Human+AI Crowd Task Assignment Considering Result Quality Requirements

Masaki Kobayashi,[1] **Kei Wakabayashi,** [2] **Atsuyuki Morishima** [3]
University of Tsukuba, Japan
[1] makky@klis.tsukuba.ac.jp, [2] kwakaba@slis.tsukuba.ac.jp, [3] morishima-office@ml.cc.tsukuba.ac.jp

Abstract

This paper addresses the problem of dynamically assigning tasks to a crowd consisting of AI and human workers. Currently, crowdsourcing the creation of AI programs is a common practice. To apply such kinds of AI programs to the set of tasks, we often take the "all-or-nothing" approach that waits for the AI to be good enough. However, this approach may prevent us from exploiting the answers provided by the AI until the process is completed, and also prevents the exploration of different AI candidates. Therefore, integrating the created AI, both with other AIs and human computation, to obtain a more efficient human–AI team is not trivial. In this paper, we propose a method that addresses these issues by adopting a "divide-and-conquer" strategy for AI worker evaluation. Here, the assignment is optimal when the number of task assignments to humans is minimal, as long as the final results satisfy a given quality requirement. This paper presents some theoretical analyses of the proposed method and an extensive set of experiments conducted with open benchmarks and real-world datasets. The results show that the algorithm can assign many more tasks than the baselines to AI when it is difficult for AIs to satisfy the quality requirement for the whole set of tasks. They also show that it can flexibly change the number of tasks assigned to multiple AI workers in accordance with the performance of the available AI workers.

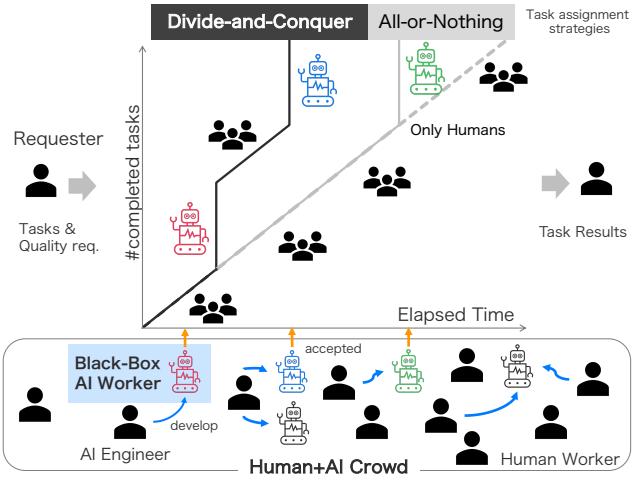

Figure 1: Our proposed method gradually assigns tasks to AI workers in the Human+AI crowd via the "divide-and-conquer" strategy while satisfying the quality requirement. Thus, it reduces human worker efforts better than do "all-or-nothing" strategies, such as active learning-based assignments.

Introduction

Recently, the creation of AI programs has been outsourced using crowdsourcing platforms such as Kaggle[1] and AIcrowd[2]. In this context, volunteers with expertise in IT actively develop software to solve real-world problems, such as natural disaster situations. These platforms provide a means for people unfamiliar with the complexities of AI implementation to enjoy the benefits of AI in processing their tasks.

Meanwhile, crowdsourcing platforms that utilize human workers, such as Amazon Mechanical Turk, can similarly process the tasks. However, there are few insights into how to combine these two types of resources in processing a fixed number of tasks to obtain high-quality data. Our goal is to provide an approach for integrating the power of human workers and AIs in practical settings.

This paper deals with the following problem: Given a set $T = \{t_1, \ldots t_M\}$ of classification tasks (e.g., image labeling) and a quality requirement q, we want to find the optimal assignment of tasks to humans and AIs. We say the assignment is optimal when the number of tasks assigned to humans is minimized (and therefore the number of tasks assigned to AIs is maximized) under the condition that the answers of the workers satisfy the quality requirement q given by the requesters. The task assignment will be decided incrementally, and the tasks completed by humans can be used as training and test datasets for AIs. Therefore, the main concern in this situation is the prioritization of tasks. As studies on active learning have demonstrated, the order of tasks significantly affects the quality of AIs trained using those tasks. Such a setting is especially important when we have limited human resources, or we urgently need to obtain the task results with the help of AI in a domain that may have insuffi-

[1]https://www.kaggle.com

[2]https://www.aicrowd.com

cient training data, such as in natural disaster situations.

A typical approach to prioritize tasks is to apply active learning to decide which tasks should be performed by humans (Zhu et al. 2010). In this method, to guarantee the quality of the AI, we need to assign some random tasks to humans to create a test dataset. When the AI undergoing active learning satisfies the quality requirement q on the test dataset, we can assign all of the remaining tasks to the AI (Figure 1 (Top)). However, this approach has the following shortcomings: (1) The method assigns tasks to AIs by the "all-or-nothing" strategy; consequently, we cannot exploit any of the benefits of the AI before the AI is accepted. (2) The method assumes a single particular active learning ML model; consequently, we cannot leverage many other AIs in the crowd, and such the process is clearly not scalable.

In this paper, we propose a method that addresses these potential issues by adopting a "divide-and-conquer" strategy. The core idea of the proposed method is to evaluate the quality of an AI partially on subsets of tasks that the AI classified into the same class, which we call **task clusters**. The proposed method decides to assign tasks in a task cluster to an AI if the AI satisfies a sufficient quality on the tasks in the task cluster (Figure 1 (Bottom)). Using this strategy, we can decide the assignment of AI-easy tasks to the AIs earlier, meaning that we are able to exploit the AI's answers before the entire set of tasks T is completed (which addresses shortcoming (1) listed above). Additionally, the proposed algorithm does not depend on a single active learning ML model but can take many different *black-box* AIs, which we call **AI workers**[3]. They can join or leave at any time during the assignment process. In the proposed method, the output of the AI workers is transformed into task clusters and evaluated individually. Therefore, the participation of multiple AI workers is handled naturally as a set of task clusters and is used to find a better solution, which addresses shortcoming (2) listed above.

The contributions of this paper are as follows:

(1) Human+AI Crowd Task Assignment Problem: This paper introduces a novel human+AI crowd task assignment problem (HACTAP) that assigns a fixed set of given tasks to human and AI workers. This problem differs from active learning problems in that we must determine which data items should be given labels by AI workers. In addition, we must guarantee the quality requirement of the requesters.

(2) Task Assignment Algorithms with Theoretical Guarantees: We propose algorithms that provide solutions to the HACTAP with some theoretical guarantees, assuming that the probability distribution of accuracy can be estimated with beta distributions.

(3) Extensive Set of Experiments using Open Benchmark and Real-world Datasets: We conducted an extensive set of experiments using open benchmark and real-world datasets. The results show that our algorithm can assign many more tasks than baselines (including the active learning-based method (Zhu et al. 2010)) to AI when it is difficult for AIs to satisfy the quality requirement for the whole set of tasks and that the algorithm can flexibly change the number of tasks assigned to multiple black-box AI workers, in accordance with the performance of available AI workers. Our code and the data used in the experiments are available on GitHub[4].

[3]We use the term 'AI' in the broad sense; AI workers are software agents based on any algorithms, including rule-based and machine learning algorithms.

Related Work

There has been some research related to the issues discussed in this paper. First, active learning (Settles 2009; Yan et al. 2011) is related to our problem in the sense that both algorithms are designed to determine which data items should be labeled by humans. Because the essential purpose of active learning is to maximize the performance of the machine learning model with a given budget (Kutsuna et al. 2012; Yang et al. 2018), active learning is a less effective solution for problems with a fixed number of tasks (Jörger, Baba, and Kashima 2016). In the experiment in Section , we show that active learning cannot fully exploit the power of the AI crowd even if we combine them using ensemble methods.

Second, ensemble learning is also related to our problem because it examines the outputs of multiple AI programs to generate an aggregate result (Beluch et al. 2018). Ensemble learning unifies all the AI workers to be a single AI worker, which leads to an "all-or-nothing" scheme in the assignment for the human+AI crowd. Because we consider a dynamic assignment of workers in our problem, we should assign easy tasks to the AI workers early in the process so that the human workers can focus on the AI-hard tasks. To address this point, the proposed method examines the individual outputs of each AI worker.

There are different patterns of human–machine collaboration for solving real-world problems (Kamar, Hacker, and Horvitz 2012; Russakovsky, Li, and Fei-Fei 2015; Chandra et al. 2020; Arous et al. 2021) while reducing human worker efforts (Yi et al. 2012). A typical approach is to use AI programs to identify data items that require attention from human workers (Nguyen, Wallace, and Lease 2015; Yang et al. 2019; Wilder, Horvitz, and Kamar 2020; Liu et al. 2020). Building a specific ML model that can output rejection to avoid missing classifications (Herbei and Wegkamp 2006) may also help determine human worker assignments. Training humans, such as crowd workers using an AI's output, is also an interesting application of human–machine collaboration (Abad, Nabi, and Moschitti 2017; Honeycutt, Nourani, and Ragan 2020; Wang and Sun 2021). In contrast, our approach identifies data items that can be processed using AI programs. We consider these two approaches to be complementary.

There is another approach to aggregate task results by post hoc label grouping (Chang, Amershi, and Kamar 2017): fine data clusters can be generated based on the nature of the data rather than using the actual fixed number of classes or the type of labels from workers. This concept can be applied to accept more task clusters from AI workers.

[4]https://github.com/crowd4u/HACTAP-Framework

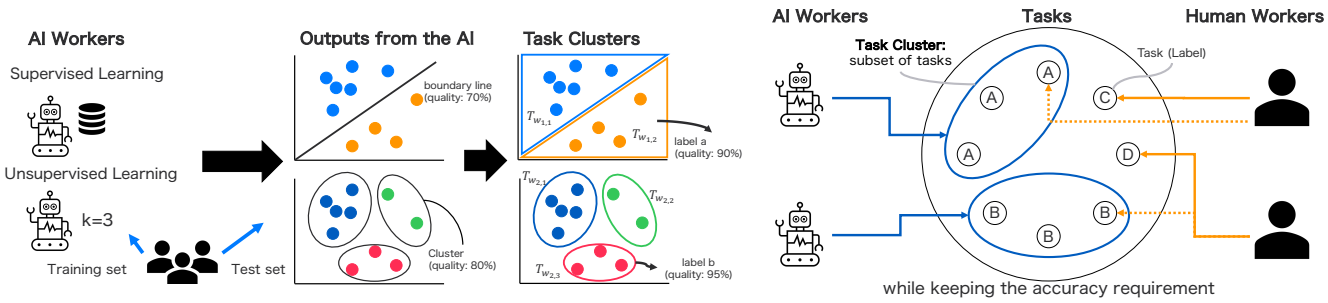

Figure 2: Left: Finding task "clusters" for which AI workers return high-quality results. Right: Our proposed methods assign tasks to human and AI workers (task cluster-wise) while keeping the quality requirement.

Section	Notation	Description
3.1	$T = \{t_1, \cdots, t_M\}$	A set of given multiple classification tasks with N classes.
3.1	$A = \{a_1, \ldots, a_N\}$	A set of the classes (labels)
3.1	$W = \{w_1, w_2, \ldots\}$, where $w_i : T \to \mathbb{N}$	A set of AI workers implementing any algorithms such as unsupervised, supervised, and rule-based algorithms. Here, we denote each AI worker w_i as a function that returns a natural number representing a cluster assigned by the AI worker to a given task.
3.2	$R_{w_i} = \{(t_j, t_k) \mid w_i(t_j) = w_i(t_k)\}$	The equivalence relation in the set of tasks T by the AI worker w_i where $w_i(t)$ is the label given to t by w_i.
3.2	$C_{w_i} = T/R_{w_i}$	A set of task clusters w_i generates. We define C_{w_i} as the set of sets of tasks with the same predicted label derived by the equivalence relation R_{w_i}.
3.2	$C = \bigcup_{w_i \in W} C_{w_i}$	All task clusters from W at the current assignment progress.
3.3	$ans : T \to \{A \times (W \cup \{`h'\})\} \cup \{(\emptyset, \emptyset)\}$	An updatable function that takes a task and returns a pair of the result and the assigned worker. Here, a human worker is denoted by h.

Table 1: Notation

Human+AI Crowd Task Assignment Problem

First, we define our problem setting as the Human+AI Crowd Task Assignment Problem (HACTAP) in this section. We then introduce task clusters, an essential concept of our algorithms. Finally, we propose two task assignment algorithms designed to handle numerous AI workers that dramatically change their performance.

Problem Definitions

Table 1 shows the notation used in this paper. We use T and A to denote the set of given multiclass classification tasks and the set of labels (i.e., class names) for T, respectively. W denotes a set of black-box AI workers implementing any algorithms developed by anonymous people. In our setting, we only have access to the training and inference functionalities of each AI worker. We assume that answers from human workers are aggregated in some way; thus, we simply denote them as 'h'.

Then, a *task assignment* S is a sequence of pairs (t_j, w_i)s, where $w_i \in W \cup \{`h'\}$ and $t_j \in T$, such that every $t_j \in T$ is assigned to a worker, that is, $|S| = |T|$. Let $0 \leq q \leq 1$ be a given accuracy requirement. We seek to calculate a task assignment in such a manner that the overall quality of the task results is at least q.

Definition 1 (Human+AI Crowd Task Assignment Problem, HACTAP)**.** We assume that we are given a set T of classification tasks, a set W of AI workers, and an accuracy requirement of $0 \leq q \leq 1$, where the accuracy is defined as the ratio of the number of task results that are the same as human task results. Then, the HACTAP computes a task assignment S such that the average accuracy of the results is at least q.

In other words, the problem intends to find different assignments of tasks to AI and humans in a way that maintains the same quality level as using only human workers. Note that this problem has a trivial solution, that is, assigning all tasks to human workers. We describe a task assignment S_1 as *more efficient* than another task assignment S_2 if S_1 has a smaller number of pairs $(t_j, `h')$s, to assign tasks to human workers other than S_2 and still satisfy the accuracy requirement; that is, the average accuracy of the results is at least q. A task assignment S_1 is *optimal*, when no assignment is more efficient than S_1 in all the task assignments that satisfy the q. However, obviously, the optimal solution is undecidable when AI workers are black boxes and their internal workings are unexaminable.

Therefore, we must estimate the accuracy of the results of AI workers from the outside to obtain a good task assignment. In addition, we must perform this estimation dynamically, as supervised AI workers are included that improve their ability as the number of data items for training increases.

Our Approach: Task Clusters

Before describing our algorithms, we introduce task clusters, the key idea necessary for maximizing the number of tasks assigned to AI workers in the HACTAP. If we identify the AI worker that provides sufficient accuracy for all tasks, it is better to assign all the remaining tasks to that AI worker. However, such ideal AI workers are not always available and are likely to take a longer time to build. As the experimental results show, our algorithms based on the idea of task clusters can assign more tasks to AI workers than an all-or-nothing approach.

We formulate the output of each AI worker as a set of task clusters (illustrated on the left side of Figure 2). Each task cluster is a subset of tasks for which the same label is given by the AI worker. In our setting, each AI worker is expected to receive a set of (task, label) pairs as training data and divide a test set into k clusters of tasks, where k can be independent of the number $|A|$ of potential labels for the task (such that unsupervised clustering algorithms can be used as AI workers). Conversely, it is not realistic for human workers to perform the same thing. In contrast to the ordinary machine learning context, AI workers do not need to have a high overall accuracy; instead, we evaluate each task cluster and accept the submission of the task cluster if its quality is sufficient relative to the required accuracy. This strategy achieves a fine-grained tradeoff control between the required quality and the number of tasks assigned to the AI workers (right side of Figure 2).

Table 1 also summarizes the notation also for task clusters. Let $R_{w_i} = \{(t_j, t_k) \mid w_i(t_j) = w_i(t_k)\}$ be the equivalence relation in the set of tasks T by the AI worker w_i. Formally, the set of task clusters obtained from an AI worker w_i is defined by $C_{w_i} = T/R_{w_i} = \{T_{w_i,1}, T_{w_i,2}, ...\}$ that is the quotient set for T. As mentioned previously, the size of C_{w_i} depends on the AI worker. In many supervised algorithms, the $|C_{w_i}|$ will be $|A|$, however, cases other than that are also assumed. For example, if the clustering algorithm divides the tasks into k clusters, the $|C_{w_i}|$ will be k. The overall task clusters from AI workers W is denoted by $C = \bigcup_{w_i \in W} C_{w_i}$.

Clusterwise Test-based Assignment (CTA)

CTA uses the answers provided by human workers to statistically test each task cluster generated by AI workers. Through the statistical tests, CTA determines whether the quality of a task cluster satisfies the given accuracy requirement q, at a minimum (right side of Figure 2). If it holds, CTA accepts answers to the tasks in the task cluster and assigns the tasks to the AI worker who generated the cluster.

Input and Output. CTA takes a set T of tasks, a set W of AI workers, an accuracy requirement q, and a significance level α. The output is a sequence $[(t_1, w_{t_1}), \ldots, (t_k, w_{t_k})]$ of (task, worker) pairs, which is the history of the executing $assign(t, w)$ function in the procedure.

Procedure. Algorithm 1 explains the CTA procedure. In short, it executes $assign(t, w)$ successively and updates the function ans so that it records the task result (label) $a \in A$ obtained by the assignment (t, w). Formally, $ans : T \to A \times (W \cup \{\text{`}h\text{'}\}) \cup \{(\emptyset, \emptyset)\}$ is an updatable function that takes a task as input and returns a pair consisting of the task result and the assigned worker who provided the result.

Initially, $ans(t)$ returns $(\emptyset, \emptyset)$ for every t, which implies that no task has obtained the result. The algorithm updates ans until $ans(t)$ returns no $(\emptyset, \emptyset)$ for any t. We ask humans to complete a task t, where $ans(t) = (\emptyset, \emptyset)$. CTA randomly assigns tasks to human workers (Line 2). Each time we receive a label (e.g., `a` or `b`) for a task t from a human worker (Lines 3), we perform the following steps (Lines 4 to 9. See the left side of Figure 2, too): For each $T_{w_i,j} \in C$ (the j^{th} task cluster submitted by AI worker w_i), we conduct a statistical test. The test verifies whether the ratio of label $\hat{a}$ in $T_{w_i,j}$ is greater than the accuracy requirement q or not by using the observation of human answers already provided. Here, the $\hat{a}$ is the most frequent label type found in the human answers for that task cluster. For this step, any statistical test can be applied if the test guarantees that the ratio of $\hat{a}$ is greater than q at a given significance level α. In our experiment, we used a binomial test in which the number of trials $n_{w_i,j}$ and the number of positive outcomes $n^{(p)}_{w_i,j}$ were calculated as follows:

$$
\begin{aligned}
n_{w_i,j} &= |\{t \in T_{w_i,j} \mid ans(t) = (a, \text{`}h\text{'}), a \in A\}| \\
n^{(p)}_{w_i,j} &= |\{t \in T_{w_i,j} \mid ans(t) = (\hat{a}, \text{`}h\text{'})\}|
\end{aligned}
$$

If the test is positive, we update the labels of $ans(t)$ for every $t \in \{t' \in T_{w_i,j} \mid ans(t') = (\emptyset, \emptyset)\}$ to $(\hat{a}, w_i)$.

Theorem 1 explains how CTA guarantees the accuracy requirement. The proof of Theorem 1 is provided in the appendix.

Theorem 1 (output quality of CTA)**.** *CTA calculates the task assignments for AI workers such that the accuracy of the task results is at least q with a $(1-\alpha)^l$ probability, where l is the number of statistical tests, and α is the significance level for each statistical test.*

When the number of tasks is large, the number of accepted task clusters l tends to be large, which entails a considerably high statistical error rate. This issue arises from how each task cluster is tested independently, motivating us to develop another method, which is explained in the next section, Global Test-based Assignment (GTA).

Global Test-based Assignment (GTA)

This subsection explains an assignment algorithm that theoretically guarantees the overall accuracy of the task results. In contrast to CTA, which tests each task cluster to verify whether the accuracy is at least q, GTA tests the overall accuracy of all task results, not just those of each task cluster. GTA directly computes the probability distribution of the

Algorithm 1 Clusterwise Test-based Assignment (CTA)

Require: A set T of tasks, a set W of AI workers, the accuracy requirement q, and the significance level α.
Ensure: A sequence of (task, worker) pairs.
1: **for all** $t \in T$ s.t. $ans(t) = (\emptyset, \emptyset)$ in a random order **do**
2: $a' \leftarrow task\_result(assign(t, \text{'h'}))$
3: update ans so that $ans(t) = (a', \text{'h'})$
4: **for all** $T_{w_i,j} \in C$ **do**
5: **if** $statistical\_test(T_{w_i,j}, q, \alpha)$ **then**
6: $\hat{a} = \underset{a \in A}{arg\,max} |\{t \mid t \in T_{w_i,j}, ans(t) = (a, \text{'h'})\}|$
7: for $t' \in T_{w_i,j}$ s.t. $ans(t') = (\emptyset, \emptyset)$, $assign(t', w_i)$ and update ans so that $ans(t') = (\hat{a}, w_i)$
8: **end if**
9: **end for**
10: **end for**

Algorithm 2 The $statistical\_testing$ function for GTA

Require: A task cluster candidate $T_{w_i,j}$, the accuracy requirement q, and the significance level α.
Ensure: True if $T_{w_i,j}$ is acceptable and false otherwise.
1: let $\Gamma_{accepted}$ be a set of accepted task clusters
2: let $\Gamma = \Gamma_{accepted} \cup \{T_{w_i,j}\}$
3: let n be the number of iterations
4: let $v_{T_i,j} \sim Beta(1+T_i.r, 1+T_i.c) \forall T_i \in \Gamma, 1 \leq j \leq n$
5: $success = 0$
6: **for** j in range(n) **do**
7: $acc_j = \frac{\sum_{T_i \in \Gamma} v_{T_i,j} T_i.size}{\sum_{T_i \in \Gamma} T_i.size}$
8: **if** $acc_j \geq q$ **then**
9: $success = success + 1$
10: **end if**
11: **end for**
12: return $(1 - (success/n)) < \alpha$

overall accuracy based on those over the results in each task cluster.

The probability distribution of the overall accuracy was computed as follows: In the GTA, the probability distribution of the accuracy of each task cluster is modeled as a beta distribution. By observing the human worker labels obtained thus far, we can compute the posterior distribution of the accuracy of each task cluster by applying the beta-binomial conjugacy. Let $\Gamma = \{T_1, T_2, \ldots\}$ denote a subset of the task clusters[5]. We assume that V_{T_i} is a random variable that indicates the accuracy of the task cluster T_i. The prior probability of V_{T_i} is assumed to follow a uniform beta distribution, for example, $P(V_{T_i}) = Beta(1, 1)$. The posterior of V_i is calculated as $P(V_{T_i} \mid T_i) = Beta(1+T_i.r, 1+T_i.c)$, where $T_i.r$ is the number of correct tasks and $T_i.c$ is the number of incorrect tasks labeled by human workers in T_i. Let Acc be the overall accuracy of $\bigcup_{T_i \in \Gamma} T_i$ which is a random variable transformed from V, and is defined as follows:

$$Acc = \frac{\sum_{T_i \in \Gamma} V_{T_i} T_i.size}{\sum_{T_i \in \Gamma} T_i.size}, \quad (1)$$

where $T_i.size$ is $|\{t' | t' \in T_i, ans(t') = (\emptyset, \emptyset)\}|$. Subsequently, we can calculate the probability of whether the task assignments satisfy the accuracy requirement by assessing the distribution of the overall accuracy $P(Acc \mid \Gamma)$.

$$P(Acc < q \mid \Gamma) = \int_0^q P(Acc = a \mid \Gamma) da. \quad (2)$$

$P(Acc < q \mid \Gamma) < \alpha$ must be satisfied to ensure that the overall accuracy is at least q at a significance level α.

Global Test-based Assignment (GTA). Now we introduce GTA, which is the same as CTA in Algorithm 1, except that the *statistical_test* function (Line 5) does not conduct a cluster-wise test, but a global test as shown in Algorithm 2, which conducts a Monte Carlo simulation to compute the expression (2). GTA keeps $\Gamma_{accepted}$, which is a set of accepted task clusters, including a special task cluster T_{human}, which is the set of tasks answered by human workers.

Theorem 2 explains the quality guarantee of GTA. We discuss the proof of Theorem 2 in the appendix.

Theorem 2 (Correctness of GTA)**.** *GTA outputs an assignment such that the overall accuracy exceeds q as $J \to \infty$, where the J is the sample size of the Monte Carlo approximation.*

Experiment Using a Benchmark Dataset

First, we experimented using an open benchmark dataset to verify whether the outputs of the proposed algorithm actually satisfy the accuracy requirements and whether it could handle the trade-off between the requirements and the cost (i.e., the number of tasks performed by human workers).

Settings

We used 10-class classification tasks (i.e., $|A| = 10$) using 10,000 randomly selected images from Kuzushiji-MNIST (Clanuwat et al. 2018). Further, we used 15 types of AI workers implementing different algorithms with scikit-learn 0.23.1 and its default parameters. We compared our methods with two baseline algorithms: Worker-wise Random Sampling Test based assignment (WTA) and Active learning-based assignment (ALA), both of which evaluate the overall accuracy of AI workers. They require AI workers to output a probability because they adopt a weighted voting ensemble to deal with more than one AI worker, and ALA must determine the next task based on that. Therefore, among the 15 AI workers, they can only deal with 10. Note that ALA introduces a lack of flexibility in implementation (loss of parallelism, need dynamic interactions with platforms, etc.). Appendix shows the details and a full list of the 15 AI workers.

Baselines and Algorithm Details

This subsection explains the details of the algorithms compared in the experiments.

[5]We used T_k instead of $T_{w_i,j}$ because it is not important to determine from which AI worker each task cluster originated.

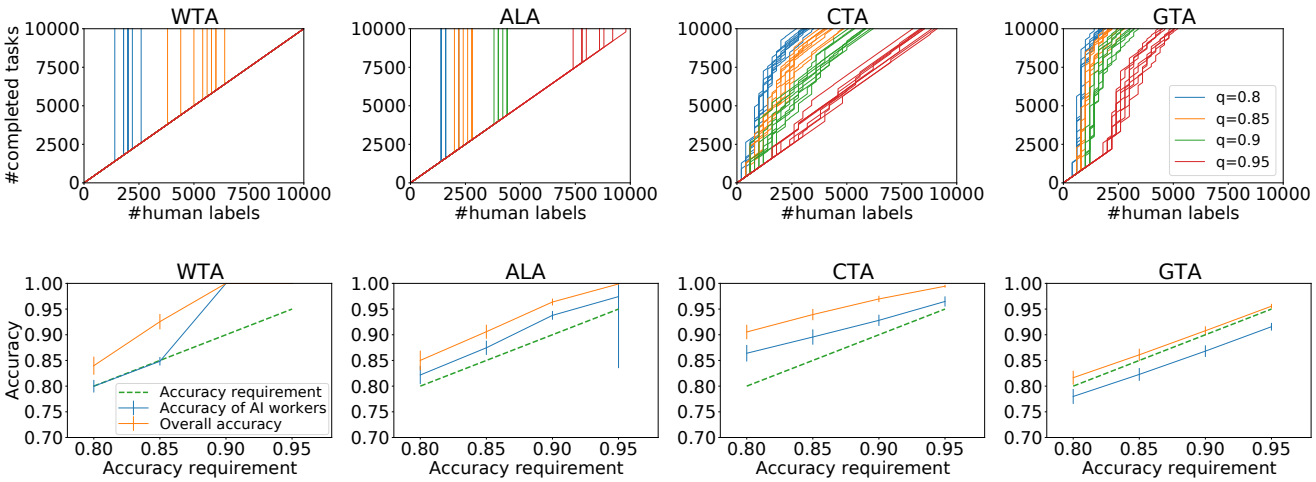

Figure 3: Experiment results on KMNIST. The top graphs show the relationship between the number of tasks completed by human workers and the number of completed tasks by human+AI workers. The bottom-row graphs show the overall accuracy of the task results with each accuracy requirement q (error bars represent SD).

Worker-wise Random Sampling Test-based Assignment (WTA). We introduce a worker-wise test-based assignment that examines whether the entire output from the AI worker has achieved the accuracy requirement (Algorithm 3). In our experiments, we combined the outputs from the set of AI worker candidates by weighted majority voting when there were multiple machine learning models.

Active Learning-based Assignment (ALA). We introduce ALA, which takes committee-based active learning using vote entropy sampling. In ALA, half of the human worker assignments are determined by the query strategy, and the other half is randomly assigned, in line 2 of Algorithm 3. The AI worker is trained for each human worker assignment using queried (task, label) pairs and evaluated with a random set. If the accuracy is equal to or better than the accuracy requirement, we assign all of the remaining tasks to the AI worker (Zhu et al. 2010). Note, however, that ALA violates our problem settings because (1) the AI worker is not a black box, and (2) the AI worker is assumed to join at the beginning of the execution (not in the middle of the execution).

Clusterwise Test-based Assignment (CTA). We used a binomial test at Line 6 in Algorithm 1 to verify whether $T_{w_i,j}$ is connected to $\hat{a}$. Here, the number of trials n is $n = |\bigcup_{t \in T_{w_i,j}} ans(t) = (*, \text{`}h\text{'})|$ (* is any label), and the number of successes $m$ ($\leq n$) is $m = |\bigcup_{t \in T_{w_i,j}} ans(t) = (\hat{a}, \text{`}h\text{'})|$.

Global Test-based Assignment (GTA). We set $n = 100,000$ for the number of iterations for the Monte Carlo simulation).

In our experiments, we evaluated the AI worker and their task clusters only when we obtained 200 new task results from the human workers. In CTA and GTA, we repeated lines 2 to 3 in Algorithm 1 to obtain the results of randomly choosing 200 tasks. We set the significance level $\alpha = 0.05$ in CTA and GTA.

The evaluation order for the available task clusters is also an interesting problem. We believe that this order can be extended to the reward design of AI workers. Because this research focuses on task assignment algorithms that satisfy the accuracy requirement, we evaluated the task clusters in a random order.

Algorithm 3 Worker-wise Random Sampling Test based assignment (WTA)

Require: A set T of tasks, an AI worker w, and the accuracy requirement q.
Ensure: A sequence of (task, worker) pairs.
1: **while** $accuracy(w) < q$ or $\exists t \in T$ s.t. $ans(t)! = (\emptyset, \emptyset)$ **do**
2: let $t \in T$ s.t. $ans(t) = (\emptyset, \emptyset)$ in a random order
3: $a' \leftarrow task\_result(assign(t, \text{`}h\text{'}))$
4: update ans so that $ans(t) = (a', \text{`}h\text{'})$
5: **end while**
6: for all $t' \in T$ s.t. $ans(t') = (\emptyset, \emptyset)$, $assign(t', w)$ and update ans so that $ans(t') = (w(t'), w)$

Results

The experiments were run 100 times for each setting on a computer with a Ryzen 9 3950X 16-Core Processor, 64 GB RAM, GeForce RTX 3090 GPU, Ubuntu 18.04, and Python 3.8.2.

Figure 3 shows the results of the experiment. The graphs in the top row illustrate the relationship between the number of tasks completed by the human workers and the number of tasks completed by human+AI workers for each algorithm. The lines in the same color show the simulation results for the same pair for a given accuracy requirement q with dif-

ferent random seeds. Each figure contains only 10 lines randomly chosen from one hundred trials for every setting.

Overall, all algorithms behaved well in response to changes in the required accuracy; they assigned more tasks to AI workers as the required accuracy decreased. There are several points that are worth noting. First, CTA and GTA complete tasks more quickly because they accept task clusters from AI workers without waiting for the overall accuracy that satisfies the requirement. Second, ALA performs better than WTA in that it can assign tasks to the AI worker even in settings with high accuracy requirements ($q = 0.9, 0.95$), because AI learns more efficiently. Third, ALA often performed better than CTA; it assigned more tasks to AI workers when the accuracy requirements were low ($q = 0.8, 0.85$). When q = 0.8 (blue line), ALA reached 100% (10000) tasks when the human workers completed 1000–2200 tasks, while CTA reached 100% with 2161–3780 tasks from human workers. This is because (1) ALA could train the AI worker with a few human worker labels that the AI worker selected, and (2) CTA needed some samples from human workers to accept task clusters from AI workers statistically. In particular, when the required accuracy is not high and active learning easily accomplishes the requirement, ALA accepts the task results of the AI model in the early stage. As this suggests, the all-or-nothing approach works well if an AI worker can process the entire task set with a quality higher than the required accuracy. Fourth, GTA performed the best among the four strategies, especially in high accuracy requirement settings. Note that GTA always produces better or at least comparable performance to ALA, even when the required accuracy is low. As shown next, GTA maximizes the number of tasks assigned to AI workers as long as the required accuracy is satisfied, which was achieved by our task cluster-based approach that allows fine control of the resulting quality.

The bottom row of Figure 3 shows the overall accuracy of the task results for each accuracy requirement q. The solid lines show the accuracy of the whole (Human+AI) task results and that of the task results by AI workers. The dotted line represents the quality requirements that should be satisfied.

The overall accuracy with CTA is always above q despite the weak theoretical guarantee. In contrast, GTA satisfies the accuracy requirement, and the result accuracy is much closer to the requirement. This is the result of fine control of accuracy through its statistical estimation over the entire set of task results.

In summary, our findings are as follows: First, WTA and ALA worked to meet the requirements of HACTAP, but they were suboptimal when the required accuracy was high. Second, CTA assigns tasks to AI workers conservatively, so the quality requirement is satisfied in many cases (even though there is no good theoretical guarantee), but is sometimes too conservative in assigning tasks to AI workers. In other words, the overall quality far exceeds the accuracy requirement. Third, our GTA, which has a theoretical guarantee, performed well while assigning more tasks to AI workers than the other algorithms.

In CTA, the final part of tasks are often assigned to a human worker, but in GTA they are often assigned to an AI worker. This is because the actual requirement for AI workers decreases in the final part by increasing the number of task assignments to human workers.

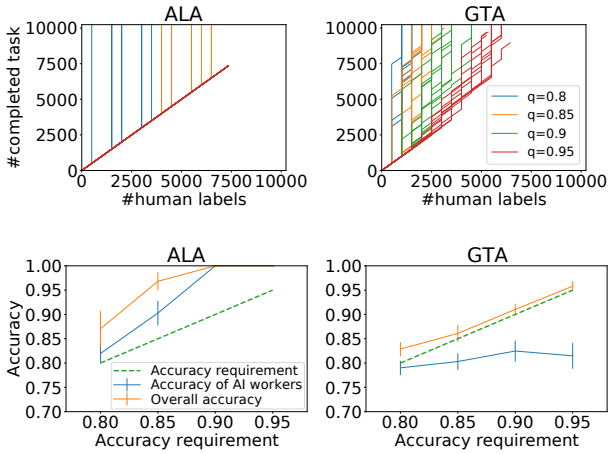

Figure 4: Experimental results on a real-world dataset. The top two graphs show the results of the task assignments. The bottom two graphs show the quality of the task results.

Experiment using a Real-World Dataset

We also confirmed the performance of our algorithms using a real-world application, an international natural disaster response drill[6]. In natural disaster responses, the requesters, such as local governments, need task results as soon as possible, as the situation can be rapidly developing and information is needed to make the up-to-date decisions. However, it is often difficult to predict what kind of system is required in advance, and it takes time to build such systems after a disaster occurs. We believe that the fully automatic integration of human and AI workers will be effective in such a situation.

Settings

We used aerial photographs of the Western-Japan Big Flood of 2018 that were captured by the Geospatial Information Authority of Japan[7], which includes 106 aerial photos. Specifically, we selected 10 images from them showing the surroundings of the actual flooded area. Then, we divided each image into 1024 blocks; hence, 10240 tasks were prepared for the experiment. In each task, the workers classified an image into three classes: non-flooded, flooded and covered by clouds. We collected labels from human workers using Amazon Mechanical Turk. The breakdown of the labels obtained was as follows: 4736 “non-flooded” labels, 2327 “flooded” labels, and 282 “covered by clouds” labels. The remaining 2895 tasks remained unlabeled.

Three classifiers based on deep neural networks were used as the AI workers. One of the AIs was developed by an

[6]https://crowd4u.org/events/mind-cndd/index.html

[7]https://www.gsi.go.jp/BOUSAI/H30.taihuu7gou.html

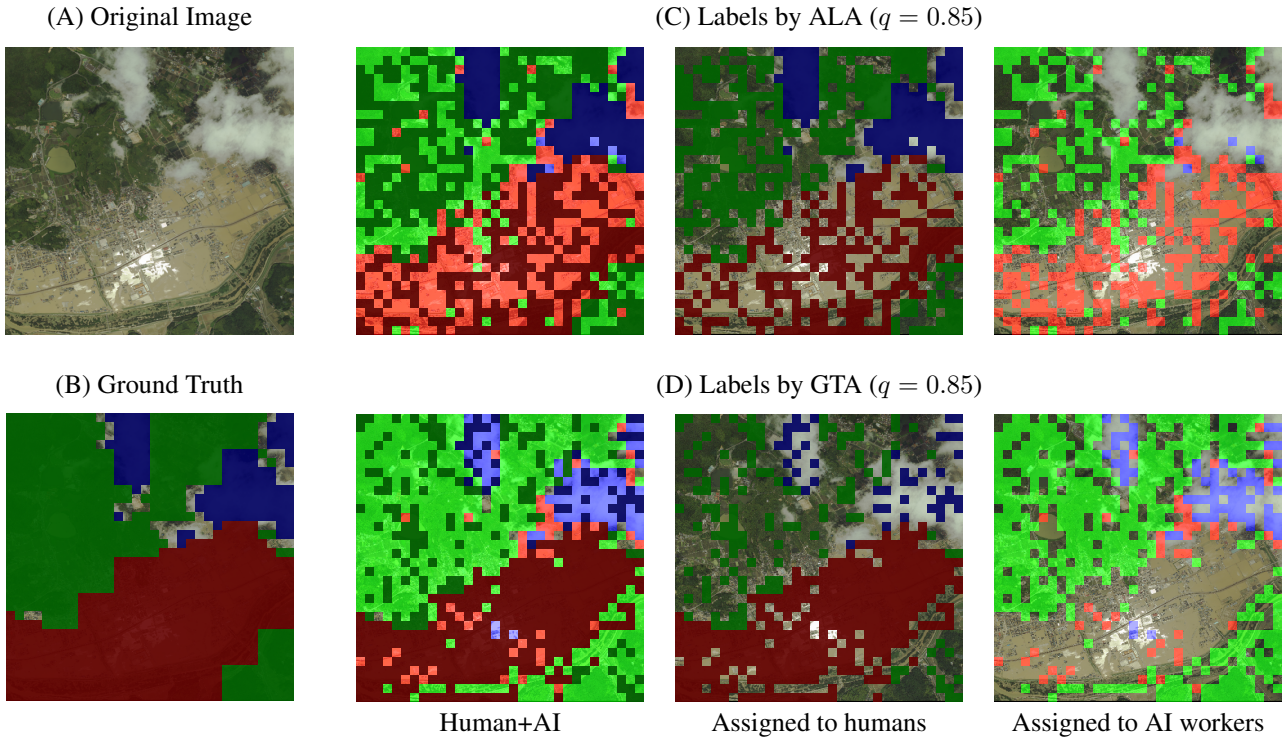

Figure 5: Experimental results of the part of an aerial photo. (A) The original aerial photo which contains a flooded area. (B) The ground truth labeled by human workers from Amazon Mechanical Turk. (C) The result labels assigned by ALA. (D) The result labels assigned by GTA.

anonymous developer for this disaster response project on a commercial crowdsourcing service. Although our idea does not require access to the code of AI workers, in this experiment, we received the corresponding executable code written in Python and Keras. It consisted of an eight-layer network, including convolution, pooling, dense, and dropout layers. The others were the ResNet-18 and VGG-16 models provided by the torchvision library in PyTorch. We paid the programmers approximately 240USD for 2h of labor (as reported by the crowd worker).

Results

We conducted the experiment ten times for each quality requirement, on the same computer as in Section 4.

In short, the result was similar to that of the simulation experiment; GTA assigned more tasks to AI workers than ALA, especially with higher accuracy requirements ($q = 0.9, 0.95$). However, ALA often performed better with lower accuracy requirements ($q = 0.8, 0.85$) because AI workers may easily reach the required performance. The top-side graphs in Figure 4 show the results of ALA and GTA with quality requirements $q = 0.8, 0.85, 0.9, 0.95$. The top two show the relationship between the number of tasks completed by human workers and the number completed by human+AI workers. The bottom two show the actual final accuracy by quality requirements. In the setting of $q = 0.8$ and 0.85, both GTA and ALA assigned a similar number of tasks to AI workers while maintaining the requirement. In the setting of $q = 0.9$ and 0.95, ALA assigned all tasks to humans because the AI worker did not satisfy this requirement. In contrast, GTA accepted task results from AI workers. The actual accuracy of the task clusters was lower than the requirement; however, GTA allowed them by considering the overall accuracy.

Figure 5 illustrates how our divide-and-conquer approach effectively found the task clusters that AI workers are good at. Figures 5 (A) and (B) is a part of the original image and the ground truth given by human workers, respectively. Figures 5 (C) and (D) show what tasks (fragments) of the images were assigned to human and AI workers by ALA and GTA with $q = 0.85$, respectively. GTA found that the AI worker was good at finding non-flooded parts in the forest area and immediately assigned the tasks in those areas to AI workers. Human workers were assigned tasks that focused on areas where the decision was more difficult.

Limitations and Future Work

Task type. We focused exclusively on image classification tasks in this study. However, our framework can be applied to classification tasks with other data types such as natural language and audio. Extending our framework to tackle other task types, such as semantic segmentation (Zhou et al. 2017) and object detection (Lin et al. 2014), is an interesting direction for future work.

Definition of accuracy of task results. We define accuracy as the ratio of the number of task results that are the same as human task results. The assumption here is that we want AI workers to behave like human workers. However, there may be scenarios in which we want to use accuracy as the ratio of the number of tasks that are the same as some ground truth, with the expectation that AI workers will perform better than other human workers. For this purpose, we need higher-quality task results to test the AI workers. A potential approach is to incorporate techniques such as (Daniel et al. 2018; Yang et al. 2019) for quality improvements of task results into our framework. This is an interesting future work.

Consideration of monetary cost and speed. We focused principally on maximizing the number of classification tasks assigned to AI workers while satisfying the accuracy requirement of the result. The underlying assumption is that human resources are more precious than AI resources, and we may not have enough people. However, we can consider more complex scenarios in which we need various labor costs for workers. It is relatively easy to set an objective function to consider the factor and optimize the assignment accordingly. Considering speed is trickier, we need to maximize parallelism for the fast completion of tasks, but it is a tradeoff between finding the best assignment and maximizing parallelism. Actually, CTA is the best when considering parallelism because it requires fewer sequential processes than ALA and GTA. The question of how to handle the tradeoff is also an interesting issue for future work.

Task clusters to be considered. The methods in this paper dealt with the task clusters given by AI workers as is. However, our definition of task clusters does not require it and states that the number of task clusters does not have to be the same as that of the classification labels. Therefore, we can artificially divide the task clusters given by AI workers to obtain a larger number of task clusters in smaller sizes. Such a division can be done by introducing additional (built-in) k-means, taking intersections of task clusters from other AI workers, using heuristics, etc. The obtained task clusters may help us assign tasks to AI workers more quickly.

Malicious AI workers. This paper did not deal with the problem of explicitly removing malicious AI workers. Low-quality or spam workers do not necessarily harm our framework because poor task clusters will not survive the statistical test. However, malicious AI developers may develop AI workers that returns correct answers to tasks in the test set, but arbitrary answers to other tasks. A potential approach to deal with such a problem is to compare task clusters from all AI workers to each other or use a built-in AI worker to produce meaningful task clusters. Then, we may find malicious AI workers by comparing the task clusters and finding ones that are inconsistent with others.

Efficient evaluation of task clusters. Although CTA and GTA are scalable, in the sense that they do not depend on a particular AI model and can deal with any number of AI workers simultaneously, it does not mean that they are computationally efficient in terms of the number of AI workers and task clusters. Potential approaches to deal with the problem include evaluating the task clusters in parallel and pruning clusters that are unlikely to be accepted. Pursuing these directions is one of our future work.

Conclusion

We introduced a novel Human+AI Crowd Task Assignment Problem (HACTAP) that satisfies the accuracy requirement of requesters. Our approach in solving HACTAP is based on accepting AI outputs partially instead of fully. We provided some theoretical analyses of the proposed algorithms.

The experimental results demonstrated that the outputs of the algorithms satisfied the accuracy requirements and that the algorithms can flexibly change the number of tasks assigned to AI workers according to the quality requirements.

Our results suggest the possibility of efficient task processing by appropriately sharing tasks between AI workers and human workers.

Acknowledgments

We would like to thank Yukino Baba, Keishi Tajima, Masaki Matsubara, Hiroyoshi Ito, and Kazuyo Tanaka for their valuable comments. We also thank the crowd workers on Crowd4U, including those who participated in the Indonesia–Japan International Cyber Natural Disaster Drill, for contributing to the advancement of crowd science and technologies. This work was supported by JST CREST Grant Number JPMJCR16E3 including AIP challenge, Japan.

Proof of Theorem 1

Proof outline. This is a straightforward conclusion based on the family wise error rate (FWER). □

Theorem 1 clearly shows that it is difficult to guarantee that the accuracy of the final CTA result is at least q. In the general statistical context, FWER control methods such as the Bonferroni correction, are applicable to multiple statistical tests by adjusting the actual significance level. However, in the HACTAP, FWER control methods cannot be applied directly because the number of task cluster candidates is unknown until the assignment is completed. There are two potential approaches to address this issue: (1) adjusting the significance level to be acceptable for the FWER by estimating the number of statistical tests, and (2) applying online FWER control methods (Javanmard, Montanari et al. 2018).

Proof of Theorem 2

Proof outline. Although the accumulated density cannot be analytically calculated, we can obtain the samples from the distribution $P(acc \mid \Gamma)$ by sampling each R_i from the posterior beta distribution $Beta(1+T_i.r, 1+T_i.c)$. By approximating the accumulated density $P(acc > q \mid \Gamma)$ using the Monte Carlo method, the requirement can be approximately verified using the following equation:

$$\frac{1}{J}\sum_{j=1}^{J}\delta\left(\frac{\sum_i R_{i,j}T_i.size}{\sum_i T_i.size} < q\right), \tag{3}$$

where $\delta(\cdot)$ is the delta function that returns 1 if the predicate $\cdot$ is true, and 0 otherwise. Based on the assumption given in Eq. (1) and (2), this quantity converges to $P(acc < q \mid \Gamma)$ when $J \to \infty$. The J means the sample size of the Monte Carlo approximation. This coincides with the quantity Algorithm 2 calculates. Therefore, GTA approximately guarantees that the overall accuracy is greater than the accuracy requirement q under the model assumption. □

AI Workers Participated in Experiment 1

The machine learning models used in Experiment 1 are listed below. Each item refers to the class name of the machine learning model implemented by scikit-learn.

1. MLPClassifier
2. ExtraTreeClassifier
3. LogisticRegression
4. KMeans (Used only by CTA/GTA)
5. DecisionTreeClassifier
6. SVC (probability=True option was used in WTA and ALA)
7. KNeighborsClassifier
8. GaussianProcessClassifier
9. MultinomialNB
10. AdaBoostClassifier
11. PassiveAggressiveClassifier (Used only by CTA/GTA)
12. RidgeClassifier (Used only by CTA/GTA)
13. RidgeClassifierCV (Used only by CTA/GTA)
14. ComplementNB
15. NearestCentroid (Used only by CTA/GTA)

References

Abad, A.; Nabi, M.; and Moschitti, A. 2017. Autonomous Crowdsourcing Through Human-Machine Collaborative Learning. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, 873–876. New York, NY, USA: ACM. ISBN 978-1-4503-5022-8. doi:10.1145/3077136.3080666. URL http://doi.acm.org/10.1145/3077136.3080666.

Arous, I.; Yang, J.; Khayati, M.; and Cudre-Mauroux, P. 2021. Peer Grading the Peer Reviews: A Dual-Role Approach for Lightening the Scholarly Paper Review Process. In *Proceedings of the Web Conference 2021*, WWW '21, 1916–1927. New York, NY, USA: Association for Computing Machinery. ISBN 9781450383127. doi:10.1145/3442381.3450088. URL https://doi.org/10.1145/3442381.3450088.

Beluch, W. H.; Genewein, T.; Nurnberger, A.; and Kohler, J. M. 2018. The Power of Ensembles for Active Learning in Image Classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9368–9377. Los Alamitos, CA, USA: IEEE Computer Society. doi:10.1109/CVPR.2018.00976. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00976.

Chandra, A. L.; Desai, S. V.; Balasubramanian, V. N.; Ninomiya, S.; and Guo, W. 2020. Active learning with point supervision for cost-effective panicle detection in cereal crops. *Plant Methods* 16(1): 1–16.

Chang, J. C.; Amershi, S.; and Kamar, E. 2017. Revolt: Collaborative Crowdsourcing for Labeling Machine Learning Datasets. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI 2017)*. ACM - Association for Computing Machinery.

Clanuwat, T.; Bober-Irizar, M.; Kitamoto, A.; Lamb, A.; Yamamoto, K.; and Ha, D. 2018. Deep Learning for Classical Japanese Literature. *CoRR* abs/1812.01718. URL http://arxiv.org/abs/1812.01718.

Daniel, F.; Kucherbaev, P.; Cappiello, C.; Benatallah, B.; and Allahbakhsh, M. 2018. Quality Control in Crowdsourcing: A Survey of Quality Attributes, Assessment Techniques, and Assurance Actions. *ACM Comput. Surv.* 51(1): 7:1–7:40. ISSN 0360-0300. doi:10.1145/3148148. URL http://doi.acm.org/10.1145/3148148.

Herbei, R.; and Wegkamp, M. H. 2006. Classification with Reject Option. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique* 34(4): 709–721. ISSN 03195724. URL http://www.jstor.org/stable/20445230.

Honeycutt, D.; Nourani, M.; and Ragan, E. 2020. Soliciting human-in-the-loop user feedback for interactive machine learning reduces user trust and impressions of model accuracy. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 8, 63–72.

Javanmard, A.; Montanari, A.; et al. 2018. Online rules for control of false discovery rate and false discovery exceedance. *The Annals of statistics* 46(2): 526–554.

Jörger, P.; Baba, Y.; and Kashima, H. 2016. Learning to Enumerate. In Villa, A. E.; Masulli, P.; and Pons Rivero, A. J., eds., *Artificial Neural Networks and Machine Learning – ICANN 2016*, 453–460. Cham: Springer International Publishing. ISBN 978-3-319-44778-0.

Kamar, E.; Hacker, S.; and Horvitz, E. 2012. Combining Human and Machine Intelligence in Large-Scale Crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '12, 467–474. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 0981738117.

Kutsuna, N.; Higaki, T.; Matsunaga, S.; Otsuki, T.; Yamaguchi, M.; Fujii, H.; and Hasezawa, S. 2012. Active learning framework with iterative clustering for bioimage classification. *Nature communications* 3: 1032.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft COCO: Common Objects in Context. In Fleet, D.; Pajdla, T.; Schiele, B.; and Tuytelaars, T., eds., *Computer Vision – ECCV 2014*, 740–755. Cham: Springer International Publishing. ISBN 978-3-319-10602-1.

Liu, A.; Guerra, S.; Fung, I.; Matute, G.; Kamar, E.; and Lasecki, W. 2020. Towards Hybrid Human-AI Workflows for Unknown Unknown Detection. In *Proceedings of The Web Conference 2020*, WWW '20, 2432–2442. New York, NY, USA: Association for Computing Machinery. ISBN 9781450370233. doi:10.1145/3366423.3380306. URL https://doi.org/10.1145/3366423.3380306.

Nguyen, A. T.; Wallace, B. C.; and Lease, M. 2015. Combining Crowd and Expert Labels Using Decision Theoretic Active Learning. In *Proceedings of the 3rd AAAI Conference on Human Computation and Crowdsourcing*. aaai.org.

Russakovsky, O.; Li, L.; and Fei-Fei, L. 2015. Best of both worlds: Human-machine collaboration for object annotation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2121–2131. doi:10.1109/CVPR.2015.7298824.

Settles, B. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Wang, Z.; and Sun, H. 2021. Teaching Active Human Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 5850–5857.

Wilder, B.; Horvitz, E.; and Kamar, E. 2020. Learning to Complement Humans. In Bessiere, C., ed., *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, 1526–1533. ijcai.org. doi:10.24963/ijcai.2020/212. URL https://doi.org/10.24963/ijcai.2020/212.

Yan, Y.; Rosales, R.; Fung, G.; and Dy, J. G. 2011. Active Learning from Crowds. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, 1161–1168. USA: Omnipress. ISBN 978-1-4503-0619-5. URL http://dl.acm.org/citation.cfm?id=3104482.3104628.

Yang, J.; Drake, T.; Damianou, A.; and Maarek, Y. 2018. Leveraging Crowdsourcing Data for Deep Active Learning An Application: Learning Intents in Alexa. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, 23–32. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-5639-8. doi:10.1145/3178876.3186033. URL https://doi.org/10.1145/3178876.3186033.

Yang, J.; Smirnova, A.; Yang, D.; Demartini, G.; Lu, Y.; and Cudre-Mauroux, P. 2019. Scalpel-CD: Leveraging Crowdsourcing and Deep Probabilistic Modeling for Debugging Noisy Training Data. In *The World Wide Web Conference*, WWW '19, 2158–2168. New York, NY, USA: ACM. ISBN 978-1-4503-6674-8. doi:10.1145/3308558.3313599. URL http://doi.acm.org/10.1145/3308558.3313599.

Yi, J.; Jin, R.; Jain, S.; Yang, T.; and Jain, A. 2012. Semi-Crowdsourced Clustering: Generalizing Crowd Labeling by Robust Distance Metric Learning. In Pereira, F.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems*, volume 25, 1772–1780. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2012/file/dd45045f8c68db9f54e70c67048d32e8-Paper.pdf.

Zhou, B.; Zhao, H.; Puig, X.; Fidler, S.; Barriuso, A.; and Torralba, A. 2017. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 633–641.

Zhu, J.; Wang, H.; Hovy, E.; and Ma, M. 2010. Confidence-Based Stopping Criteria for Active Learning for Data Annotation. *ACM Trans. Speech Lang. Process.* 6(3). ISSN 1550-4875. doi:10.1145/1753783.1753784. URL https://doi.org/10.1145/1753783.1753784.