyper-parameters.

The remainder of this paper studies the effect of label smoothing on model calibration in NLP tasks when training labels are crowdsourced. The typical approach to deal with crowdsourced data is to aggregate votes for each item into a "gold" label, often using majority voting or expectation maximization methods [23], [24]. An alternative approach is that of creating an empirical distribution over crowd votes. This has been referred to as the *soft target* approach, [25], [26] and has shown to be effective in improving model performance in image classification [27] and emotion recognition from audio [28], [29].

While existing work has found that model performance is sensitive to noise in soft targets [30], none has investigated the effect of label fusion methods (widely used in crowdsourcing to aggregate human answers into one label, e.g., D&S [31] and GLAD [32]) on model training.

4. Label Smoothing

The crowdsourcing-label fusion-training pipeline is a commonly used pipeline and addressing the impact on calibration is therefore crucial to being able to generate "trustworthy" models. Specifically, in this work, we propose soft target methods that can incorporate any label fusion method: we present label fusion methods that accept raw crowd votes and output label probability distributions for every sample in the dataset, and where these label distributions are then used as soft targets to train well-calibrated neural models.

We evaluate our approach on 13 crowdsourced datasets and evaluate the effect of both soft targets and label smoothing in training the multi-layer perceptron and DistilBERT, a deep transformer model [33]. Our results show that soft targets are more effective for model calibration than label smoothing. We further demonstrate that our proposed soft target methods substantially improve both model performance and probability calibration across datasets of different noise levels, and this improvement is more obvious when they are used to train the deep transformer model.

We consider multi-class text classification task with L classes $\{1, 2, .., L\}$ where a classifier predicts $p(l|x)$, the probability that document x belongs to class l. We assume that, as commonly done, training aims at minimizing cross-entropy loss $loss = -\sum_l^L log(p(l|x)) \cdot \pi(l|x)$, where $\pi(l|x)$ here takes 1 if l is equal to the true label l^* and 0 otherwise, i. e., $\pi(l|x)$ here is the *hard target*.

In deep learning, label smoothing can be considered as a regularization technique for preventing the model from overfitting and from becoming overconfident in the classification decisions [14]. Thus, the ground-truth label distribution can be smoothed by adjusting $\pi(l|x)$ as follows:

$$\pi_{ls}(l|x) = \begin{cases} (1-\alpha) + \frac{\alpha}{L}, & l = l^*, \\ \frac{\alpha}{L}, & l \neq l^*; \end{cases} \tag{4}$$

where $\alpha \in [0, 1]$ is a hyper-parameter determining the amount of smoothing.

Soft Targets. We now introduce our approaches for soft targets leveraging crowd labels. When a requester crowdsources a dataset, it is common to collect several crowd labels per sample that allow us to infer the probability distribution over classes. For each pair (x, l), we can compute probabilistic confidence of sample x has ground-truth label l using a label fusion technique $\mathcal{F}$ that aims to map crowd votes into a class decision or a probability distribution π_f across classes, typically based on trying to estimate workers' accuracy A and factor it in while aggregating crowd votes. Formally, $\mathcal{F}$ is a function that takes crowd labels as input, and outputs a set of workers' accuracies A and a probability assignment π_f over the classes for every sample:

$$\mathcal{F}: \ \textit{crowd labels} \rightarrow \langle \pi_f, A \rangle, \tag{5}$$

Common examples of fusion methods are, e.,g., GLAD [32], or D&S [31]. By doing so, we can create *soft targets* according to Eq. 5 and incorporate them in the loss function. Note that now $\pi_f(l|x)$ comes from the 'natural' distribution of crowd-contributed labels as well as fusion methods, rather than from arbitrary smoothing with preselected hyper-parameters as in label smoothing methods. Furthermore, *each data point is smoothed differently, according to the ambiguity as perceived by the crowd.*

5. Experimental Setup

5.1. Datasets

Our study relies on 5 binary and 8 multi-class crowdsourced datasets. As finding public crowdsourced datasets with individual crowd votes is challenging, we chose ten datasets[6] (Table 1) provided by Appen for different text classification tasks. These datasets come from text-based crowdsourcing tasks (mainly classification), but the only "ground truth" information available comes from aggregating workers' votes. Neither the individual votes, nor the details on how the crowd experiment was run are available.

6. https://appen.com/resources/datasets/

TABLE 1: Dataset properties.

Dataset	Class	Train/val/test	Noise ratio
1. First GOP debate sentiment analysis (GOP2-sentiment)	2	6195/ 576/ 628	0.07
2. Disasters on social media (Disaster-relevance)	2	7557/ 740/ 684	0.09
3. Do these chemicals contribute to a disease? (Chemicals&Disease)	2	3088/ 186/ 162	0.16
4. Economic news article tone and relevance (News-relevance)	2	5098/ 452/ 526	0.13
5. Corporate messaging (Corporate-messaging)	3	2615/ 120/ 117	0.08
6. First GOP debate sentiment analysis-Sentiment (GOP3-sentiment)	3	11669/ 540/ 348	0.19
7. Twitter sentiment analysis: Self-driving cars (Self-driving-cars)	3	5399/ 150/ 147	0.04
8. Drug relation database (Drug-relation)	3	1866/ 75/ 72	0.01
9. Indian terrorism deaths database (Deaths-in-India)	3	25622/ 300/ 237	0.20
10. GOP tweets subject categorization (debate-subject)	5	1206/ 150/ 135	0.04

TABLE 2: Macro F1 and ECE results in % for NN1 and D-BERT with smoothed and soft targets.

	1.GOP2-sentiment		2.Disaster		3.Chemicals&Disease		4.News		5.Corporate Messaging		6.GOP3-sentiment		7.Self-driving cars		8.Drug Relation		9.Deaths in India		10.Debate-subject	
	F1	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**	**F1**	**ECE**
NN1-Hard labels	78,5	**11,0**	82,4	**9,8**	74,5	20,4	75,1	13,9	89,8	**3,2**	65,3	6,4	66,7	6,3	66,0	7,3	**83,2**	7,4	88,2	4,0
NN1-Soft	**79,7**	18,3	**82,9**	20,2	70,4	**6,4**	**75,8**	15,3	86,4	5,4	**66,6**	**4,9**	66,8	5,8	71,1	3,6	82,1	5,8	**88,9**	**2,1**
NN1 ($\alpha = 0.05$)	73,4	14,3	82,5	17,6	**76,5**	18,6	73,4	14,3	**90,6**	5,2	64,9	7,5	70,4	**5,0**	**73,7**	**3,0**	80,5	**4,8**	88,1	7,4
NN1 ($\alpha = 0.1$)	73,7	13,2	82,3	19,0	68,7	13,6	73,7	**13,2**	**90,6**	9,8	65,5	8,6	**70,8**	7,8	69,8	8,5	82,3	**4,8**	88,1	6,0
D-BERT-Hard labels	**78,2**	8,4	78,5	13,6	**75,2**	22,3	72,0	21,4	**89,7**	9,0	59,3	31,8	**59,0**	25,5	**80,7**	13,7	81,4	13,3	80,7	18,1
D-BERT-Soft	**78,2**	11,2	**82,3**	10,7	71,4	**16,4**	**72,7**	18,9	87,9	4,1	**63,1**	**11,1**	55,0	**17,7**	79,4	**4,9**	**84,0**	6,0	**83,7**	**3,8**
D-BERT ($\alpha = 0.05$)	59,2	18,2	79,9	12,8	65,6	25,8	72,6	**17,5**	87,1	6,7	59,9	25,3	52,3	25,2	**80,7**	8,9	81,2	10,3	80,0	15,4
D-BERT ($\alpha = 0.1$)	76,6	**3,3**	81,0	**3,5**	73,3	26,3	70,3	24,6	**89,7**	**2,3**	63,3	18,2	56,0	21,3	79,1	8,0	81,6	**5,1**	80,7	12,5

This situation required us to examine the ground-truth we had and assess how noisy are the aggregated labels. Two authors manually re-annotated each dataset to analyze how accurate the labels are. We checked the content (eg., tweets) and annotated it. We observed that on average, we do not agree with 10% of the aggregated crowd labels (and up to 20% for some datasets!). This indicates that crowdsourced labels can be noisy even after collecting multiple votes per sample. Our manual annotations of test datasets are presented in the (anonymized) project repository[7].

The aforementioned datasets give us only one statistic (i.e., confidence) about the distribution of labels[8]; however, we wanted to explore if having individual crowd votes rather than the aggregated ones and employing different label fusion algorithms might give us more insights. To this end, we use three additional textual datasets with actual crowd labels as well as the ground truths: i) Movie-Reviews (3-classes) [34], ii) Reuters-21578 (8-classes) [35], iii) Amazon-Reviews (binary)[9].

5.2. Configurations and Training

We evaluated: i) a simple one-layer neural network (NN1) with text vectorized via tf-idf, and ii) fine-tuned the DistilBERT model D-BERT [33] (6 layers, 768 hidden dim, 12 heads, 65M parameters) to compare the behavior on different models. We trained them with the cross-entropy loss using: i) *hard* targets (one-hot encoded labels), ii) *label smoothing* (Eq. 4), and iii) the proposed *soft* (Eq. 5) targets. We tested the impact of three label fusion methods: i) Majority Voting (MV), ii) D&S [31], which models worker reliability, and iii) GLAD [32], which further considers the task difficulty. Following [15], [36], we considered $\alpha = 0.05$ and $\alpha = 0.1$ for label smoothing.

7. https://github.com/Evgeneus/Label-Smoothing-in-Text-Classification
8. https://tinyurl.com/Calculate-a-Confidence-Score
9. https://github.com/TrentoCrowdAI/crowdsourced-datasets

TABLE 3: Performance of NN1 and D-BERT with targets obtained from different fusion methods.

Model	Movie-Reviews		Reuters-21578		Amazon-Reviews	
	ECE	**F1**	**ECE**	**F1**	**ECE**	**F1**
NN1-Hard labels (MV)	4,2	58,5	**4,7**	63,6	**11,0**	**95,8**
NN1-Soft (MV)	**3,1**	**59,8**	5,3	65,1	12,6	**95,8**
NN1 (α=0.05, MV)	8,2	53,5	6,1	64,6	17,3	95,7
NN1 (α=0.1, MV)	10,4	55,3	5,4	**65,5**	20,8	**95,8**
NN1-Hard labels (DS)	14,3	57,6	3,6	**72,4**	11,0	**95,8**
NN1-Soft (DS)	**8,7**	**58,5**	4,5	70,9	**10,6**	95,7
NN1 (α=0.05, DS)	9,5	56,1	**2,9**	70,8	17,3	**95,8**
NN1 (α=0.1, DS)	**8,7**	55,2	4,7	69,7	20,8	**95,8**
NN1-Hard labels (GLAD)	**4,5**	**57,7**	**3,1**	67,1	**10,5**	**96,0**
NN1-Soft (GLAD)	6,6	56,9	4,1	**70,6**	10,9	95,7
NN1 (α=0.05, GLAD)	9,3	52,5	3,7	68,6	17,1	95,8
NN1 (α=0.1, GLAD)	7,0	55,7	4,4	69,6	20,6	95,8
D-BERT-Hard labels (MV)	36,5	56,3	-	-	5,9	**93,0**
D-BERT-Soft (MV)	**21,1**	54,9	-	-	**5,3**	92,4
D-BERT (α=0.05, MV)	25,7	**57**	-	-	7,3	92,4
D-BERT (α=0.1, MV)	21,6	56,7	-	-	11,5	**93,0**
D-BERT-Hard labels (DS)	34,2	56,5	-	-	3,3	92,7
D-BERT-Soft (DS)	34,1	53,6	-	-	**2,0**	**93,1**
D-BERT (α=0.05, DS)	**23,0**	**58,5**	-	-	7,7	92,8
D-BERT (α=0.1, DS)	31,6	49,4	-	-	10,3	**93,1**
D-BERT-Hard labels (GLAD)	36,1	56,6	-	-	9,2	91,1
D-BERT-Soft (GLAD)	**20,4**	**58,2**	-	-	**2,1**	**94,1**
D-BERT (α=0.05, GLAD)	29,5	54,9	-	-	7,7	92,8
D-BERT (α=0.1, GLAD)	24,5	55,6	-	-	12,8	93,4

For each (NN1, training target) configuration, we performed grid search for the following hyperparameters on validation set: min Document Frequency, num. TF-IDF features, word n-gram range, learning rate, weight decay, and class weights. For each (D-BERT, training target) configuration, using validation set we searched for learning rate and a few options for class weights depending on the class distribution; the rest of the parameters remained as default. The networks were trained using ADAM optimizer iterated up to 500 epochs for NN1 and 100 epochs for D-BERT with early stopping. We set the batch size of 32 for D-BERT and the sequence length of 512 WordPiece tokens. The experiments details, datasets, and source code are available online[7].

Evaluation Metrics: To evaluate both prediction and confi-

TABLE 4: Macro F1 and ECE results in % with sHard targets; bold numbers shows sHard outperforms hard targets.

	1.GOP2-sentiment		2.Disaster		3.Chemicals&Disease		4.News		5.Corporate Messaging		6.GOP3-sentiment		7.Self-driving cars		8.Drug Relation		9.Deaths in India		10.Debate-subject	
	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE	F1	ECE
NN1-sHard	**79,8**	**6,8**	**82,9**	**9,2**	71,3	**15,4**	74,9	**11,1**	**90,6**	**2,0**	**66,0**	**5,8**	66,0	**4,2**	66,0	**5,6**	82,9	**4,6**	**88,9**	**3,4**
D-BERT-sHard	76,8	**5,5**	**80,5**	**12,4**	**77,1**	**20,3**	**72,6**	**15,9**	**90,6**	**7,6**	**60,1**	**28,1**	57,0	27,0	80,7	**11,9**	**86,2**	**9,3**	78,7	20,7

dence quality for the classification models, we use two metrics: i) macro F1 score and ii) ECE, which measures the difference in expected accuracy and expected confidence [37]. The smaller the ECE score, the better the calibration of the model.

6. Results

Table 2 reports the results of label smoothing and soft targets with NN1 and D-BERT. Across the ten Appen datasets, results are inconclusive: soft targets provided comparable results in terms of F1 while gave mixed results for ECE across the datasets: in many cases label smoothing collapsed ECE score in NN1 model (by 0.9% for α=0.05 and 1.5% for α=0.1 on average). The same applies independently of how we encode text (eg, the cited additional material report on LSTM encoding)[7].

D-BERT is a more interesting case as we know that calibration issues arise mainly with deep nets [2]. Training D-BERT model with soft labels from crowd gives an improvement in ECE for nine datasets (7.2% of average improvement across 10 datasets), and a boost in F1 for 6 out of 10 datasets. In contrast, label smoothing also can improve ECE (from 1% to 9% depending on the chosen α parameter) but always harm F1 score.

On datasets with individual crowd labels, we can also experiment with different fusion methods. We show the results in Table 3. Label smoothing here shows mixed results (improves on ECE on Movie-Reviews data but does significantly worse in ECE on Amazon-Reviews where the training labels are very accurate - and we do not have results for D-BERT on Reuters-21578 as the data did not contain raw texts) In contrast, the proposed soft target method improved ECE (up to 15.7%) across all datasets. Notably, for GLAD fusion method, soft target method enhanced *both* F1 (up to 15.7%) and ECE (up to 7.1%) on D-BERT model across the datasets with both high and low levels of label noise. This is in part due to the better performance of GLAD in truth inference and due to the fact that GLAD generates less skewed label distributions allowing the soft method to handle the noises.

7. Soft-Hard Targets

While the results from soft targets are promising, we also consider a different approach that does the one-side smoothing, that of the most likely label as identified by the fusion method. We call this approach as soft-hard (*sHard*) as it can be seen as a soft target approach but also as a hard one where samples are weighted by the ground truth probability of the most likely label (note that in this case $\pi(l|x)$ is no longer a valid distribution):

$$\pi_{sh}(l|x) = \begin{cases} \pi_f(l|votes, F), & l = l^*, \\ 0, & l \neq l^*, \end{cases} \quad (6)$$

We tested *sHard* (Eq. 6) targets following the configurations explained in Section 5.2. Results show that NN1 with *sHard* targets *always* led to improvement of ECE (1.7% on average), while F1 remained comparable to the models trained on hard labels (Table 4). When we compare the ECE performance of *sHard* to *Soft* and *label smoothing*, we see that *sHard* outperforms *label smoothing* on 7 datasets and *Soft* targets on 6 datasets. Training D-BERT model with sHard targets provides an improvement of ECE for 8 datasets out of ten (1.8% of average improvement across 10 datasets), and improves F1 for 7 datasets as to using *Hard* labels during the training. Finally, *sHard* improves ECE on 2 datasets that have individual crowd votes while remaining comparable on 1 dataset (Table 5).

TABLE 5: Performance of NN1 and D-BERT with sHard targets obtained from different fusion methods.

Model	Movie-Reviews		Reuters-21578		Amazon-Reviews	
	ECE	F1	ECE	F1	ECE	F1
NN1-sHard (MV)	4,3	**60,2**	5,1	**64,3**	**10,3**	95,5
NN1-sHard (DS)	**10,5**	**58,2**	3,7	72,3	**10,4**	95,7
NN1-sHard (GLAD)	5,7	**58,6**	3,4	67,1	10,8	95,8
D-BERT-sHard (MV)	**35,9**	**57,4**	-	-	**4,9**	92,5
D-BERT-sHard (DS)	**30,0**	**57,9**	-	-	3,3	92,1
D-BERT-sHard (GLAD)	**35,2**	56,4	-	-	**5,1**	**93,3**

TABLE 6: Avg. improvement in % to hard labels.

Model	F1	ECE	Model	F1	ECE
NN1-sHard	0.04	**-1.7**	D-BERT-sHard	**0.6**	-1.8
NN1-Soft	0.1	-0.7	D-BERT-Soft	0.3	**-7.2**
NN1 (α=0.05)	**0.3**	0.9	D-BERT (α=0.05)	-3.6	-1.1
NN1 (α=0.1)	-0.4	1.5	D-BERT (α=0.1)	-0.3	-5.2

Table 6 summarizes the results by showing the average improvement of the models trained on sHard/Soft/Smoothed targets to the models trained on hard targets (our baselines). Results show that our proposed *Soft* and *sHard* target methods substantially improves the model calibration.

8. Limitations - and the Road Ahead

This work scratches the surfaces of hybrid classification services and the science behind them. We have shown the centrality of calibration in contexts where the loss function is skewed and where the cost of errors is high compared to the cost of asking humans. We have also shown the effect of soft and soft-hard targets in text classification with crowdsourced data, across several datasets and fusion methods. The effect on calibration error and the benefits of the proposed

approach are manifest for deep models, that are known to be more affected by calibration issues. While promising, the initial work requires deeper investigations: we need to expand experiment to other deep network architectures and get a deeper understanding of what drives the behaviors we are seeing. The same is true for label fusion methods and related datasets and classification problems, including especially the classification problems with a high number of classes and varying degree of noise. Finally, we need to integrate the modules into an hybrid classification service and test it "end to end", progressively building a science that can eventually put classification tasks on autopilot so that crowd and ML, and their integration, becomes a commodity rather than an art and a hard challenge.

Acknowledgments

This work was supported by the Russian Science Foundation (Project No. 19-18-00282).

References

[1] E. Krivosheev, F. Casati, M. Baez, and B. Benatallah, "Combining crowd and machines for multi-predicate item screening," *Proc. ACM Hum.-Comput. Interact.*, vol. 2, no. CSCW, Nov. 2018. [Online]. Available: https://doi.org/10.1145/3274366

[2] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 1321–1330.

[3] E. Krivosheev, A. Bozzon, and F. Casati, "Active hybrid classification," in *Procs of NeurIPS Workshop on Human in the Loop Dialogue Systems*, 2020.

[4] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 1321–1330.

[5] J. Ramirez, E. Krivosheev, M. Baez, F. Casati, and B. Benatallah, "Crowdrev: a platform for crowd-based screening of literature reviews," *arXiv preprint arXiv:1805.12376*, 2018.

[6] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the em algorithm," *Journal of the Royal Statistical Society. Series C Applied Statistics*, vol. 28, no. 1, 1979.

[7] E. Krivosheev, F. Casati, and B. Benatallah, "Crowd-based multi-predicate screening of papers in literature reviews," in *WebConf 2018*, 2018, p. 55–64.

[8] E. Kamar, S. Hacker, and E. Horvitz, "Combining human and machine intelligence in large-scale crowdsourcing," ser. AAMAS '12. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 467–474.

[9] J. Cheng and M. S. Bernstein, "Flock: Hybrid crowd-machine learning classifiers," in *Proceedings of ACM CSCW*. New York, NY, USA: ACM, 2015.

[10] C. Rodriguez, F. Daniel, and F. Casati, "Crowd-based mining of reusable process model patterns," in *Business Process Management*, S. Sadiq, P. Soffer, and H. Völzer, Eds. Springer International Publishing, 2014, pp. 51–66.

[11] W. S. Lasecki, C. D. Miller, I. Naim, R. Kushalnagar, A. Sadilek, D. Gildea, and J. P. Bigham, "Scribe: Deep integration of human and machine intelligence to caption speech in real time," *ACM Communications*, vol. 60, no. 9, 2017.

[12] W. Callaghan, J. Goh, M. Mohareb, A. Lim, and E. Law, "Mechanicalheart: A human-machine framework for the classification of phonocardiograms," *Proc. ACM Hum. Comput. Interact.*, vol. 2, no. CSCW, pp. 28:1–28:17, 2018.

[13] A. T. Nguyen, B. C. Wallace, and M. Lease, "Combining Crowd and Expert Labels using Decision Theoretic Active Learning," *Hcomp 2015*.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,*, 2016. [Online]. Available: http://arxiv.org/abs/1512.00567

[15] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" in *NeurIPS*, 2019.

[16] H. Zhang, F. Petitjean, and W. Buntine, "Bayesian network classifiers using ensembles and smoothing," *Knowledge and Information Systems*, 03 2020.

[17] H. Zhang, F. Petitjean, and W. Buntine, "Hierarchical gradient smoothing for probability estimation trees," in *Advances in Knowledge Discovery and Data Mining*. Cham: Springer International Publishing, 2020, pp. 222–234.

[18] B. Chen and C. Cherry, "A systematic comparison of smoothing techniques for sentence-level BLEU," in *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, Maryland, USA, Jun. 2014. [Online]. Available: https://www.aclweb.org/anthology/W14-3346

[19] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," *CoRR*, vol. abs/1612.02695, 2016. [Online]. Available: http://arxiv.org/abs/1612.02695

[20] Z. Chen, W. Lin, S. Wang, L. Xu, and L. Li, "Image quality assessment guided deep neural networks training," 2017.

[21] M. Mezzini, "Empirical study on label smoothing in neural networks," 2018.

[22] M. Lukasik, S. Bhojanapalli, A. K. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" in *International Conference on Machine Learning*, 2020.

[23] P. Welinder, S. Branson, S. Belongie, and P. Perona, "The multidimensional wisdom of crowds," in *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'10. Red Hook, NY, USA: Curran Associates Inc., 2010, p. 2424–2432.

[24] N. Q. V. Hung, N. T. Tam, L. N. Tran, and K. Aberer, "An evaluation of aggregation techniques in crowdsourcing," in *International Conference on Web Information Systems Engineering*, vol. 8181, 10 2013, pp. 1–15.

[25] E. Wulczyn, N. Thain, and L. Dixon, "Ex machina: Personal attacks seen at scale," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 1391–1399.

[26] A. M. Aung and J. Whitehill, "Harnessing label uncertainty to improve modeling: An application to student engagement recognition." in *FG*, 2018, pp. 166–170.

[27] J. C. Peterson, R. M. Battleday, T. L. Griffiths, and O. Russakovsky, "Human uncertainty makes classification more robust," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9617–9626.

[28] B. Zhang, G. Essl, and E. Mower Provost, "Predicting the distribution of emotion perception: capturing inter-rater variability," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 51–59.

[29] A. Ando, S. Kobashikawa, H. Kamiyama, R. Masumura, Y. Ijima, and Y. Aono, "Soft-target training with ambiguous emotional utterances for dnn-based speech emotion classification," in *2018 IEEE ICASSP*. IEEE, 2018, pp. 4964–4968.

[30] Y. Xue and M. Hauskrecht, "Efficient learning of classification models from soft-label information by binning and ranking," in *The Thirtieth International Flairs Conference*, 2017.

[31] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the em algorithm," *Journal of the Royal Statistical Society. Series C Applied Statistics*, vol. 28, no. 1, 1979.

[32] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *Advances in neural information processing systems*, 2009, pp. 2035–2043.

[33] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[34] F. Rodrigues and F. C. Pereira, "Deep learning from crowds," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[35] F. Rodrigues, M. Lourenco, B. Ribeiro, and F. C. Pereira, "Learning supervised topic models for classification and regression from crowds," *IEEE transactions on PAMI*, vol. 39, no. 12, 2017.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, 06 2017.

[37] M. P. Naeini, G. F. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using bayesian binning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, ser. AAAI'15. AAAI Press, 2015, p. 2901–2907.