CrowdRL: An End-to-End Reinforcement Learning Framework for Data Labelling

Kaiyu Li*, Guoliang Li*, Yong Wang*, Yan Huang†, Zitao Liu† and Zhongqin Wu†
*Tsinghua University, Beijing, China †TAL Education, Beijing, China
ky-li18@mails.tsinghua.edu.cn, liguoliang@tsinghua.edu.cn, wangy18@mails.tsinghua.edu.cn
galehuang@100tal.com, liuzitao@100tal.com

***Abstract*—Data labelling is very important in many database and machine learning applications. Traditional methods rely on humans (workers or experts) to acquire labels. However, the human cost is rather expensive for a large dataset. Active learning based methods only label a small set of data with large uncertainty, train a model on these labelled data, and use the trained model to label the remainder unlabelled data. However they have two limitations. First, they cannot judiciously select appropriate data (task selection) and assign the tasks to proper humans (task assignment). Moreover, they independently process task selection and task assignment, which cannot capture the correlation between them. Second, they simply infer the truth of a task based on the answers from humans and the trained model (truth inference) by independently modeling humans and models. In other words, they ignore the correlation between them (the labelled data may have noise caused by humans with biases, and the model trained by the noisy labels may bring additional biases), and thus lead to poor inference results.**

To address these limitations, in this paper, we propose CrowdRL, an end-to-end reinforcement learning (RL) based framework for data labelling. To the best of our knowledge, CrowdRL is the first RL framework designed for the data labelling workflow by seamlessly integrating task selection, task assignment and truth inference together. CrowdRL fully utilizes the power of heterogeneous annotators (experts and crowdsourcing workers) and machine learning models together to infer the truth, which highly improves the quality of data labelling. CrowdRL uses RL to model task assignment and task selection, and designs an agent to judiciously assign tasks to appropriate workers. CrowdRL jointly models the answers of workers, experts and models, and designs a joint inference model to infer the truths. Experimental results on real datasets show that CrowdRL outperforms state-of-the-art approaches with the same (even fewer) monetary cost while achieving 5%-20% higher accuracy.

***Index Terms*—reinforcement learning, crowdsourcing, data labelling, truth inference**

I. Introduction

Data labelling is important in many database and machine learning applications, as the quality of labeling data (training data) could highly influence the performance of model training [17], [18]. Crowdsourcing has become an important way for data labelling, because it is easy to recruit workers on crowdsourcing platforms [10], [20], [23], [28], [33], [40], [47]. Besides, labels from workers are more accurate than algorithms, e.g., recognizing an image. However, acquiring labels from crowdsourcing workers has two challenges.

*Guoliang Li is the corresponding author. This work was supported by NSF of China (61632016,61925205,62041204), National Key R&D Program of China (2020AAA0104500), Huawei, BNRist, and TAL education.

Challenge 1: Neglect the Correlations between Task Selection and Task Assignment. Many tasks require qualified workers with expertises to label. For example, crowdsourcing workers cannot decide if a medical image contains a tumor. Thus it requires to involve experts to label the task. There are two problems we need to address: (1) *Task Selection*: how to select appropriate unlabelled data to label [26]; (2) *Task Assignment*: how to assign the selected tasks to appropriate annotators (crowdsourcing workers or experts) [49]. Existing studies first select some unlabelled tasks with high uncertainty to label [26] and then assign the selected tasks to appropriate annotators [49]. However, they neglect the correlation between task selection and assignment. For example, they first select some tasks to label but cannot find appropriate annotators for the selected tasks. Thus it calls for a unified framework for task selection and assignment.

Challenge 2: Neglect the Correlation between Workers, Experts, and Learned Models. It is expensive to recruit many workers to label the data, especially for a large dataset. To address this problem, active learning (AL) based methods [8], [26] are proposed. These methods iteratively train a model using labelled data, select unlabelled data with the maximum uncertainty, ask humans to label them [26], and then train a model using the labelled data and use the trained model to label other remainder unlabelled data. Note that the AL methods assume the answers from annotators are correct, but in fact annotators may return noisy results. Thus *truth inference* is proposed to infer the truth of each task from the labelled results of multiple annotators. Existing studies focus on inferring the truth of labels by annotators, but neglect that machine learning models trained by the labelled data can also be used to infer the truth, which could reduce the monetary cost of recruiting annotators. For example, for an unlabelled image of tumor, we need 5 medical experts to label it. Supposing a trained model classifies it as 'positive', we recruit three medical workers and if they all label it as 'positive', then we can label this image as 'positive' with fewer cost. A simple method that involves a trained model into the truth inference is to take the model as an annotator. However, the labelled data may have noise caused by annotators with known biases, and thus the model trained by the noise labels may bring additional biases. Moreover, the biases of the trained model depend on the noises of annotators and are hard to model. Thus it calls for a joint model to capture the unknown distribution of the expertises of annotators and the trained model.

An End-to-End Reinforcement Learning Based Data Labeling Framework. In this paper, we propose CrowdRL, an end-to-end reinforcement learning (RL) framework for data labelling. CrowdRL designs a unified data labelling framework for integrating the processes of task selection, task assignment and truth inference into a unified framework. In other words, CrowdRL utilizes both the cost-effectiveness of machine learning models and high accuracy of human labors, which is a better trade-off of monetary cost and labelling quality. Our unified optimization framework can get better labelling quality than isolating the two steps. Specifically, CrowdRL first selects a small portion of tasks and asks annotators to label them. Then CrowdRL iteratively repeats the following steps until it labels all the data or the budget is used up: (1) trains a model using labelled data, uses this model to label some unlabelled tasks with high confidence, and updates annotators' quality and the set of unlabelled data; (2) selects a batch of objects to label and assigns these tasks to appropriate annotators; (3) infers the true labels of these objects based on the answers from annotators.

Unified Task Selection and Assignment. CrowdRL uses RL to model task assignment and task selection, and designs an agent to judiciously assign tasks to appropriate workers. Specifically, we formalize the answers of questions that have been answered by annotators, the cost and quality of annotators as current '`State`' of CrowdRL. We formalize the joint operation of task selection and task assignment as the '`Action`' of CrowdRL. In each iteration of labelling, we model the policy of taking an action by predicting an expected optimal action based on the current state using a deep Q-network [24] (DQN). By replaying the experience of taking actions of task assignments and task selections, and getting feedbacks from the labeling history, the policy model will be iteratively updated and becomes better and better.

Joint Truth Inference Model. CrowdRL fully utilizes the power of heterogeneous annotators (experts and workers) and trained models together to infer the data labels, which highly increases the accuracy of data labelling. Specifically, we propose a joint inference model to jointly model the unknown distribution of the expertises of annotators and the model, and use the joint model to infer the truth of each task.

Main Contributions. We make the following contributions.

(1) We propose CrowdRL, an end-to-end reinforcement learning model for data labelling. To the best of our knowledge, we are the first to propose a unified data labelling framework based on an RL model (Section III).

(2) CrowdRL models task assignment and task selection together, and designs an agent to judiciously assign tasks to appropriate workers using a neural network (Section IV).

(3) CrowdRL jointly models the answers of workers, experts and models, and designs a joint inference model to infer the truths (Section V).

Notation	Definition
$\mathbb{O} = \{o_i\}$	a set of objects
$\mathbb{C} = \{c_j\}$	a set of classes
$\mathbb{W} = \{w_i\}$	a set of annotators
$\Pi^i = \{\pi^i_{j,k}\}$	a $\lvert\mathbb{C}\rvert \times \lvert\mathbb{C}\rvert$ confusion matrix of w_i
y_i	true label of o_i
$\hat{y}_i^{w_j}$	label of o_i from annotator w_j
$\hat{\mathbf{y}}_i$	answer set of o_i from multiple annotators
$p(\cdot)$	probability function
ϕ	classifier for a multi-class classification
$\phi_{c_j}(o_i)$	$p(y_i = c_j; \phi)$, i.e., probability of ϕ labels o_i as c_j
B	budget of cost
S, A, E, R	state, action, environment and reward of CrowdRL

TABLE I: Table of Notations.

(4) We have conducted experiments on three real-world datasets, and experimental results show that our method outperformed state-of-the-art approaches by 5%-20% higher accuracy while keeping the same (even lower) monetary cost. (Section VI).

II. Preliminaries

A. Problem Definition

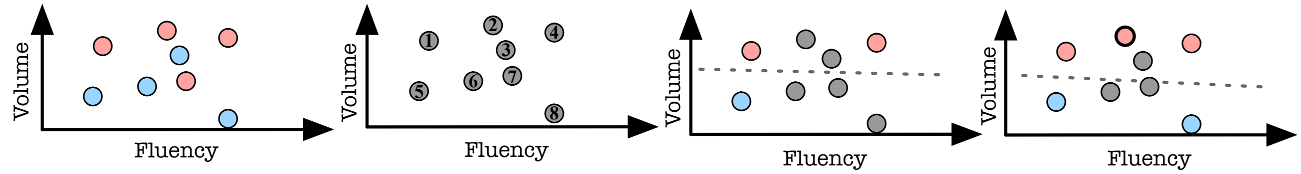
Data Model. Consider a set of objects $\mathbb{O} = \{o_i\}$ where each object o_i has a true label y_i. However, the true label of o_i is unknown and we only know y_i is in a set of given labels $\mathbb{C} = \{c_j\}$. We have two ways to get the label of y_i – asking annotators (crowdsourcing workers or experts) to label o_i or using a classifier algorithm to compute a label of o_i.

Annotator Model. There are two types of annotators in our model – workers from crowdsourcing platforms and experts with domain knowledges on $\mathbb{O}$. For ease of presentation, we denote the set of all annotators as $\mathbb{W} = \{w_j\}$. Given an object o_i, if an annotator w_j labels the object, s/he returns a result $\hat{y}_i^{w_j} \in \mathbb{C}$, and the answer set of object o_i from multiple annotators is denoted as $\hat{\mathbf{y}}_i$.

Following the classical definition [48], [49], the expertise of annotator w_i could be formalized as a $|\mathbb{C}| \times |\mathbb{C}|$ confusion matrix $\Pi^i = \{\pi^i_{j,k}\}$, where $\pi^i_{j,k}$ denotes the probability of acquiring a label c_k for an object with true label c_j from annotator w_i. Notice that we do not know the true value of Π^i in advance, but we iteratively update the estimation of Π^i during the labelling process which is denoted as $\hat{\Pi}^i$.

Classifier Model. Given an object o_i, a classifier algorithm ϕ predicts a result $\hat{y}_i^c \in \mathbb{C}$ of y_i, i.e., $\phi(o_i) = \hat{y}_i^c$. Note that the classifier model may need to train the model with some training data, which can be gotten by assigning some objects to some workers/experts and inferring the truth based on these labelled results.

Labelling Workflow. Usually annotators have higher labelling quality than classifiers but take more monetary cost, and thus a labelling process should judiciously select annotators and classifiers to label the objects. Initially, we select a small portion of objects in $\mathbb{O}$ and ask annotators to label them. Then, we iteratively (1) train a classifier ϕ using labelled data and use ϕ to label some unlabelled objects with high confidence (labelled set enrichment), (2) select a batch of objects to be labelled (task selection), (3) assign these tasks to annotators



(a) Ground Truth of Classification (b) Unlabelled Objects (c) 1st Iteration of Labelling (d) 2nd Iteration of Labelling

Fig. 1: Labelling Example. There are 8 videos of primary school students' oral reports. We aim to label them for a binary classification task. The red circles and blue circles represent excellent presentations (*positive*) and awful presentations (*negative*) respectively. Suppose we consider two features, fluency (x-axis) and volume (y-axis) of the videos.

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	cost	quality
$w_1(worker)$	1	1	-1	1	0	-1	-1	1	1	0.65
$w_2(worker)$	0	1	-1	0	0	-1	-1	0	1	0.62
$w_3(worker)$	-1	1	-1	1	-1	-1	-1	-1	1	0.60
$w_4(expert)$	1	1	-1	-1	-1	-1	-1	0	5	0.985
$w_5(expert)$	-1	1	-1	-1	0	-1	-1	-1	5	1.0

TABLE II: $\mathsf{S}(3)$ (Labelling History and Annotators' Costs and Qualities)

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8
w_1	$\times$	3	1	$\times$	$\times$	1	3	4
w_2	$\times$	1	1	$\times$	$\times$	2	2	1
w_3	$\times$	1	1	$\times$	$\times$	1	0	3
w_4	$\times$	2	2	$\times$	$\times$	1	1	0
w_5	$\times$	2	4	$\times$	$\times$	2	1	2

TABLE III: Distribution of $Q(\mathsf{S}(2), \mathsf{A}(2); \theta)$

	$+$	$-$
$+$	0.60	0.40
$-$	0.30	0.70

TABLE IV: Confusion Matrix of w_1

	$+$	$-$
$+$	**0.98**	0.02
$-$	0.01	**0.99**

TABLE V: Confusion Matrix of w_4

$\times$: $-\infty$
$+$: *positive*
$-$: *negative*

(task assignment), and (4) infer the true labels of these objects based on the answers from annotators (truth inference). We repeat these steps until the budget of asking annotators to label objects is used up.

Labelled Set Enrichment. After a classifier ϕ is trained, it is used to classify unlabelled objects in $\mathbb{O}$. However, the objects classified with low confidence might be wrongly classified. For example, if o_i is classified as positive with a confidence of 0.52, it might be indeed a negative object. Thus, in the step of labelled set enrichment, we only use ϕ to label the objects with high confidence, and the others remain to be unlabelled.

Task Selection (TS). To iteratively label objects, the object set $\mathbb{O}$ is split into several batches and we ask the annotators to label a batch of objects in each iteration. Given labelling history and unlabelled objects, TS considers which unlabelled objects are selected in each iteration.

Task Assignment (TA). Given annotators with expertises and a batch of objects selected by a TS algorithm, a TA algorithm focuses on how to assign these questions to appropriate annotators to obtain the true labels with the maximum probability.

Traditional methods independently process TS and TA, which neglect their correlation. For example, suppose o_i and o_j are top-2 selected objects by a task selection method, e.g., bootstrap uncertainty [26]. Suppose o_i is hard to be labelled (all the annotators cannot correctly label it), but o_j is easy to be labelled. If we consider task selection and assignment independently, they will first select o_i but the labelling quality will be low. However, selecting o_j is more useful for the labelling process. To this end, we jointly process TS and TA together and model TS and TA as a unified operation.

Truth Inference (TI). Given an object o_i, it may be labeled by workers, experts and classifier algorithms. We can combine the answer and infer the truth based on the answer set to infer a label $\hat{y}_i$. Given answers from these annotators, a TI model aims at inferring $\hat{y}_i$ with the maximum probability. A naive method is majority voting [48]. However, different annotators have different expertise for labelling data, e.g., some experts are more experienced and some workers may be not qualified for labelling $\mathbb{O}$. Moreover the labeling quality of a classifier depends on the labelling quality of annotators, and in other words, the annotators and the classifier have correlations. Thus we integrate the answers from workers, experts and classifier algorithms by considering their expertises (or confidences).

Problem Formulation. Given a set $\mathbb{O}$ of objects and a budget B, we aim to (1) task selection and assignment: we select unlabelled objects and assign objects to workers, experts, classifiers such that the total cost is within B. (2) truth inference: for each object, we infer the truth in order to maximize the quality of labeling $\mathbb{O}$. [1]

Example 1: As shown in Figure 1. Suppose 8 videos of students' oral reports are to be labelled in a binary classification task, i.e., $\mathbb{C}$={'*positive*', '*negative*'} and $|\mathbb{C}| = 2$. The ground truth is shown in Figure 1 (a). The red circles and blue circles represent excellent presentations (*positive*) and awful presentations (*negative*) respectively. Formally, $\mathbb{O} = \{o_1, o_2, \ldots, o_8\}$ where each $o_i \in \mathbb{O}$ represents a video clip. Given 3 workers and 2 experts as shown in Table II, denoted as $\mathbb{W} = \{w_1, w_2 \ldots w_5\}$. Suppose the true label of o_1 is $y_1 =$'*positive*' and we employ w_1, w_2 and w_4 to label it. The answers from them are $\hat{\mathbf{y}}_1$ = {'*positive*', '*negative*', '*positive*'} and $\hat{y}_1^{w_1} =$'*positive*'. We could infer the true label of o_1 as $\hat{y}_1 =$'*positive*' using truth inference models, e.g., majority voting [48]. The confusion matrices of w_1 and w_4, i.e., Π^1 and

[1] In this paper, the workers could not communicate with each other. We do not make any assumptions that there are some rules to label the datasets such as CrowdGame [21]. Also, we do not make any assumptions about the distribution of worker quality or tasks.

Π^4 are shown in Tables IV and V. The element $\pi^4_{2,2} = 0.99$ denotes w_4 has a probability of 0.99 to label a negative object as '*negative*'. The budget is $\mathsf{B} = 30$ and the cost of employing one worker and expert are 1 and 5 respectively. Initially, we do not know the true labels of these 8 examples, thus they are all marked as grey circles as shown in Figure 1(b). Here, we study how to label all of the 8 videos in $\mathbb{O}$ with maximum labelling quality before running out 30 units of budget.

B. Related Work

We review related works – crowdsourcing, active learning and reinforcement learning algorithms for better understanding of CrowdRL labelling framework.

1) Crowdsourcing Methods: Crowdsourcing aims to harness workers' knowledge to process machine-hard tasks [6], [7], [36], [45]. In crowdsourcing, requesters split a complicated task into many micro-tasks and publish them on crowdsourcing platforms such as Amazon Mechanical Turk. The employed workers answer the questions while getting monetary rewards. Task selection [8], [18], task assignment [32], [43], [49], and truth inference [3], [19], [27], [48] are three crucial problems.

However, traditional crowdsourcing frameworks consider TS, TA and TI independently. Intuitively, the answer quality of tasks are highly related to selected annotators for answering them. Thus, CrowdRL integrates these three steps into a unified RL model. Additionally, existing crowdsourcing methods do not consider using answers from learning model to infer the truth. CrowdRL utilizes the data features to infer the true labels along with annotators' answers, which is different from existing studies [29], [39].

2) Active Learning Methods: Active learning (AL) methods combine both machine and human labors to solve artificial intelligence tasks in recent years [44]. An AL method aims at judiciously labelling a small subset of objects in a dataset and uses these labeled data to train a model [4], [26]. An AL method iteratively labels a batch of objects and selects the examples with the highest uncertainties in each iteration. Among these AL methods, statistical AL model and its variants are the most popular [4], [8], [26].

Different from traditional AL methods, CrowdRL uses a deep neural network to select the tasks and integrate TS and TA as a unified operation. The algorithm of TS and TA become smarter and smarter as it is integrated into a reinforcement learning model, which could update the strategy by considering the feedback in each labelling iteration. To our best knowledge, we are the first to model a human-in-the-loop labelling task as a unified reinforcement learning framework by using heterogeneous annotators. Although Shan et al. [32] introduced RL techniques for crowdsourcing, it only uses RL framework for trading-off the benefit of both requesters and workers in TA. Instead, CrowdRL focuses on building an end-to-end framework for the whole labelling workflow.

Some existing studies propose the concept of "AI Worker", which utilizes learned models (e.g., classification or clustering methods [15]) as AI workers. In each labelling iteration, the human workers first labeled some objects and then it learned a model based on the labeled data as an AI worker. Then, the "AI Worker" predicted the labels for the unlabelled objects. For each object, if the confidence of the prediction was higher than a threshold, it would be labelled by the AI worker; otherwise it would be assigned to human workers. However, they assume the human workers always return correct answers, and they use learned models and human workers independently. Comparing with these methods, CrowdRL uses a joint inference model by integrating answers from human annotators and machine algorithms, and the model parameter, the quality of human annotators and the truth of labels will be inferred jointly.

The method of data programming aims at integrating the results from several weak supervised sources to infer the truth, e.g., Snorkel [29], Osprey [5] and GOGGLES [9]. They first define some labelling functions (LFs) or weak rules, and then use the rules (written by human experts) to infer the truth. However, in many scenarios, e.g., our audio labeling task, it is hard to define such rules. Thus our method is more general for data labeling.

3) Reinforcement Learning: Reinforcement learning (RL) methods use iteration algorithms (e.g., value iteration, policy iteration or both) to find the optimal or sub-optimal solution for an optimization of control problem [35] by iteratively updating the parameters of RL models and expect the models to converge to the optimal. Due to the rapid development of deep learning, the techniques of deep reinforcement learning (DRL) have attracted the attention of researchers [25]. In the past decade, DRL and its variants have outperformed both traditional close-loop control methods, e.g., model predictive control [11], [14], and supervised learning methods, e.g., classical neural network, on many tasks [25], [34], [38].

Traditional algorithms fail to integrate the whole process of data labelling, and thus TS, TA and TI are independent, which may potentially decrease the precision of all of the three components and finally leads to low labelling accuracy. In this paper, we model the process of data labelling as an end-to-end RL model using deep neural network. The challenge is to model each of the components of RL, e.g., designing the `State` and `Action` of an RL model. Different from other RL frameworks, the `Environment` part of CrowdRL, i.e., the feedback, is determined by labels from annotators. Comparing with traditional RL problems, the feedback of CrowdRL includes much uncertainty and need to be well formalized, e.g., the feedbacks of playing Go [34] is based on the frequency of winning the game, the feedback of autonomous driving is based on physical rules [31].

III. Reinforcement Learning Framework for Data Labelling

A. Overview of CrowdRL

In this section, we propose CrowdRL, which integrates task selection, task assignment, and truth inference into a unified end-to-end reinforcement learning framework for data labelling. The overview design of CrowdRL is shown in Fig 2. The technical details of the `Agent` part and `Environment` part are discussed in Section IV and V respectively.

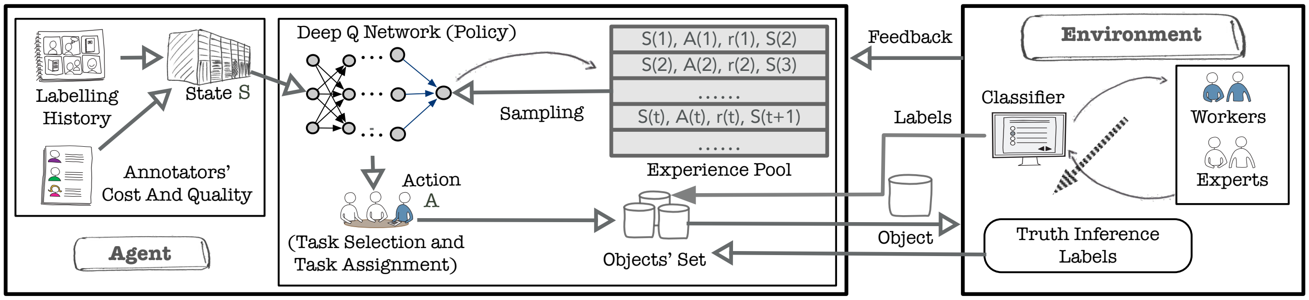


Fig. 2: Reinforcement Learning Based Data Labelling Framework.

CrowdRL uses a unified end-to-end reinforcement learning framework to jointly address the three problems, task selection, task assignment, and truth inference. It mainly includes six components: (1) `Environment` infers the truth based on the answers from workers and classifiers; (2) `Agent` analyzes the historical process and maintains this information into `States`. Then it makes an `Action` to select objects and assigns objects to appropriate workers based on the `Reward` according to historical labelling qualities and costs; (3) `State` captures the labelling history, including annotators' estimated qualities and costs; (4) `Action` conducts the joint operation of task selection and assignment on the `Environment` by utilizing the `State`; (5) `Reward` is formalized as the weighted summation of the number of objects labelled by classifier ϕ and the cost of future labelling iteration, i.e., the long-term reward; (6) `Policy` is a function f : `State` $\rightarrow$ `Action`, i.e., the strategy of conducting an `Action` based on the current `State`. CrowdRL uses a deep network to represent policy f.

We formally describe the workflow of CrowdRL in Algorithm 1. First, we initialize state $\mathsf{S}(0)$ and initialize it by annotators' costs and qualities according to labelling history. Initially, we select a small portion (with a ratio of $\alpha \in (0, 1)$) of the objects, and ask the annotators to label them. In each labelling iteration, we train the classifier model using these labelled objects and label some unlabelled objects with high confidence, i.e., labelled set enrichment, then the `Agent` decides how to make actions of task selection and assignment based on the feedback (`Reward`) from the `Environment`. `Environment` conducts truth inference based on the annotators' answers of these assignments, computes a reward of the assignment, and updates the classifier. After several labelling iterations, `Agent` has enough experience to learn an optimal policy for task selection and assignment.

B. *CrowdRL Modeling*

We discuss the details of each component of CrowdRL.

State S. Before the t-th iteration of labelling, we could observe the answered questions and the labels of these questions. Additionally, we have the estimated qualities and costs of annotators. Intuitively, we aim at selecting objects which could be answered with high quality and low cost, by observing current seen information.

To this end, we consider two important features of states as shown in Figure 2: (1) Labelling history: we model the labelling history as a $|\mathbb{O}| \times |\mathbb{W}|$ matrix, where the element $\mathsf{S}[i, j]$ at i-th row and j-th column denotes the answer of labelling o_j from i-th annotator, each value of the elements $\mathsf{S}[i, j]$ has $|\mathbb{C}| + 1$ possibilities:

$$\mathsf{S}(t)[i,j] = \begin{cases} -1 & w_i \textit{ has not labelled } o_j \textit{ until } t \\ c\ (\neq -1) & w_i \textit{ labels } o_j \textit{ as class } c,\ c \in \mathbb{C} \end{cases}$$

Thus, the labelling history has totally $(|\mathbb{C}| + 1)^{|\mathbb{O}||\mathbb{W}|}$ possibilities, i.e., the scale of state space is $(|\mathbb{C}| + 1)^{|\mathbb{O}||\mathbb{W}|}$; (2) Annotator's cost and quality: $\mathsf{S}(t)[i, |\mathbb{O}| + 1]$ denotes the cost of the i-th annotator and $\mathsf{S}(t)[i, |\mathbb{O}| + 2]$ denotes the estimated quality of the i-th annotator. Note that the confusion matrix Π^i is invisible for us, thus we only update the estimation $\hat{\Pi}^i$ of Π^i at the end of each iteration. We use the value $\frac{tr(\hat{\Pi}^i)}{|\mathbb{W}|} = \frac{\sum_{\pi^i_{j,j} \in \hat{\Pi}^i} \pi^i_{j,j}}{|\mathbb{W}|}$ as the overall estimated quality of w_i where $tr(\cdot)$ denotes the trace of a matrix, i.e., the summation of all the elements on the main diagonal of a matrix. The cost of each annotator is stable over the labelling process. For example, the state in Table II corresponds to Figure 1 (d), objects o_1, o_4, o_5 and o_8 are labelled by annotators and o_2 is labelled by classifier ϕ, for o_1, it is labelled by w_1, w_2 and w_4 as '`positive`', '`negative`' and '`positive`' respectively. The cost of employing a worker and an expert are 1 unit and 5 units respectively. The estimated quality of w_4 is $\frac{0.98+0.99}{2} = 0.985$ based on the confusion matrix in Table V.

Action A. We combine task selection and task assignment as a unified joint operation to benefit the process of labelling. We model the action $\mathsf{A}(t)$ as a pair (i, j) which denotes assigning o_j to w_i. Thus there $|\mathbb{O}||\mathbb{W}|$ possibilities of $\mathsf{A}(t)$. As discussed in Section IV, we use a function $Q(\mathsf{S}(t), \mathsf{A}(t))$ to denote the 'Q-value' (long-term reward) of taking action $\mathsf{A}(t)$ on state $\mathsf{S}(t)$. We would compute the value function of each $Q(\mathsf{S}(t), \mathsf{A}(t))$ and select the combination of tasks and annotators with biggest 'Q value' as the optimal action.

Environment E. At the t-th iteration, the `Agent` conducts an action $\mathsf{A}(t)$ on the environment E and gets feedback including the reward. Then $\mathsf{S}(t)$ is updated to $\mathsf{S}(t + 1)$. Specifically, the environment E could infer the true labels of the selected objects from $\mathsf{A}(t)$ and update the estimations of annotators' qualities. Our labelling model considers the prediction from classifier ϕ in E to infer the true labels of objects. The details of designing E are demonstrated in Section V.

Reward R. In the labelling task, we aim to get the highest labelling quality after running out the budget, thus the long-term reward at t-th iteration is denoted as:

$$\mathsf{R}(t) = \sum_{T=t}^{+\infty} \gamma^{T-1} r(T) \tag{1}$$

where $\gamma \in (0, 1]$ is the discount factor and $r(t)$ is the reward of the t-th iteration of labelling.

In each iteration, once the labelled set is enriched, we retrain the classifier ϕ and label some unlabelled objects with high confidence. Intuitively, if many objects are labelled by ϕ, it indicates both high accuracy (the classifier is trained by labelled data) and low cost of the labelling process. Besides, we also consider the monetary cost at each iteration, thus we should integrate the cost of employing annotators into $r(t)$. We denote $r(t) = \lambda r_\phi(t) + \frac{\mu}{r_{cost}(t)}$ where $r_\phi(t) = \frac{|labelled\ objects\ by\ \phi|}{|unlabelled\ objects|}$, $r_{cost}(t)$ is the monetary cost, λ and μ are the weights of $r_\phi(t)$ and $r_{cost}(t)$ respectively.

***Example** 2:* See the example in Figure 1 (b), (c) and (d). Initially, for ease of presentation, assume $\alpha = 0.25$, i.e., labelling 8*0.25=2 objects. We select o_1 and o_4 and ask the annotators to label them. For each iteration, we select one object and ask three annotators to label it. After the first iteration, three objects o_1, o_4 and o_5 are labelled as shown in Figure 1 (c). We use $f(\mathsf{S}(2))$ to predict action $\mathsf{A}(2)$. Suppose our prediction model selects object o_8 and assign it to annotators w_1, w_3 and w_5. The environment E infers the label for o_8, e.g., '$negative$', and compute $r_{cost}(2) = 1+1+5 = 7$. We train a classifier and label o_2 as '`positive`' with high confidence. We represent such state in Table II. The costs and qualities of the 5 annotators are also shown in Table II. The reward of enrichment is $r_\phi(2) = \frac{1}{4}$. Then, $\mathsf{S}(2)$ would be updated to $\mathsf{S}(3)$ and the third iteration of labelling starts.

C. Modeling of Markov Decision Process

Additionally, we should clarify that our model can be formalized as a Markov Decision Process (MDP), i.e., current state is only determined by the state in the last iteration. Otherwise, we cannot apply iteration algorithms for updating the policy of `State` $\to$ `Action` to solve an RL problem.

Considering the t-th iteration, CrowdRL takes `Action` $\mathsf{A}(t)$ based on current `State` $\mathsf{S}(t)$, then `Environment` gives feedback – `Reward` $\mathsf{R}(t)$ to `Agent` and enriches the labelled set, and then `State` $\mathsf{S}(t)$ is updated to $\mathsf{S}(t+1)$.

In our labelling model, the `States`, `Actions` and `Rewards` are sequentially updated,

$$\{\mathsf{S}(1), \mathsf{A}(1), r(1), \mathsf{S}(2), \mathsf{A}(2), r(2), \dots\}$$

After each iteration, $\mathsf{A}(t)$ is updated to $\mathsf{A}(t+1)$. We have:

$$p(\mathsf{S}(t)) = p(\mathsf{S}(t) \mid \mathsf{S}(1), \mathsf{A}(1), \dots \mathsf{S}(t-1), \mathsf{A}(t-1)) \tag{2}$$

where $t \geq 2$ and $p(\cdot)$ is the probability function. When $\mathsf{S} = \mathsf{S}(t)$, we take $\mathsf{A}(t)$ of task selection and assignment, then `Environment` gives feedback to the `Agent`, i.e., provides labels from annotators and rewards. Then the `State` is

Algorithm 1: Workflow of CrowdRL

Input: A Set of Unlabelled Objects $\mathbb{O}$, budget B.
Output: Labels of Objects in $\mathbb{O}$.

Initialize State S(t), t=0;
Sampling $\alpha \in (0, 1)$ portion of the objects and ask annotators to label them;
while *Some objects are unlabelled or running out B* **do**
  // Labelled set Enrichment
  Train classifier ϕ using labelled data;
  Rating each unlabelled object o_i using ϕ;
  // $\phi_{c_j}(o_i)$ denotes the probability of $y_i = c_j$ given by ϕ
  for *each o_i is not labelled* **do**
    Find j, k such that $\phi_{c_j}(o_i) \geq \phi_{c_k}(o_i) \geq \phi_{c_l}(o_i)$, for any $c_l \in \mathbb{C}$ and $l \neq j, k$;
    if $\phi_{c_j}(o_i) - \phi_{c_k}(o_i) \leq \delta$ **then**
      o_i remains to be unlabelled;
    else
      $y_i \leftarrow \arg\max_{c_j}\{\phi_{c_j}(o_i)\}$
  Update $\mathsf{S}(t)$;
  // Task selection and assignment
  Conduct $\mathsf{A}(t) = f(\mathsf{S}(t))$;
  // Truth Inference
  Inferring the true labels for selected objects using both annotators and classifier ϕ;
  $t \leftarrow t+1$;

updated into $\mathsf{S}(t+1)$. $\mathsf{S}(t+1)$ is only determined by $\mathsf{S}(t)$ and $\mathsf{A}(t)$. Thus Equation 2 can be rewritten as

$$p(\mathsf{S}(t)) = p(\mathsf{S}(t) \mid \mathsf{S}(t-1), \mathsf{A}(t-1)) \tag{3}$$

where $t \geq 2$. Suppose there is a function $f : \mathsf{S}(t) \to \mathsf{A}(t)$, for each $\mathsf{S}(t)$, $f(\mathsf{S}(t))$ could give the best policy of $\mathsf{A}(t)$. Thus Equation 3 can be rewritten as

$$p(\mathsf{S}(t)) = p(\mathsf{S}(t) \mid \mathsf{S}(t-1),\ f(\mathsf{S}(t-1))) = p(\mathsf{S}(t) \mid \mathsf{S}(t-1))$$

As each o_i is labelled by $|\hat{\mathbf{y}}_i|$ annotators, once $\mathsf{A}(t)$ is determined by $\mathsf{S}(t)$, as the labelling processes of different annotators are independent from each other, the above equation could be rewritten as:

$$p(\mathsf{S}(t)) = p(\hat{\mathbf{y}}_i \mid y_i; \mathsf{S}(t-1)) = \prod_{\hat{y}_i^w \in \hat{\mathbf{y}}_i} p(\hat{y}_i^w \mid y_i; \Pi, \mathsf{S}(t-1))$$

where Π is the confusion matrix of worker w. The state $\mathsf{A}(t)$ is observable in each iteration and the confusion matrix of each annotator is stable. Thus the probability distribution from $\mathsf{S}(t-1)$ to $\mathsf{S}(t)$ are determined. Obviously, our labelling framework is a Markov Decision Process (MDP). Thus, a reinforcement learning algorithm for solving the optimal sequential actions could be applied.

IV. Agent: Unifed Task Selection and Assignment

The `Agent` in CrowdRL analyzes the historical process and maintains this information into `States` $\mathsf{S}(t)$, then it makes an `Action` to select objects and assigns objects to appropriate annotators based on the `Reward`, i.e., using a policy $f : \mathsf{S}(t) \to \mathsf{A}(t)$. We study how to model policy f

and compute f which predicts the optimal action $\mathsf{A}^*(t)$ from the `Agent` perspective as shown in the left part of Figure 2.

A. Modeling of Policy

A naive method to conduct $\mathsf{A}(t)$ based on $\mathsf{S}(t)$ is enumerating all of the possible sequences of actions and selecting the optimal action sequence $\{\mathsf{A}^*(t), \mathsf{A}^*(t+1), \dots\}$. Based on the Bellman Equation [31], if this sequence is optimal, the action $\mathsf{A}^*(t)$ must be the optimal action which could lead to a maximal long-term reward. Thus we apply $\mathsf{A}^*(t)$ which could lead to the maximum long-term reward $\mathsf{R}(t) = \sum_{T=t}^{+\infty} \gamma^{T-1} r(T)$. Formally, we aim to find an ideal f where $\mathsf{A}^*(t) = f(\mathsf{S}(t))$.

We use $Q(\mathsf{S}(t), \mathsf{A}(t))$ to describe the value function (long-term reward) of taking an action $\mathsf{A}(t)$ on $\mathsf{S}(t)$, i.e., using Q table [25], [30], [31] to describe the distribution of f. Formally, given $\mathsf{S}(t)$, we aim at finding an action $\mathsf{A}(t)$ which could maximize the long-term reward:

$$\begin{aligned} Q^*(\mathsf{S}(t), \mathsf{A}(t)) = \mathbb{E}_{\mathsf{S}(t+1)}[r(t)+ \\ \gamma \max_{\mathsf{A}'} Q^*(\mathsf{S}(t+1), \mathsf{A}^{'})|\mathsf{S}(t), \mathsf{A}(t)] \end{aligned} \tag{4}$$

We iteratively learn the value $Q(\mathsf{S}(t), \mathsf{A}(t))$ by conducting optimal action $\mathsf{A}^*(t)$ where $Q(\mathsf{S}(t), \mathsf{A}^*(t))$ is maximal, i.e., $\mathsf{A}^*(t) = \arg\max_{\mathsf{A}'} Q(\mathsf{S}(t), \mathsf{A}')$, and then update the Q-function, i.e., update the policy:

$$\begin{aligned} Q(\mathsf{S}(t), \mathsf{A}(t)) \leftarrow (1-\beta)Q(\mathsf{S}(t), \mathsf{A}(t)) + \beta(r(t) \\ + \gamma \max_{\mathsf{A}'} Q(\mathsf{S}(t+1), \mathsf{A}')) \end{aligned} \tag{5}$$

where $\beta \in [0, 1]$ is the learning rate.

However, it is impractical because the state space and action space are too big (recall that the scale of state space is $(|\mathbb{C}| + 1)^{|\mathbb{O}||\mathbb{W}|}$), and we cannot enumerate all of the possibilities. Besides, we could not predict all of the cases in the future. Hence, we introduce Deep Q-Network (DQN) [25], [34] to describe such relation, i.e., model $f(\cdot)$ as a deep neural network in place of a Q-table. We approximate Q value as $Q^*(\mathsf{S}(t), \mathsf{A}(t)) \approx Q(\mathsf{S}(t), \mathsf{A}(t); \theta)$, where θ is the parameter set of the Q-network as shown in the `Agent` part of Figure 2. Comparing with iteratively updating each value in the Q-table, we iteratively update the parameter θ to find the optimal policy by solving the problem of optimization of the loss function:

$$\begin{aligned} L(\theta) =& \mathbb{E}_{\{(\mathsf{S}(t), \mathsf{A}(t), r(t), \mathsf{S}(t+1))\}}[(r(t) + \gamma \max_{\mathsf{A}(t+1)} Q^*(\mathsf{S}(t+1), \\ & \mathsf{A}(t+1), \theta) - Q(\mathsf{S}(t), \mathsf{A}(t); \theta))^2] \end{aligned}$$

The last issue is to feed training data to iteratively learn the optimal solution of $L(\theta)$. Inspired by the classical deep Q-network [25], which indicates that human make decision by referring part of the historical experience, we use the strategy of experience replay i.e., sampling training data from historical experience pool $\{(\mathsf{S}(t), \mathsf{A}(t), r(t), \mathsf{S}(t+1))\}$ (See Figure 2).

B. Optimal Action Selection

For each time we select the action, a naive way [25] is using a greedy method $\mathsf{A}(t) = \arg\max_{\mathsf{A}'} Q(\mathsf{S}(t), \mathsf{A}')$. However, it may lead to local optimization rather global optimization without any '*exploration*' for better choices, as the Q-value of a current optimal action may become bigger and bigger if it is repeatedly selected. We propose a dynamic action selection policy inspired by UCB1 algorithm [2]. We select action

$$\mathsf{A}(t) = \arg\max_{\mathsf{A}'}[Q(\mathsf{S}(t), \mathsf{A}') + \sqrt{\frac{2ln(n')}{n}}] \tag{6}$$

where n represents the times of choosing action A' for state $\mathsf{S}(t)$ and n' denotes the total times of updating Q value for state $\mathsf{S}(t)$. It combines both the '*exploration*' and '*greedy*' strategies. If an action A' is selected for too many times, the term $\sqrt{\frac{2ln(n')}{n}}$ will decrease and A' will be less likely to be selected. If $Q(\mathsf{S}(t), \mathsf{A}')$ increases, action A' will be more likely to be selected.

We set the value of $Q(\mathsf{S}, \mathsf{A}) = -\infty$ when A refers to labelling o_j, and o_j has been labelled in previous iterations, in case of duplicated labelling. In our value iteration process, these Q values would retain to be $-\infty$ if we initially set it as $-\infty$. It helps us filter invalid operations of TS from the output of the neural network. For example, in the Table II and III, if o_1 has been labelled, we will set $Q(\mathsf{S}(2), \mathsf{A}(2)) = -\infty$ if $\mathsf{A}(2) = (1, j)$ where $j \in \{1, 2, 3, 4, 5\}$, i.e., each element in the first column of Table III is $-\infty$.

In this paper, we follow the classical design of DQN [25]. Note that other variants of DQN [13], [38] can also be integrated into our framework.

Discussion. We need to assign multiple tasks to many annotators in each iteration of labelling, rather than assign an object to one annotator. Suppose we aim to employ k annotators for each object, we compute the `Top-k` Q values for each object and compute the summation of these k values. Then we select objects with the largest summation of these `Top-k` Q values to be labelled by using a "MinHeap" algorithm [1].

Example 3: In the second iteration of labelling in Figure 1, objects o_1, o_4 and o_5 are labelled and we select an optimal $\mathsf{A}(2)$ based on $\mathsf{S}(2)$. We show all of the values $Q(\mathsf{S}(2), \mathsf{A})$ in Table III for all the possibilities of action A. The sign of '$\times$' denotes $-\infty$ here, thus we could not select o_1, o_4 and o_5 again. The summation of the `Top-3` Q values of o_8 is 9, which is the biggest. Thus we select o_8 and assign it to w_1, w_3 and w_5.

V. Environment: Joint Truth Inference

Recall that the `Environment` part infers the true labels of given objects, retrains the classifier to enrich the label set and gives feedback including `Reward` to the `Agent` part. From the perspective of environment, we propose a novel truth inference method which integrates the annotator model and classifier model into a unified truth inference algorithm as shown in the right part in Figure 2. Then we demonstrate the process of labelled set enrichment.

A. CrowdRL Truth Inference Model

1) Basic Idea of CrowdRL Truth Inference Model: Given labels set $\hat{\mathbf{y}}_i$ from annotators for o_i, traditional truth inference algorithms mainly use a majority voting (MV) strategy [37], i.e., assign the label given by majority of the annotators to o_i, or expectation maximization (EM) algorithm [48], which computes the weighted summation of labelling answers by iteratively updating the qualities of annotators and the true labels, to infer the truth. For example, for the label answers of o_1 in Table II, the answer set is {'positive', 'negative', 'positive'}, using a majority voting strategy, the inferred label is 'positive'.

Note that using labels from annotators only may cause bias when annotators make mistakes in some cases. For example, when labelling a tumor in a medical image, if five medical students are employed and two of them mis-label it as 'negative' while the others correctly label it as 'positive', we cannot give a prediction with a high confidence. However, if a tumor recognition algorithm trained by history data classifies it as 'positive', combined with the answers from annotators, we can label it as 'positive' with a high confidence. Thus, using the prediction results from classifier can improve inference quality.

Intuitively, since we have labelled many objects and train a classifier for them, we could consider how to use not only the labels from annotators but also the prediction from the classifier. Notice that the classifier ϕ is trained by labelled examples, which are gotten from previous labelling iteration. In a sense, we reuse the human labors of labelling in previous labelling iterations. A naive method is regarding classifier ϕ as a special 'annotator', which could give an label $\hat{y}_i^c = c_j$, where $c_j \in \mathbb{C}$, for object o_i with a confidence $\phi_{c_j}(o_i)$. In Figure 3(a), workers, experts and the classifier are regards as annotators with different quality, and the true label y_i would be inferred by the answers from $\hat{\mathbf{y}}_i$ and $\hat{y}_i^c$ using MV or EM algorithm. For example, In Figure 1(c) and Table II, $\hat{\mathbf{y}}_8 =$ {'positive', 'negative' and 'negative'} respectively, suppose $\hat{y}_8^c =$'negative', we could give a confident inference for $y_8 =$'negative' using MV.

However, since the classifier ϕ is trained by labelled data with noises, these noises are caused by annotators with known biases. Besides, the learning algorithm of training ϕ would bring additional biases, thus the biases of $\phi_{c_j}(o_i)$ is composite, such biases are hard to model and using such a label to infer the truth is not reasonable.

Thus, we propose to jointly model the unknown distribution of the expertises of annotators and the classifier. Based on currently seen labelled objects, we make joint inference for the parameters of the classifier, the expertises of annotators and the true labels. Then the biases caused by the classifier and annotators would be easily to be modeled. Since no composite biases are introduced, the inference will be more accurate.

2) Learning Algorithm for CrowdRL Truth Inference: Formally, we use a classifier ϕ for multi-class classification task with parameter Θ. For each object o_i, ϕ could give a inference $\hat{y}_i^c$. Since we do not know the true label y_i, thus we regard y_i as a latent value. We can only infer y_i based on answers from annotators $\hat{\mathbf{y}}_i$ and $\hat{y}_i^c$. Recall that we denote the confusion matrix Π^j to denote the expertise of the j-th annotator. For j-th annotator, the label $\hat{y}_i^{w_j}$ is the noisy

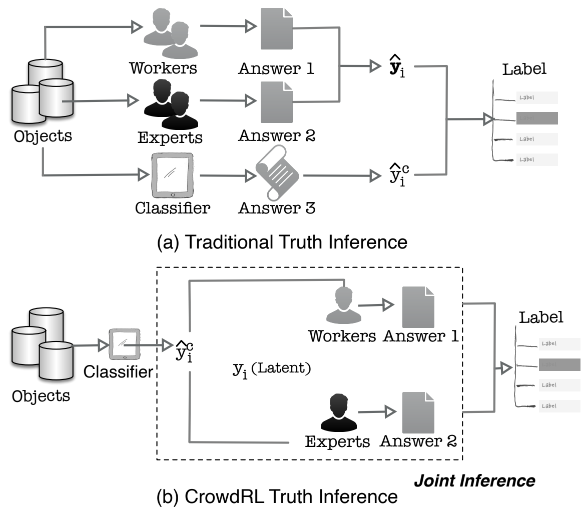


Fig. 3: Truth Inference: Traditional Methods V.S. CrowdRL. (a) Traditional methods simply integrate workers' answers, experts' answers and classifier's answers by computing a weighted summation of them; (b) CrowdRL jointly model the answers of annotators and the classifier, to make joint truth inference with a maximum probability, which is more accurate.

version of $\hat{y}_i$. Since the labelling result of different annotators are independent, the whole likelihood of collected labels from annotators is:

$$p(\mathbb{L}|\Theta, \{\Pi^j\}) = \prod_{i=1}^{|\mathbb{L}|}[p(y_i|\phi,\Theta)\prod_{j=1}^{|\mathbb{W}|}p(\hat{y}_i^{w_j}|y_i,\Pi^j)] \quad (7)$$

where $\mathbb{L}$ denotes all the training data labelled by annotators. We aim at finding the truth of objects which causes to the observed labels in $\mathbb{L}$ with maximum expectation, for ease of solving the solution of Equation 7, we rewrite the formation as log-expectation:

$$\begin{aligned}&\mathbb{E}[ln(p(\mathbb{L}|\Theta,\Pi^1,\ldots,\Pi^{|\mathbb{W}|}))]\\ =&\sum_{i=1}^{|\mathbb{L}|}\sum_{y_i} q(y_i) ln(p(y_i|\phi,\Theta)\prod_{j=1}^{|\mathbb{W}|}p(\hat{y}_i^{w_j}|y_i,\Pi^j))\end{aligned} \quad (8)$$

where $q(y_i)$ is the posterior which could be obtained by using estimated parameters in the last labelling iteration:

$$q(y_i=c) \propto p(y_i=c \mid \phi_{last},\Theta_{last})\prod_{j=1}^{|\mathbb{W}|}p(y_i^{w_j} \mid y_i=c,\Pi^j_{last})$$

where Π^r_{last} and Θ_{last} are parameters in previous iterations.

For finding a maximum value of Equation 8, we could iteratively update the value of Θ and each Π^j meanwhile. Finally, we find the maximum likelihood for the model parameters. Normally, the annotator expertise is given by:

$$\pi^j_{c,d} = \frac{\sum_{i=1}^{|\mathbb{L}|} q(y_i=c)\mathbb{I}(\hat{y}_i^{w_j}=d)}{\sum_{i=1}^{|\mathbb{L}|} q(y_i=c)}$$

where

$$\mathbb{I}(\hat{y}_i^{w_j} = d) = \begin{cases} 1 & \hat{y}_i^{w_j} = d \\ 0 & otherwise \neq c \end{cases}$$

For heterogeneous annotators, we believe that experts are more confident than crowdsourcing workers. Previous methods regard workers and experts as annotators with different qualities [29], [39], however, may decrease the confidence of an expert during the running of an EM-style algorithm. For example, if an expert makes mistakes, his/her quality may be updated to be lower and finally s/he would not be confident. Thus our model designs the mechanism of bounding the quality of experts. For class c_i, if the $\pi_{i,i}^j < \epsilon$ and annotator w_j is an expert, we set

$$\hat{\pi}_{c,l}^j = \begin{cases} \epsilon & l = c \\ (1-\epsilon)\frac{\pi_{c,l}^j}{(\sum_{k=1}^{|\mathbb{C}|} \pi_{c,k}^j) - \pi_{c,l}^j} & l \neq c \end{cases}$$

where ϵ is a threshold.

Our truth inference algorithm is easy to be integrated into our CrowdRL framework and converge to a numerical solution of both annotators' quality and the true labels. The trinity of machine learning model inference, answers from workers and experts are integrated for inferring the truth. The confidence of experts are also bounded with our framework. See Figure 3(b).

B. CrowdRL Labelled Set Enrichment

After truth inference, the labelled data set has changed and since we retrain a classifier ϕ using labelled data, we will use ϕ to label some objects with high confidence. We rate each unlabelled object o_i using ϕ, i.e., $\phi_{c_j}(o_i) = p(y_i = c_j|\phi)$. If there exists $|\phi_{c_j}(o_i) - \phi_{c_k}(o_i)| \leq \delta$, the true label of o_i is ambiguous, thus o_i remains to be unlabelled. Otherwise, we label $y_i = \arg\max_{c_j}\{\phi_{c_j}(o_i)\}$. Then the labelled set is enriched as shown in Figure 2. The pseudo code of the labelled set enrichment is shown in Algorithm 1.

For example, after inferring o_8 as '`positive`' in Example 3, suppose the trained classifier labels o_2 as '`positive`' with a confidence of 0.9 and '`negative`' with a confidence of 0.1. The classifier predict o_3 as '`positive`' with a confidence of 0.55 and '`negative`' with a confidence of 0.45. If we set $\epsilon = 0.2$, $|\phi_{positive}(o_2) - \phi_{negative}(o_2)| = 0.9 - 0.1 = 0.8 > 0.2$, thus we can label y_2='`positive`' with high confidence and then the labelled set is enriched. Since $|\phi_{positive}(o_3) - \phi_{negative}(o_3)| = 0.55 - 0.45 = 0.1 < 0.2$, we can not label o_3 using the classifier in this iteration.

VI. Experiments

A. Experimental Methodology

1) Datasets: We used three real-world datasets including two datasets of video clips and one image dataset.

Speech12 and Speech3 [41] were collected by the biggest online education company – TAL[2]. The datasets contained many video clips of oral presentations from pupils in Chinese. In each video, a pupil was asked to talk about his or her thinking process of solving a mathematical question. The task was to assess the student's oral expression ability and label each video as either '`positive`' (excellent presentation) or '`negative`' (awful presentation). The answers were provided by both professional teachers in TAL or workers from TAL crowdsourcing marketplace. The ground truths were generated by integrating answers from five professional teachers in TAL using majority voting. Speech12 contained 2344 videos from the first and second grades and Speech3 contained 1898 videos from the third grade in a primary school, which were thought to have different abilities of expression.

For video clips, intuitively, the prosodic characters and the text of the speeches could represent the motions and contents, thus we extracted two types of features, contextual features (such as statistics of part-of-speech tags [12], number of consecutive duplicated words and number of interregnum words) encoded in 50-dimension float vector and prosodic features (such as signal energy, loudness, voice speed and silence duration percentage) encoded in 1582-dimension float vector. We denoted the contextual, prosodic and concatenated features of the two types of features as S12C, S12P, S12CP, S3C, S3P and S3CP respectively.

Fashion [22] was a social image dataset for fashion and clothing. Each image was published as a question for identifying whether or not it is fashion-related. There were totally 32,398 questions and each question was answered by 3 annotators.

2) Baselines: In this paper, we focused on building an end-to-end labeling framework. Thus, we selected four end-to-end labeling frameworks as baselines.

DLTA [46]. In this framework, the labeling process was divided into multiple iterations. Each iteration consisted of two steps, label inference and label acquisition. In the label inference step, it used an EM (Expectation-Maximization) algorithm to complete the process of answer aggregation. In the label acquisition step, given the budget, it selected proper objects for labeling to maximize the benefits.

OBA [15]. It trained a model based on the labelled data as "AI workers" (e.g., it used traditional classification or clustering methods, e.g., KNN). It used a human-in-the-loop process to label the data. In each labelling iteration, the human workers first labeled some objects and the labelled set would be updated. Then, the "AI Worker" predicted the labels for all of the unlabelled objects. For each object, if the confidence of the prediction was higher than a threshold, it would be labelled, otherwise it would be assigned to human workers in the following iterations. It assumed that the human worker could always give true labels.

IDLE [16]. It was an end-to-end multi-level classification framework. On the first level, it collected cost-effective truth inference from crowdsourcing workers whose answers have potentially high bias and variance. On the second level, experts provided confident answers. For ambiguous cases, the objects would be labeled as "unsolvable". The task selection process was random, and it used EM algorithms for truth inference.

[2]http://www.100tal.com/

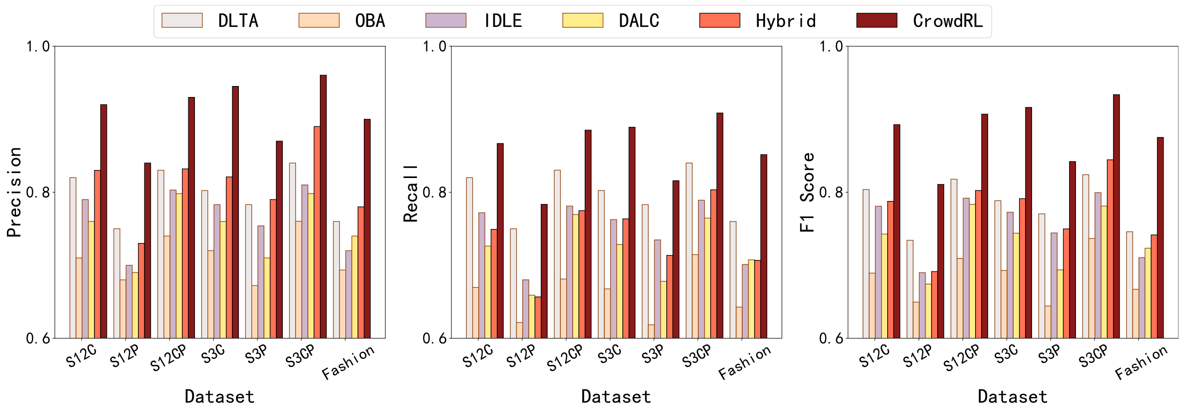


Fig. 4: Labelling Quality with the Same Budget.

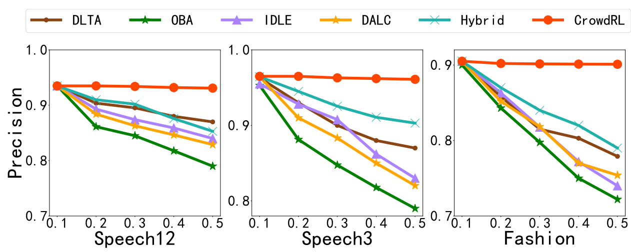


Fig. 5: Scalability (Sampling the number of objects with ratio of 0.1, 0.2, 0.3, 0.4, 0.5).

DALC [42]. It provided a unified Bayesian model to infer the true labels and parameters of the classification model to reach an optimal learning efficiency simultaneously. In each labeling iteration, it selected some most informative tasks and the annotators with the highest expertise for these tasks.

Hybrid. Additionally, we constructed a hybrid human-in-the-loop framework as a baseline. In each labeling iteration, it used a MinExpError algorithm [26] based on the method of bootstrap, which selected the object whose labels from annotators were different from the label predicted by current classifier with the maximum probability. It used a DQN for task assignment as used in [32], which outperformed most task assignment or arrangement algorithms. For truth inference, it used a PM algorithm [48] by iteratively updating the annotators' qualities and the estimated label for an object until both of them converged.

3) Metrics.: We focused on three metrics: (1) Precision ($Prec$), (2) Recall (Rec), (3) F1 Score ($F1$).

4) Setting.: We implemented all the algorithms on a Macbook pro with 2.4 GHz Intel Core i5 CPU and 16GB RAM with Intel Iris Plus Graphics 655 1536MB. We used Pytorch 1.6.0 and CUDA 10.1 for our machine learning framework.

As our algorithm was an off-policy reinforcement learning framework, we trained our deep Q-network offline. In our experiment, we used a "cross training methodology", i.e., when evaluating one dataset online, we used the other datasets to train the reinforcement learning model offline in advance. We used a fully connected neural network with a sigmoid output layer as the classifier ϕ.

B. Experiment Results

1) Comparing with Different Methods: We evaluated CrowdRL by comparing it with different baselines. The default setting was $\alpha = 5\%$ and $|\mathbb{W}| = 5$ for SC12 and SC3, $|\mathbb{W}| = 3$ for Fashion where α was the initial sampling rate, the costs of employing one worker and expert were 1 unit and 10 units of budget respectively. We set the labelling budget as 10000 units for both SC12 and SC3. We set the budget of Fashion as 160000 units. The results were shown in Figure 4.

We had the following observations: (1) CrowdRL outperformed baselines because CrowdRL used a unified framework to combine the processes of TS, TA and TI, and thus could give more reasonable arrangement for the labelling process. Thus, the labelling quality of CrowdRL would be higher than baselines; (2) **OBA** performed the worst because it assumed that human annotators always gave the true labels, which was not practical in most cases; (3) CrowdRL gave higher accuracy than baselines by 5%-20% on the 6 cases for speech recognition tasks. It indicated that CrowdRL outperformed baselines for hard tasks. This was because it considered not only the labels from annotators but also the features of objects; (4) **IDLE** performed worse than **DLTA**, because **IDLE** randomly assigned the tasks to annotators, thus it did not consider to select the best objects to label which could decrease the uncertainty or bias of labelled data, and more labelled objects would be required. Besides, the cost of training domain experts for **IDLE** was big; (5) Labelling the two speech recognition datasets with concatenated features, i.e., S12CP and S3CP, was more effective than using either contextual or prosodic features only. This was because the classifier for higher vector space could give better prediction; (6) **Hybrid** performed better than the other four baselines most of the time, because bootstrap uncertainty could highly reduce the cost in an AL labelling process, which used weighted sampling techniques as its selection methods. Besides, **Hybrid** considered the annotators' expertises as probability distribution, and thus could arrange more proper annotators for a task; (7) CrowdRL performed better on speech recognition than image classification because we made better jobs on feature extraction for the two speech recognition datasets.

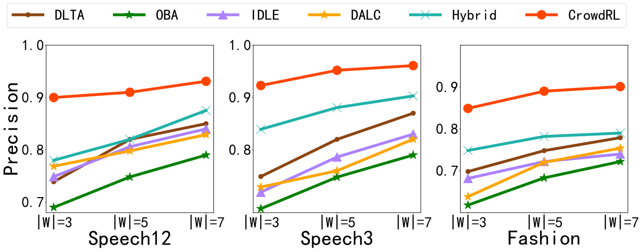


Fig. 6: Varying $|\mathbb{W}|$.

Summary. With the same monetary cost, CrowdRL outperformed existing algorithms by 5-20% higher quality, because CrowdRL integrated TS, TA and TI together and used both human labors and learned models to infer the truth.

2) Varying Parameters: We evaluated CrowdRL with the five baselines by varying the parameters in the labelling process. For the cases of video classification, we used concatenated features, i.e., S12CP and S3CP. The default setting was $\alpha = 5\%$ (initial sampling rate), $|\mathbb{W}| = 5$ for SC12 and SC3, $|\mathbb{W}| = 3$ for Fashion. We set the budget as 10000 units for SC12 and SC3, 160000 units for Fashion.

Scalability. We evaluated the scalability by using $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ of the datasets. The precision of the 3 datasets were shown in Figure 5. We had the following observations: (1) With the increase of dataset scale, CrowdRL converged to a high precision though the budget was limited. However, the precisions of baselines decreased as the data scale increased. This was because CrowdRL could find the optimal strategy for task selection and assignment, thus it could find a small portion of the dataset to label with limited budget, whose distribution was similar to the population; (2) With the increase of data scale, all the methods achieved lower precision. This was because if the dataset was too small, we needed to label nearly all of the objects. However, if the dataset was too big, we might just need to label a small portion; (3) The two datasets of speech recognition were more sensitive for the increase of data scale, because it was more difficult to answer and needed more labels for training accurate classifiers.

Varying $|\mathbb{W}|$. We varied $|\mathbb{W}|$ as $\{3, 5, 7\}$ for the three datasets. The experimental results were shown in Figure 6. We had following observations: (1) CrowdRL outperformed baselines at each settings of the number of annotators because CrowdRL used the trinity of expert, worker and machine algorithm to increase the labelling accuracy, thus the accuracy of labelled data was high. Baselines might need many experts to label the examples thus the costs were higher; (2) Baselines were more sensitive for the increase of annotators' number because they were not accurate when the annotators were not sufficient. However, CrowdRL had reached nearly the highest accuracy and thus has limited space for improvement; (3) The Fashion dataset was not very sensitive to the increase of number of annotators but the video datasets were very sensitive. It was because the task of labelling an object as "Fashion-related" or not was easier than labelling an oral report of solving a mathematical problem.

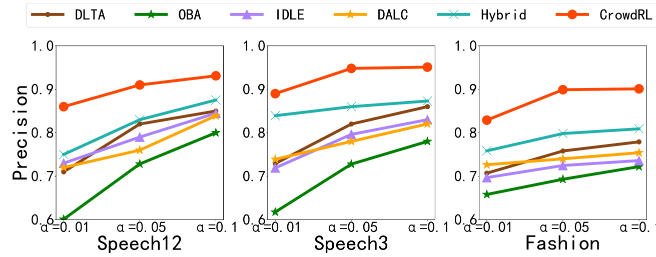


Fig. 7: Varying α.

Varying α. We set the initial sampling rate as 0.01, 0.05 and 0.1. The experimental results were shown in Figure 7 We observed the changing of precision of the three datasets. We had following observations. (1) CrowdRL outperformed baselines especially when α was small, because CrowdRL could use few labelled objects to infer the truth of unlabelled objects with the help of our end-to-end framework. (2) When α was big enough, all of the methods were not sensitive to the change of α, because all of the human-in-the-loop methods just needed to label parts of the objects.

Summary. With the same cost, CrowdRL outperformed existing end-to-end labeling frameworks, e.g., 5-20% higher quality. CrowdRL could use a limited budget to achieve a higher labelling performance in different settings.

3) Ablation Experiment: We evaluated the effect of each component of CrowdRL, i.e., comparing CrowdRL by not using each of our three main techniques: task selection, task assignment and join inference model. Let 'M1' denoted the method without using our task selection method (using random task selection), 'M2' denoted the method without using our task assignment method (using random assignment), and 'M3' denoted the model without using our joint inference method (using PM algorithm [48] as inference model). We compared the precision of these methods with the same budget and setting as discussed in Section VI-B1. The experiment results were shown in Figure 8. We found that each component of CrowdRL could effect the performance. 'M1' and 'M2' performed better than 'M3' in the datasets of Speech3 and Fashion, because it was more important to model the task assignment and selection as a unified operation.

VII. Conclusion

We propose CrowdRL, an end-to-end reinforcement learning framework for labelling datasets using heterogeneous annotators (experts and workers) with limited budget. We integrate task selection, task assignment, and truth inference together, which can judiciously assign tasks to appropriate workers and infers the truths based on answers from workers, experts and classifiers. In each iteration, we use a learned deep Q-network to make decision for task selection and assignment, then we infer the true labels of these objects by using an expectation maximization algorithm. CrowdRL considers the relation between objects and annotators and combine the task assignment and selection as a unified operation. Experimental results show that CrowdRL outperforms baselines by 5%-20% higher accuracy while keeping the same monetary cost.

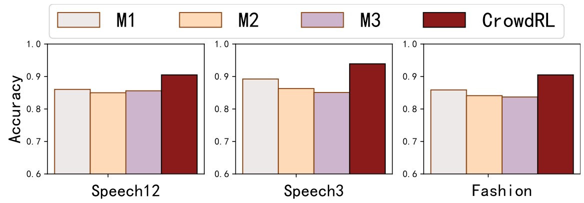


Fig. 8: Ablation Experiment for Verifying the Effect of Each Component (M1: without our task selection; M2: without our task assignment; M3: without our joint inference).

REFERENCES

[1] M. D. Atkinson, J. Sack, N. Santoro, and T. Strothotte. Min-max heaps and generalized priority queues. *Commun. ACM*, 29(10):996–1000, 1986.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002.

[3] B. I. Aydin, Y. S. Yilmaz, Y. Li, Q. Li, J. Gao, and M. Demirbas. Crowdsourcing for multiple-choice question answering. In *AAAI*, pages 2946–2953, 2014.

[4] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *ICML*, volume 382 of *ACM*, pages 49–56. ACM, 2009.

[5] E. Bringer, A. Israeli, Y. Shoham, A. Ratner, and C. Ré. Osprey: Weak supervision of imbalanced extraction problems without code. In *SIGMOD*, pages 4:1–4:11. ACM, 2019.

[6] C. Chai, J. Fan, G. Li, J. Wang, and Y. Zheng. Crowdsourcing database systems: Overview and challenges. In *ICDE*, pages 2052–2055. IEEE, 2019.

[7] C. Chai, G. Li, J. Li, D. Deng, and J. Feng. A partial-order-based framework for cost-effective crowdsourced entity resolution. *VLDB J.*, 27(6):745–770, 2018.

[8] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. In *NIPS*, pages 705–712, 1994.

[9] N. Das, S. Chaba, R. Wu, S. Gandhi, D. H. Chau, and X. Chu. GOGGLES: automatic image labeling with affinity coding. In *SIGMOD*, pages 1717–1732. ACM, 2020.

[10] Z. Gharibshah, X. Zhu, A. Hainline, and M. Conway. Deep learning for user interest and response prediction in online display advertising. *Data Science and Engineering*, 5(1):12–26, 2020.

[11] D. Görges. Relations between model predictive control and reinforcement learning. *IFAC-PapersOnLine*, 50(1):4920–4928, 2017.

[12] T. Gui, Q. Zhang, H. Huang, M. Peng, and X. Huang. Part-of-speech tagging for twitter with adversarial neural networks. In *EMNLP*, pages 2411–2420. Association for Computational Linguistics, 2017.

[13] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. G. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, pages 3215–3222. AAAI Press, 2018.

[14] H. K. Khalil and J. W. Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[15] M. Kobayashi, K. Wakabayashi, and A. Morishima. Quality-aware dynamic task assignment in human+ai crowd. In *WWW*, pages 118–119. ACM / IW3C2, 2020.

[16] W. Lee, C. Chang, P. Yang, C. Huang, M. Wu, C. Hsieh, and K. Chuang. Effective quality assurance for data labels through crowdsourcing and domain expert collaboration. In *EDBT*, pages 646–649. OpenProceedings.org, 2018.

[17] G. Li, J. Wang, Y. Zheng, and M. J. Franklin. Crowdsourced data management: A survey. *IEEE Trans. Knowl. Data Eng.*, 28(9):2296–2319, 2016.

[18] G. Li, Y. Zheng, J. Fan, J. Wang, and R. Cheng. Crowdsourced data management: Overview and challenges. In *SIGMOD*, pages 1711–1716. ACM, 2017.

[19] K. Li, X. Zhang, and G. Li. A rating-ranking method for crowdsourced top-k computation. In *SIGMOD*, pages 975–990, 2018.

[20] Y. Li, X. Wu, Y. Jin, J. Li, and G. Li. Efficient algorithms for crowd-aided categorization. *VLDB*, 13(8):1221–1233, 2020.

[21] T. Liu, J. Yang, J. Fan, Z. Wei, G. Li, and X. Du. Crowdgame: A game-based crowdsourcing system for cost-effective data labeling. In *SIGMOD*, pages 1957–1960. ACM, 2019.

[22] B. Loni, L. Y. Cheung, M. Riegler, A. Bozzon, L. Gottlieb, and M. A. Larson. Fashion 10000: an enriched social image dataset for fashion and clothing. In *MMSys*, pages 41–46. ACM, 2014.

[23] J. L. Mingda Li, Hongzhi Wang. Mining conditional functional dependency rules on big data. *Big Data Mining and Analytics*, 03(01):68, 2020.

[24] V. Mnih, K. Kavukcuoglu, and et al. Human-level control through deep reinforcement learning. *Nat.*, 518(7540):529–533, 2015.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[26] B. Mozafari, P. Sarkar, M. J. Franklin, M. I. Jordan, and S. Madden. Scaling up crowd-sourcing to very large datasets: A case for active learning. *PVLDB*, 8(2):125–136, 2014.

[27] A. G. Parameswaran, H. Park, H. Garcia-Molina, N. Polyzotis, and J. Widom. Deco: declarative crowdsourcing. In *CIKM*, pages 1203–1212, 2012.

[28] X. Qin, Y. Luo, N. Tang, and G. Li. Deepeye: An automatic big data visualization framework. *Big Data Mining and Analytics*, 1(1):75, 2018.

[29] A. Ratner, S. H. Bach, H. R. Ehrenberg, J. A. Fries, S. Wu, and C. Ré. Snorkel: rapid training data creation with weak supervision. *VLDB J.*, 29(2-3):709–730, 2020.

[30] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. In *ICLR*, 2016.

[31] H. M. Schwartz. *Multi-agent machine learning: A reinforcement approach*. John Wiley & Sons, 2014.

[32] C. Shan, N. Mamoulis, R. Cheng, G. Li, X. Li, and Y. Qian. An end-to-end deep RL framework for task arrangement in crowdsourcing platforms. In *ICDE*, pages 49–60. IEEE, 2020.

[33] C. Shan, N. Mamoulis, G. Li, R. Cheng, Z. Huang, and Y. Zheng. A crowdsourcing framework for collecting tabular data. *IEEE Trans. Knowl. Data Eng.*, 32(11):2060–2074, 2020.

[34] D. Silver, J. Schrittwieser, K. Simonyan, and et al. Mastering the game of go without human knowledge. *Nat.*, 550(7676):354–359, 2017.

[35] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[36] N. Ta, K. Li, Y. Yang, F. Jiao, Z. Tang, and G. Li. Evaluating public anxiety for topic-based communities in social networks. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2020.

[37] T. Tian and J. Zhu. Max-margin majority voting for learning from crowds. In *NIPS*, pages 1621–1629, 2015.

[38] H. van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, pages 2094–2100. AAAI Press, 2016.

[39] P. Varma and C. Ré. Snuba: Automating weak supervision to label training data. *PVLDB*, 12(3):223–236, 2018.

[40] J. Wang, T. Kraska, M. J. Franklin, and J. Feng. Crowder: Crowdsourcing entity resolution. *VLDB.*, 5(11):1483–1494, 2012.

[41] G. Xu, W. Ding, J. Tang, S. Yang, G. Y. Huang, and Z. Liu. Learning effective embeddings from crowdsourced labels: An educational case study. In *ICDE*, pages 1922–1927. IEEE, 2019.

[42] J. Yang, T. Drake, A. C. Damianou, and Y. Maarek. Leveraging crowdsourcing data for deep active learning an application: Learning intents in alexa. In *WWW*, pages 23–32. ACM, 2018.

[43] X. Yu, G. Li, Y. Zheng, Y. Huang, S. Zhang, and F. Chen. Crowdota: An online task assignment system in crowdsourcing. In *ICDE*, pages 1629–1632. IEEE Computer Society, 2018.

[44] F. M. Zanzotto. Viewpoint: Human-in-the-loop artificial intelligence. *J. Artif. Intell. Res.*, 64:243–252, 2019.

[45] X. Zhang, G. Li, and J. Feng. Crowdsourced top-k algorithms: An experimental evaluation. *VLDB.*, 9(8):612–623, 2016.

[46] L. Zheng and L. Chen. DLTA: A framework for dynamic crowdsourcing classification tasks. *IEEE Trans. Knowl. Data Eng.*, 31(5):867–879, 2019.

[47] Y. Zheng, G. Li, and R. Cheng. DOCS: domain-aware crowdsourcing system. *VLDB*, 10(4):361–372, 2016.

[48] Y. Zheng, G. Li, Y. Li, C. Shan, and R. Cheng. Truth inference in crowdsourcing: Is the problem solved? *PVLDB*, 10(5):541–552, 2017.

[49] Y. Zheng, J. Wang, G. Li, R. Cheng, and J. Feng. QASCA: A quality-aware task assignment system for crowdsourcing applications. In *SIGMOD*, pages 1031–1046, 2015.