Task Assignment Strategies for Crowd Worker Ability Improvement

Masaki Matsubara, Ria Mae Borromeo, Atsuyuki Morishima, Sihem Amer-Yahia

Task Assignment Strategies for Crowd Worker Ability Improvement

MASAKI MATSUBARA, University of Tsukuba
RIA MAE BORROMEO, University of Philippines Open University
SIHEM AMER-YAHIA, CNRS, Université Grenoble Alpes
ATSUYUKI MORISHIMA, University of Tsukuba

Workers are the most important resource in crowdsourcing. However, only investing in worker-centric needs, such as skill improvement, often conflicts with short-term platform-centric needs, such as task throughput. This paper studies learning strategies in task assignment in crowdsourcing and their impact on platform-centric needs. We formalize learning potential of individual tasks and collaborative tasks, and devise an iterative task assignment and completion approach that implements strategies grounded in learning theories. We conduct experiments to compare several learning strategies in terms of skill improvement, and in terms of task throughput and contribution quality. We discuss how our findings open new research directions in learning and collaboration.

CCS Concepts: • **Human-centered computing → Empirical studies in collaborative and social computing**.

Additional Key Words and Phrases: crowdsourcing for learning, task assignment, formalization

ACM Reference Format:
Masaki Matsubara, Ria Mae Borromeo, Sihem Amer-Yahia, and Atsuyuki Morishima. 2021. Task Assignment Strategies for Crowd Worker Ability Improvement. *Proc. ACM Hum.-Comput. Interact.* 5, CSCW2, Article 375 (October 2021), 19 pages. https://doi.org/10.1145/3479519

1 INTRODUCTION

Career advancement is considered a right in many physical workplaces, but it is not yet in place in online labor markets [46]. Up until now, most crowdsourcing research has catered to platforms with the goal of ensuring high worker performance, a.k.a., quality control and cost reduction. This focus does not always favor workers. The study of humans factors in crowdsourcing is a recent trend with various contributions that account for motivation [47, 48], mental stress [37], learning [18, 19, 29, 45, 60], as well as fatigue and boredom [7, 27, 49]. In this paper, we revisit task assignment and examine the *impact of learning strategies* on worker performance and skill improvement.

In physical workplaces, skill improvement strategies are regularly implemented and tested [15, 34, 39]. They include scaffolding where tasks are combined in alternating difficulty levels, and collaboration where workers learn from their interactions with higher-skilled peers. In online labor marketplaces, a few studies focused on the role of task difficulty and workers' ability to complete micro-tasks in improving skills [23], and how affinity between workers can be used to form teams that collaborate to produce high quality contributions while also improving skills [22]. Usually, such approaches require additional human cost to build training material or give feedback to workers. Moreover, there

Authors' addresses: Masaki Matsubara, masaki@slis.tsukuba.ac.jp, University of Tsukuba, 1-2 Kasuga, Tsukuba, Ibaraki, 305-8550; Ria Mae Borromeo, University of Philippines Open University; Sihem Amer-Yahia, CNRS, Université Grenoble Alpes; Atsuyuki Morishima, University of Tsukuba.

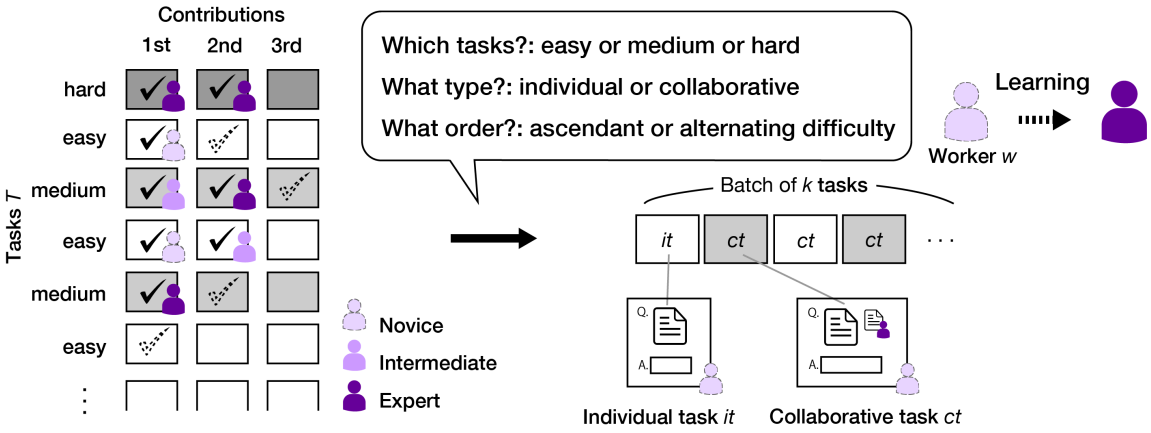

Fig. 1. Task assignment strategy for learning: Given a worker w and a set T of tasks, how to generate a sequence of k tasks for her to maximize her learning?

is little understanding of the interplay between achieving high worker performance, a.k.a., quality control and cost reduction, a platform-centric goal, and improving worker skills, a worker-centric goal.

This paper explores an approach for improving worker skills in crowdsourcing while also ensuring high worker performance; we study how to assign tasks to workers, expecting that appropriate assignments will have a positive impact on the inherent skill improvement of humans and on their overall performance. We focus on a common class of tasks referred to as "Knowledge and Comprehension tasks" in Bloom's taxonomy of educational objectives [5, 35] such as image classification, labeling, editing grammar and spelling mistakes, and speech transcription. Our question is illustrated in Figure 1. We have a set T of tasks of varying difficulty levels, each task receives N (=3 in the figure) contributions. At each iteration, some tasks have already been completed by some workers. Given a worker w and a set of uncompleted tasks, which sequence of k tasks will maximize w's learning potential? Here, learning potential is the maximum possible improvement in w's skill. We assume that worker skill and task difficulty are uni-dimensional and that the skill of a worker either remains the same or increases as time passes [43, 54].

Several studies showed that a worker learns better when contributions from higher-skilled workers are shown to them [18, 19, 29, 31]. We adopt this same model. In addition, although the impact of assignment strategies on worker's learning remains an open question, it is well-known that task ordering impacts platform-centric measures, such as quality and task throughput [1, 8, 17].

Given the above, our task assignment challenges are: **C1** - how to choose an appropriate batch of k tasks where a worker can see previous higher-skilled workers' contributions; **C2** - how to order the chosen k tasks appropriately so that the worker's skill improvement is maximized; **C3** - how to reconcile worker-centric and platform-centric goals.

To address **C1**, we formalize the *learning potential* of a worker for a task and choose k tasks that maximize the total learning potential. There are two theories underlying our framework. First, Zone of Proximal Development (ZPD) [55] is a well-known theory that defines three zones of tasks with different skill improvements; (1) A learnable zone that contains tasks a person can learn how to complete when assisted by a teacher or peer with a higher skill set, (2) a flow/comfort zone of tasks that are easy and can be completed with no help, and (3) a frustration zone of tasks that a learner cannot complete even with help. Second, the Flow theory [12] states that people are able to immerse

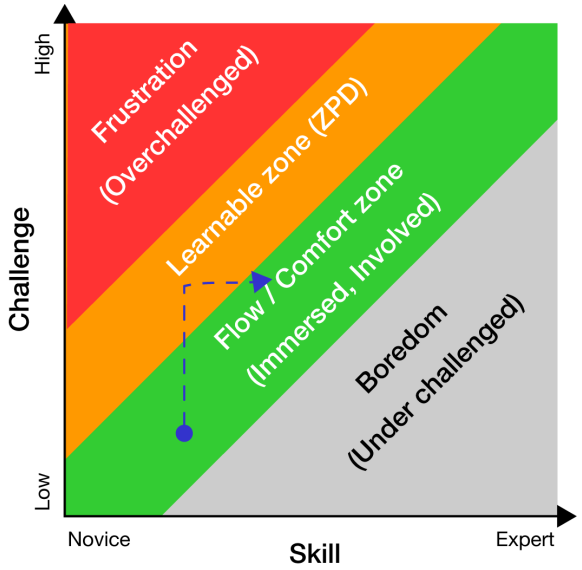

Fig. 2. Zone of Proximal Flow [4], which combines the results of ZPD and Flow Theory. In [4], it is shown that "scaffolding" tasks helps workers improve their skills by completing more challenging tasks (the dotted line).

themselves in doing things whose challenge matches their skills. Figure 2 integrates the two theories and illustrates their relationship with respect to the task challenge, the worker skill, and the affect state [4, 53]. In [4], the authors claim that to improve skills, the tasks should be either in the flow/comfort zone, or in the learnable zone on the condition that there is some "scaffolding" to help workers complete tasks that are a bit more challenging for them. This results in skill improvement (the dotted line). Our formalization builds on that and defines the learning potential for both individual tasks (mainly in the flow/comfort zone) and collaborative tasks (mainly in the learnable zone).

To address **C2**, we devise learning strategies which build on two ideas: (1) task ordering, and (2) interleaving individual and collaborative tasks. We study their impact on workers' performance and skills. Previous work found that both task ordering and task types impact contribution quality and completion time [8, 13, 17]. That is the basis for designing our four task orderings: NoOrder, a baseline where tasks are in no particular order; TotalOrder, where tasks are presented in increasing difficulty level, PartialOrder, a variation of TotalOrder, where tasks are grouped according to their difficulty and groups presented in increasing difficulty; and Alternate, that groups tasks and presents them in alternating difficulty levels.

To address **C3**, we propose an iterative task assignment process that takes a worker w, a learning strategy and a set of uncompleted tasks, and assigns to w, at each iteration, a batch of k tasks according to the learning strategy. We formalize a simple Knapsack optimization problem for skill improvement that incorporates learning strategies and solve it using a top-k search solution.

Our experiments show, with statistical significance, that the learning strategies are effective in helping workers improve their skills. More specifically, Alternate yields the highest average skill improvement for individual tasks, and workers produce the highest quality contributions, best task throughput, and highest skill improvement, when collaborative and individual tasks are interleaved.

In summary, our key findings are:

- Lessons on learning and working in physical workplaces also apply in virtual marketplaces: the ordering of tasks matters and collaborative tasks help learning.
- We can develop a task assignment algorithm that accounts for task ordering and learning.
- Alternate is the best strategy.

2 RELATED WORK

Since learning is one of the main motivations for joining the online job market [28], and most workers face "on-the-fly" learning situation [24], worker's skill improvement through task completion should constitute an important concern in crowdsourcing.

2.1 Paying Explicit Costs for Skill Improvement

For skill improvement, there have been studies on paying an extra human cost for helping other workers with advise, feedback review, Q&A, and explanation. For example, *Crowd Coach* [10] allows workers to give peers a short advice for how to do tasks while working. *Ask the Crowd* [40] lets workers ask questions and discuss answers. *AXIS* [57] lets workers provide explanations to future learners. *Atelier* [52] connects a worker as an intern to another worker as a mentor via online jobs. There are also several studies on performance improvement with additional tasks such as reviewing and justifying other answers [21, 60], reflecting own answers [30], and retaining workers via pricing [16]. Though these studies do not directly examine skill improvement, short-term performance improvement is observed and can be used to infer that workers experience some learning.

However, requesters usually have a limited budget in crowdsourcing settings. Our framework generates training material without requiring additional human costs, and while aiming to also achieve platform-centric goals.

2.2 Task Assignment Objective Functions

Another approach for skill improvement is task assignment, which controls when and to whom tasks are assigned. Assignment algorithms are usually designed to improve platform-centric and requester-centric goals such as result quality, completion time and throughput. There are also task assignment solutions that consider worker-centric goals, such as mental stress [37], motivation [48], affinity [22] and boredom [13].

It is known that carefully designed task assignment improves crowdwork quality. For instance, introducing micro-diversions [13] improves worker retention and contribution quality. Cai et. al. found that sorting impacts quality and completion time for editing tasks [8]. That is the basis for our task ordering. TotalOrder is inspired from [8] and Alternate is inspired from [13].

Many papers report that making other workers' contributions visible improves skills during task completion [19, 20, 29, 31, 32, 38, 42]. We use this kind of indirect communication among workers in our collaborative tasks. Our solution chooses collaborative tasks that have higher learning potential, without requiring requesters to pay additional costs.

2.3 Education Science and Theoretical Rationale

As Gadiraju pointed out, most crowdsourcing tasks are short and less time-consuming in nature, and workers face an "on-the-fly" learning situation [24]. In education science, such a situation is called *experiential learning* [33], i.e.,

Table 1. Task Metadata: Number indicates difficulty or contributor's skill. Three contributions are required for each task.

		Contribution		
Task ID	Difficulty	1st	2nd	3rd
t_1	Hard (5)	5	5	-
t_2	Easy (1)	1	-	-
t_3	Medium (3)	3	5	-
t_4	Easy (1)	1	3	-
t_5	Medium (3)	5	-	-
t_6	Easy (1)	-	-	-
t_7	Easy (1)	-	-	-
...	...	-	-	

learning by doing. Experiential learning is a process in which learners actively build their own understanding in a context-dependent manner, rather than learning by incorporating knowledge given by the teacher as it is the case in classrooms. An important assumption in experiential learning is that learners have different levels of skills. Therefore, experiential learning theory emphasizes the importance of choosing appropriate tasks for students, which is also the case in our context. Flow [12] and ZPD [55] theories conceptualize this idea. Recently, flow theory was applied in the physical world in on-the-job training [44], and was shown to be effective [15]. That is our rationale for adopting it as a basis to develop algorithms to choose the optimal task difficulty level for a given learner.

2.4 Learning through task completion in the Physical World

There are other learning theories applied to learning through task completion in the physical world, such as situated learning theory [41] and collaborative learning theory [6]. One representative of situated learning is *Apprenticeship* where knowledge is propagated from experts to novice workers based on based on the principle of *Legitimate Peripheral Participation* [41]. In this study, we design scaffolding based on the same principle. Collaborative learning is also effective in online learning environments like MOOCs, and studies have shown that rich interactions such as peer review, feedback and discussion promote learning [11, 14, 58].

We rely on those ideas as long as they do not require additional cost. For example, our collaborative tasks show a reference answer given by other higher skilled workers.

3 FORMALIZATION AND PROBLEM DEFINITION

Example 1. *We have three workers: a novice worker Mary, an intermediate worker John, and an expert worker Sarah. Their initial uni-dimensional skill levels θ are 1, 3, and 5, respectively. Given the tasks completed so far (Table 1), our goal is to assign to Mary a batch of $k = 5$ tasks that maximize her learning. The available tasks are chosen from $\{t_6, t_7, \ldots\}$ as individual tasks and $\{t_1, \ldots, t_5\}$ as collaborative tasks.*

Mary is a novice worker. Her ZPD [55] is medium difficulty level. She can only perform easy tasks by herself and medium tasks with the support of John or Sarah. She cannot perform hard tasks even if Sarah helps her. If we assign to Mary t_3 and t_5, we assume she can learn from seeing other workers' contributions [20]. Furthermore, assigning over-challenging tasks to her may result in frustration, and assigning under-challenging tasks may lead to boredom.

3.1 Formalism

Tasks and Workers. Crowdsourcing supports two task types: 1) individual tasks, denoted *idv*, performed by a single worker at a time, and 2) collaborative tasks, *col* completed by several workers. Whether a task is an individual or a collaborative task determines if a worker can see or not contributions from others while completing the task. For example, image labeling and text editing are individual tasks if workers do them independently, but are collaborative tasks if other workers' labels and sentences are made visible to them. In collaborative tasks, it is known that workers can learn from contributions by higher-skilled workers [20, 31]. Collaborative tasks can be completed in a fixed HTML form, and in a collaborative environment such as Google docs.

We consider a set of workers $\mathcal{W}$ and a set of tasks $\mathcal{T}$. When the distinction is not necessary, we use t to refer to either task types. Each worker completes tasks in batches. A batch $\mathbf{B}_w^i$ is a sequence of tasks of mixed type completed by a worker w at iteration i.

The skill of a worker w at iteration i is represented by θ_w^i. The difficulty of a task t is denoted by θ_t. We assume that worker skills and task difficulty are uni-dimensional and that the skills improve monotonically: the skill level remains the same or increases as a worker completes more tasks [43, 54]. θ_w^0 (the skill of a worker w at iteration 0) and θ_t are pre-computed (Section 5.1 under Exp. 1 Dataset and Flow describes for details on how skill and difficulty are estimated in our experiments). The difficulty of a batch $\theta_{\mathbf{B}}$ is defined simply as the average difficulty of all tasks in that batch:

$$\theta_{\mathbf{B}} = avg_{t \in \mathbf{B}} \theta_t$$

Learning potential. The learning potential of a batch of tasks $\mathbf{B}$ for a worker w at iteration $i+1$ depends on two factors: 1) the skill gap between w's skill and tasks in $\mathbf{B}$ and 2) w's performance factor. We now define these two concepts.

Skill gap. Let $skillgap(w, t)$ be the function that evaluates the gap between the worker skill and the task difficulty to reflect the hardness of completing t by w.

For an individual task *idv*:

$$skillgap(w, idv) = \theta_{idv} - \theta_w$$

If other workers v, whose skill θ_v is higher than θ_w, completed task t so far, t can be a collaborative task *col*. The skill gap is then reduced by other worker's support, i.e. we assume worker's skill become the average of θ_w and θ_v (If θ_v is higher than θ_t, it will be treated as θ_t). Therefore, for a collaborative task *col*:

$$skillgap(w, col) = \theta_{col} - \frac{\min(\theta_v, \theta_{col}) + \theta_w}{2}$$

In Example 1, as specified in Table 1, t_6 is an individual task, and t_5 cannot be an individual task for Mary, but should be a collaborative task. Thus, $skillgap(\text{Mary}, t_6) = 1 - 1 = 0$, $skillgap(\text{Mary}, t_5) = 3 - (3+1)/2 = 1$. In practice, $skillgap(w, col)$ is time-dependent and only those workers v who completed *col*, and whose skill is higher than w's, will help improve the skill of w. We will consider this when we formalize our problem.

Performance Factor. Let s_w^i be the performance factor of worker w at iteration i. This factor captures the difference between the *expected* performance and the *observed* performance of w for a batch of tasks $\mathbf{B}$ at iteration i.

$s_w^i \in$ [-1,1] reflects whether the observed performance is better than the expected (s_w^i is close to 1), same ($s_w^i = 0$), or worse (s_w^i is close to -1). A worker's learnable zone is extended when the performance factor is high.

$observed(w, \mathbf{B}_w^i)$ is computed as the average worker performance for tasks in $\mathbf{B}_w^i$. We measure performance as a tuple of quality and completion time. Those performance scores should be normalized, e.g. z-score: a standard deviation

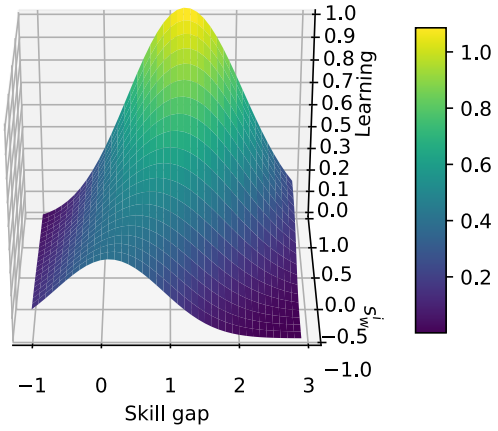

Fig. 3. Learning potential distribution

from the mean.

$$observed(w, \mathbf{B}^i_w) = \langle avg_{t \in \mathbf{B}^i_w} quality(w,t), avg_{t \in \mathbf{B}^i_w} time(w,t) \rangle$$

$expected(w, \mathbf{B}^i_w)$ is computed as the average performance of past tasks whose skill gap is the same as tasks in $\mathbf{B}$ at iteration i;

$$expected(w, \mathbf{B}^i_w) = \langle avg_{\hat{t} \in \hat{\mathbf{B}}} quality(w, \hat{t}), avg_{\hat{t} \in \hat{\mathbf{B}}} time(w, \hat{t}) \rangle,$$

where $\hat{\mathbf{B}} = \{\hat{t} \mid \hat{t} \in \{\mathbf{B}^1_w \dots \mathbf{B}^{i-1}_w\}, \theta_{\hat{t}} \simeq \theta_{t \in \mathbf{B}^i_w}\}$.

A worker's performance factor is therefore measured as follows:

$$s^i_w = \text{erf}(\text{d}(observed(w, \mathbf{B}^i_w), expected(w, \mathbf{B}^i_w))),$$

where $\text{d}(\mathbf{p}, \mathbf{q})$ is a Euclidean distance of two vector $\mathbf{p} = \langle p_1, p_2, \dots, p_n \rangle$ and $\mathbf{q} = \langle q_1, q_2, \dots, q_n \rangle$

$$\text{d}(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

and $\text{erf}(z)$ is Gauss error function of error z ($z \in \mathbb{Z}$):

$$\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-x^2} \, \text{d}x$$

We can now define the learning potential function that takes a worker w, a task t, and a batch of tasks $\mathbf{B}$, $learning(w,t)$ and $learning(w, \mathbf{B}^{i+1}_w)$, as follows:

$$learning(w,t) = \text{SN}(0, 1, s^i_w) \times e^{skillgap(w,t)}$$

$$learning(w, \mathbf{B}^{i+1}_w) = \sum_{t \in \mathbf{B}^{i+1}_w} learning(w,t)$$

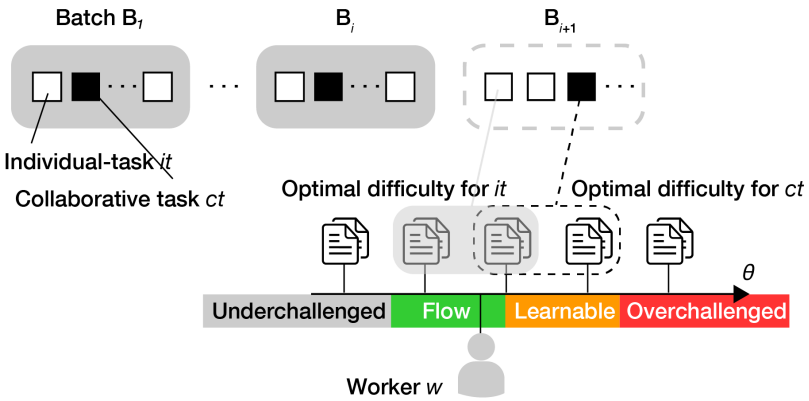

Fig. 4. Schematic illustration of our problem

where $\mathrm{SN}(0, 1, \alpha)$ is the skew normal distribution, and α is shape parameter of skewness. The distribution is right skewed if $\alpha > 0$ and is left skewed if $\alpha < 0$. When $\alpha = 0$, it corresponds to standard normal distribution. Figure 3 shows the distributions of learning potential. The optimal skill gap that maximizes learning changes according to the worker's current performance factor. That is, the higher (resp. lower) the performance factor, the higher the optimal skill gap (resp. lower). If there is a stagnation during the batch, the performance factor decreases. Thus, easier tasks will be assigned for the next batch, expecting to reduce stagnation. In example 1, if $s^i_{\mathrm{Mary}} = 0$, her optimal skill gap is around 1. For individual tasks, her learning potential is expected to be high in difficulty level (up to 2) and hence tasks t_6 and t_7 with an easy level, are optimal for her. For collaborative tasks, her learning potential is expected to be high with t_5, because $skillgap(\mathrm{Mary}, t_5) = 3 - (3 + 1)/2 = 1$. Interleaving individual and collaborative tasks in a batch will provide different levels of task difficulty.

Learning Strategies. A learning strategy defines an ordering of tasks in a batch and takes into account the type of tasks. We draw inspiration from [8, 17] and define several strategies and test them in our experiments. For task ordering: NoOrder where tasks are presented in no particular order; TotalOrder where tasks are presented in increasing difficulty; PartialOrder where tasks are grouped according to their difficulty and groups presented in increasing difficulty; Alternate where tasks are grouped according to difficulty and groups presented in alternating difficulty levels. For task type: individual tasks only; interleaving individual and collaborative tasks; collaborative tasks only.

3.2 Problem Definition

We are now ready to define our task assignment problem (Figure 4). Given a worker w and the batches of tasks completed by w up to iteration i: $\mathbf{B}^1_w \ \ldots \ \mathbf{B}^i_w$, find a batch $\mathbf{B}$ of at most k tasks to assign to worker w at iteration $i + 1$ such that:

$$\underset{\mathbf{B}}{\operatorname{argmax}}\ learning(w, \mathbf{B})$$

Our problem is to determine the right batch of tasks to provide to a worker at every iteration. At each iteration, one of NoOrder, TotalOrder, PartialOrder, or Alternate is applied before providing tasks to workers.

Challenges. Our problem raises several challenges.

1. Assigning tasks to workers. Our problem is a variant of the Knapsack Problem [9]. Items are tasks and each task has a value (in our case $v = learning(w, t)$) and a weight (in our case 1), we want to find k tasks that maximize the sum of values $\sum v_i$ under a capacity constraint k. What makes our problem simple is that the weight is equal to 1 which yields a top-k solution. Additionally, as the value of assigning a task to a worker depends on the worker and evolves over time as other workers complete tasks, we need to account for that dynamicity in the task assignment process.

2. Handling different task types. Out of the k tasks in each batch, some could be individual tasks, others collaborative. We deliberately left out from our problem statement the proportions of each task type. It could be modified by adding constraints that specify the exact numbers or bounds on each task type. It could also be left to the optimization objective to pick and choose among task types. In this work, we propose to solve the batch problem first and then choose the desired mix of task types according to a given learning strategy.

3. Updating task and worker information. Whenever a worker completes a task, the task's metadata is updated, i.e., contributor's answer and skill level are recorded, and the number of required contributions is reduced by 1. A worker's performance factor and skill also need to be updated after workers complete a batch. We do not limit worker withdrawal and we update metadata only with observed performance. We assume that a worker's skill improves monotonically: the skill level remains the same or increases as time passes [43, 54]. A worker's skill is updated as follows:

$$\theta_w^i = \max_{t \in \mathbf{B}_w^i | observed(w,t) \geq \langle \tilde{Q}, \tilde{T} \rangle} \theta_t, \theta_w^{i-1},$$

where $\tilde{Q}$ and $\tilde{T}$ are thresholds for quality and time completion at which a worker can be considered to have mastered a task. These thresholds are defined as the worst value among workers whose skill is above the difficulty of the task. Additionally, we need to set θ_t to θ_w^i if the observed performance exceeds the threshold in task t.

Worker retention in the multi-batch case needs to be carefully formalized. For instance, assume a worker could complete only one task per batch and stay the whole time (all 10 batches), and another worker could complete all tasks in the first 3 batches and leave. It is difficult to determine which one of the two has a higher retention. Taking into account such cases is planned for future work in which performance factor is considered in formulating retention.

4 OUR SOLUTION

We describe the overall architecture of our solution following which we describe our task assignment algorithm.

4.1 Overall architecture

Figure 5 shows the architecture of our solution. There are several components in the architecture and each component is called at different times (either during the completion of a batch) or between batches:

In a preprocessing phase, (1) we compute worker and task metadata, and between iterations, (2) we assign tasks to a worker (solve our problem and apply a task ordering), and (3) quantify the performance of that worker (quality of contribution and completion time) and update that worker's skill and performance factor.

To address (1), we ask available workers to perform a set of gold standard tasks to estimate worker skills and task difficulty; we can directly measure them as we did in our experiments. (Details on how skill and difficulty are estimated

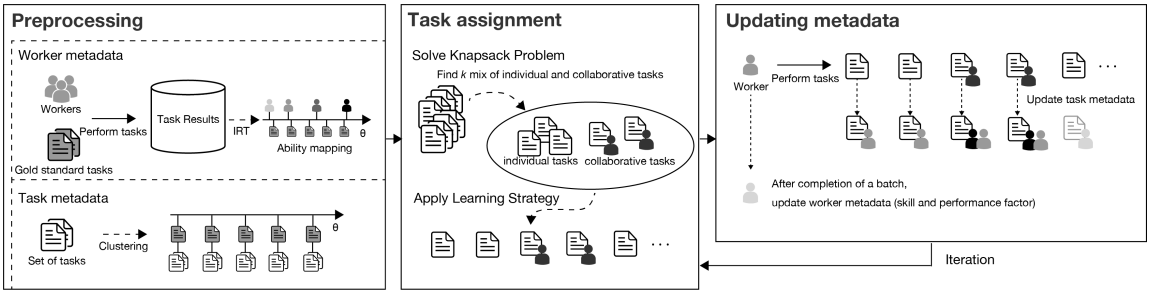

Fig. 5. Overall architecture. 1) Preprocessing: estimate worker's skill and task difficulty. 2) Task assignment: solve the Knapsack Problem and apply a learning strategy, then assign the tasks to our worker. 3) Updating metadata: update task metadata during completion of batch, and quantify the worker performance and update worker metadata after completion of batch.

Algorithm 1: Iterative Task Assignment, Ordering and Completion Algorithm

Input: $w, \theta_w^{i-1}, s_w^{i-1}, \mathcal{T}, l$
$\mathbf{B}_w^i \leftarrow$ call Algorithm 2 to solve Knapsack for worker w, tasks $\mathcal{T}$
Apply learning strategy l to $\mathbf{B}_w^i$
for *task* t *in* $\boldsymbol{B}_w^i$ **do**
    Let worker w complete task t
    Calculate $quality(w, t)$ and $time(w, t)$
    Update task metadata
end
Update worker skill θ_w^i and performance factor s_w^i
Next iteration with $w, \theta_w^i, s_w^i, \mathcal{T}, l$

in our experiments are described in Section 5.1 under Exp. 1 Dataset and Exp. 1 Flow). We can also use test theories to calculate them such as Item Response Theory (IRT) [3] to compute worker metadata and difficulty of the tasks at once. Using those difficulties as training examples, all available tasks are clustered into several difficulty levels by a k-NN algorithm. The number of clusters is given by the task requester, and feature vectors for clustering depend on the task type.

4.2 Assignment algorithm

To assign the tasks, we run Algorithm 1 that takes as input a worker w with skill θ_w^{i-1} and performance factor s_w^{i-1}, a set of available tasks $\mathcal{T}$ and a learning strategy l, and returns a set of tasks $\mathbf{B}_w^i$ for w. At a high level, the algorithm selects the tasks assigned to a given worker according to contributions from other workers, and applies a learning strategy l, NoOrder, TotalOrder, PartialOrder, or Alternate, to produce a task ordering. To solve the Knapsack problem, at the beginning of each iteration, we calculate the learning of each task $learning(w, t)$ based on θ_w^{i-1} and s_w^{i-1}, and find top-k tasks. To update task and worker metadata, each time a individual task is completed, it is marked as done. Each time a collaborative task is completed, the number of remaining workers to complete it decreases by one. Each time a full batch is completed, the performance of the worker (contribution quality and completion time) and the worker's skill are quantified.

The top-k search can be computed linearly with the number of tasks (Algorithm 2). To make this search efficient, we keep the sorted list of tasks in their difficulty. Then, we can prune irrelevant tasks that cannot have a larger learning

Algorithm 2: Solving Knapsack Problem

Input: $w, k, \mathcal{T}$ (Sorted in difficulty)
Output: $\mathbf{B}^i_w$

```
Index j ← The binary search result for finding optimal index at maximal learning potential
B^i_w s ← a tentative set of k tasks around j that are likely to have high learning potentials
Set bound_l and bound_u to the lowest and highest indexes of the range of tasks whose learning potentials are
  more than the minimum one in B^i_w
for i ← bound_l to bound_u do
    if learning(w, T[i]) ≥ lowest learning potential of the B^i_w s then
        Replace the lowest one of the B^i_w s with T[i]
    end
end
```

potential than a given threshold based on learning potential function $learning(w, t)$. Therefore, finding a good threshold is important for reducing the number of tasks to be examined. To find a good threshold, we can pick up a tentative set of k tasks that are likely to have high learning potentials (such as those in the middle range of the sorted tasks). Then, we search again for the final set of k tasks among the remaining range of tasks after pruning irrelevant tasks using the minimum learning potential in the tentative k tasks.

5 EXPERIMENTS

Our experiments are based on actual task deployments to verify the impact of learning strategies on skill improvement and their performance in terms of quality and throughput[1] of their contributions. Specifically, we first study the impact of learning strategies applied to 12 or 120 individual tasks **(Exp. 1)**. We further examine the workers' learning and performance in a series of collaborative tasks, and in a series of interleaved collaborative tasks and individual tasks **(Exp. 2)**. We use tasks that belong to the "Knowledge and Comprehension" class of Bloom's taxonomy [5]. In particular, we use image classification tasks in Exp. 1 and text editing tasks in Exp. 2. Scripts, data sets and worksheets used in the experiments are available on GitHub.[2]

Summary of Results. We observed that learning strategies are effective in helping workers improve their skills. Among the four task orderings, Alternate yielded the highest average skill improvement for individual tasks. We also found that workers obtained higher skill improvement and throughput when completing interleaved collaborative tasks and individual tasks, compared to collaborative tasks alone. All our results are statistically significant.

5.1 Exp. 1: Task ordering as a learning strategy

Exp. 1 Tasks. The task is to identify the specified blackbird given a pair of two bird images (Figure 6). The image pair may be a red-winged blackbird and bronzed cowbird, bronzed cowbird and brewer blackbird, and brewer blackbird and rusty blackbird.

Exp. 1 Dataset. We used images from the Caltech-UCSD Birds 200 images data set. The data set generated 2,186 tasks or image pairs (728 pairs of red-winged blackbird and bronzed cowbird, 702 pairs of bronzed cowbird and brewer's blackbird, and 756 pairs of brewer's blackbird and rusty blackbird). We randomly selected 120 tasks and crowdsourced the difficulty rating of each task in Amazon Mechanical Turk. Workers with more than 99% HIT acceptance rate were

[1] Quality is checked against a ground-truth and throughput is defined as the number of tasks per minute.
[2] https://github.com/virtualtaskselection/task-assignment

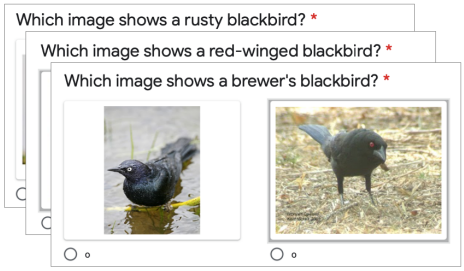

Fig. 6. The image classification task

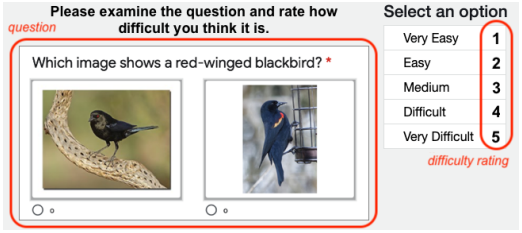

Fig. 7. Difficulty rating task in AMT

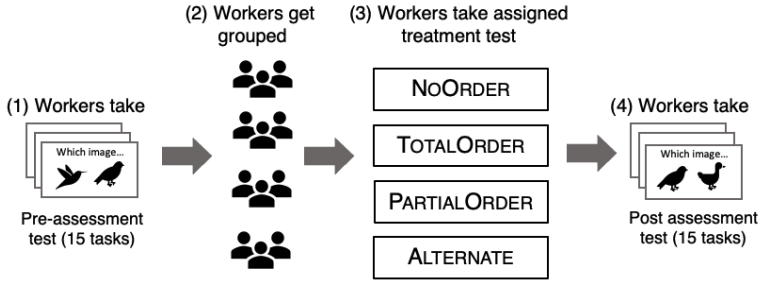

Fig. 8. Flow of Exp. 1

Table 2. Task Composition of Treatment Tests for Exp. 1

Learning Strategy	12-task Experiment	120-task Experiment
NoOrder	12 tasks randomly taken from the task data set	120 tasks randomly taken from the task data set
TotalOrder	4 easy tasks (tasks 1-4), 4 medium tasks (tasks 41-44), 4 hard tasks (tasks 81-84)	40 easy tasks (tasks 1-40), 40 medium tasks (tasks 41-80), 40 hard tasks (tasks 81-120)
PartialOrder	4 randomly selected tasks each from the easy, medium, and hard groups	40 randomly selected tasks each from the easy, medium, and hard groups
Alternate	2 easy, 2 medium, 2 easy, 4 hard, 2 medium tasks	10 easy, 10 medium, 10 easy, 10 medium, 10 hard, 10 easy, 10 medium, 10 hard, 10 medium, 10 hard, 10 easy, 10 hard

recruited and paid $0.08 to give a difficulty rating to each task or image pair. Figure 7 shows a sample difficulty rating task. Each task was rated by 5 workers. We derived the difficulty score of each task from the average ratings it received. We then ordered the tasks according to their difficulty score and categorized them as follows: easy (tasks 1-40), medium (tasks 41-80), and hard (tasks 81-120).

Exp. 1 Flow. Figure 8 illustrate the flow of Exp.1. First, a worker takes the pre-assessment test, consisting of 5 easy, 5 medium, and 5 hard tasks shown in random order. Next, a worker is assigned to a treatment group (NoOrder, TotalOrder, PartialOrder, and Alternate) and takes the corresponding treatment test. There are 30 workers in a group. The configuration of the treatment tests is specified in Table 2. After completing the treatment test, the worker takes the post-assessment test, which is similar to the pre-assessment test but with different questions.

Since completing a large number of tasks may lead to a drop in performance, we wanted to see if the number of tasks would affect the learning improvement and workers' performance. Thus, we conducted two experiments: one with 12 tasks and another with 120 tasks. Table 2 provides more details.

Table 3. Results of 12-task Experiment

Learning Strategy	Quality	Throughput (tasks/min)	Skill Improvement
NoOrder	**0.93**	7.85	0.12
TotalOrder	0.90	**7.90**	0.10
PartialOrder	0.90	4.93	0.04
Alternate	0.80	6.10	**0.15**

Table 4. Results of 120-task Experiment

Learning Strategy	Quality	Throughput (tasks/min)	Skill Improvement
NoOrder	0.77	**17.58**	0.09
TotalOrder	0.81	15.19	0.02
PartialOrder	**0.88**	14.79	0.09
Alternate	0.81	16.82	**0.18**

In the 12-task experiment, 120 workers with 100% HIT approval rate contributed. Each worker was paid a total of $0.90 for completing all the tests for an estimated duration of 9 minutes. In the 120-task experiment, another 120 workers with 100% HIT approval rate contributed. Each worker was paid a total of $3.90 for completing all the tests for an estimated duration of 39 minutes. We recruited a total of 240 workers, 120 for the 12-task experiment, and 120 for the 120-task experiment. This sample size enables us to observe 95% confidence level and 10% margin of error based on the Central Limit Theorem [51].

Exp. 1 Evaluation. We recorded the scores workers obtained in pre-assessment, treatment, post assessment and the number of tasks completed per minute (throughput). From the requester viewpoint, the measures are the quality of aggregated results and overall throughput. From the worker viewpoint, the measure is individual skill improvement. Based on the workers' scores, we calculated their skill improvement as follows:

$$\frac{post\ assessment\ score - pre\ assessment\ score}{pre\ assessment\ score}$$

Exp. 1 Results. Tables 3 and 4 summarizes results. We observed that all treatments are effective in helping workers improve their skill. In particular, Alternate yielded the highest average skill improvement in both 12-task and 120-task experiments. The results are statistically significant with $p = 0.10$ based on one-way Analysis of Variance (ANOVA) test.

In the 12-task experiment, workers obtained the highest average quality score in NoOrder. Additionally, the highest task throughput was observed in TotalOrder. In the 120-task experiment, workers obtained the highest average quality score in PartialOrder and the highest task throughput in NoOrder.

We further analyzed workers' skill improvement based on their original skill level. Based on their scores in the pre-assessment test, we classified workers into novice, intermediate, and expert. Figures 9 and 10 show the average skill improvement per group. We can see that Alternate is the best strategy for both novice and intermediate workers while there is a ceiling effect for expert workers.

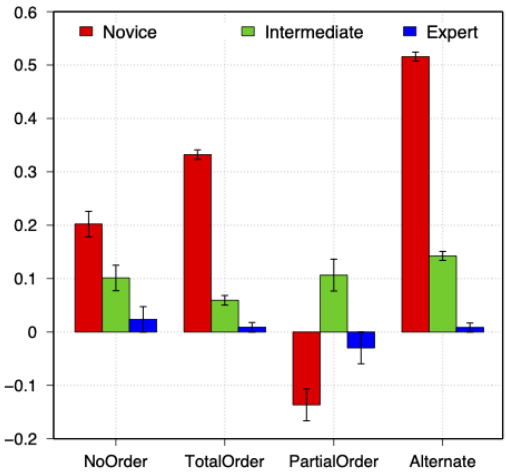

Fig. 9. Skill Improvement in the 12-task Experiment

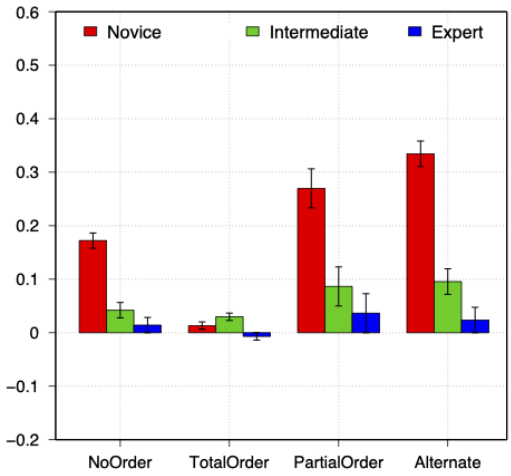

Fig. 10. Skill Improvement in the 120-task Experiment

5.2 Exp. 2: Interleaving collaborative tasks and individual tasks as a learning strategy

We also investigated how workers learn through pure collaborative tasks (CTs) or in combination with individual tasks. We conducted an experiment for novice and intermediate workers that compares two learning strategies. In the first strategy (Figure 11), we asked workers to complete a series of CTs. In the second one (Figure 12), we asked them to complete CTs interleaved with individual tasks.

Exp. 2 Tasks. We designed a collaborative and individual version of text editing task, which asks workers to correct spelling and grammar errors of English paragraphs. Each paragraph has an average of 50 words. In the collaborative version, workers are able to see answers of higher-skilled workers. From the sample task in Figure 11, we can see that *Worker 2* can see the answer of *Worker 1* and has the option to simply edit the existing answer. In this setting, the collaborator has always a relatively higher skill. In the individual version, workers do not see answers of a higher skilled worker and have to work on the task independently.

Exp. 2 Flow. First, a worker takes a pre-assessment test to measure his/her English language skills through an English grammar and vocabulary test (15 items). In the test, given 4 sentences, the worker must select the one with correct grammar and vocabulary usage. Next, a worker is classified as novice-intermediate if he/she answered $\geq$ 90% of the test correctly. Novice-intermediate workers then work using either the *CTs only* strategy (Figure 11) or the *interleaved* strategy (Figure 12). Workers who correctly answer $\geq$ 90% of the test are classified as expert workers.

In the *CTs only* strategy, workers were asked to complete 6 collaborative tasks. In the *interleaved* strategy, workers were asked to complete CTs interleaved with individual tasks. Tasks were presented to them in the following order: 2 CTs, 1 individual task, 2 CTs, 1 individual task, 2 CTs, 1 individual task. Lastly, novice-intermediate workers take a post assessment test, which is similar to the pre-assessment test but with different questions.

We asked AMT 400 workers to take the pre-assessment test: 214 were experts and 186 novice-intermediate workers. We then asked an expert worker to complete 6 text-editing tasks and used the worker's answers as input in all the CTs.

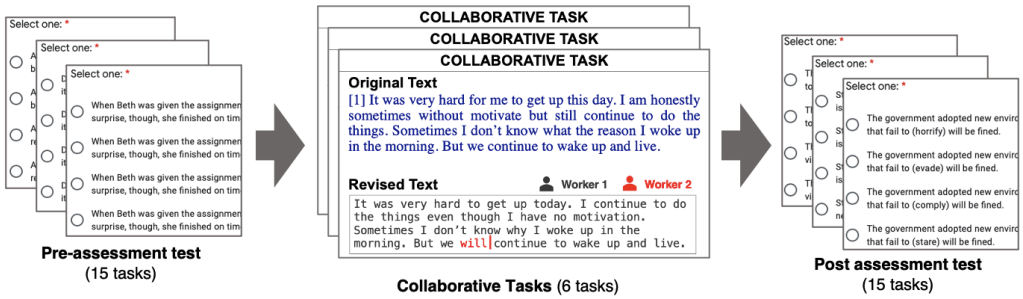

Fig. 11. Flow of Collaborative Task Learning Experiment (Collaborative Tasks Only)

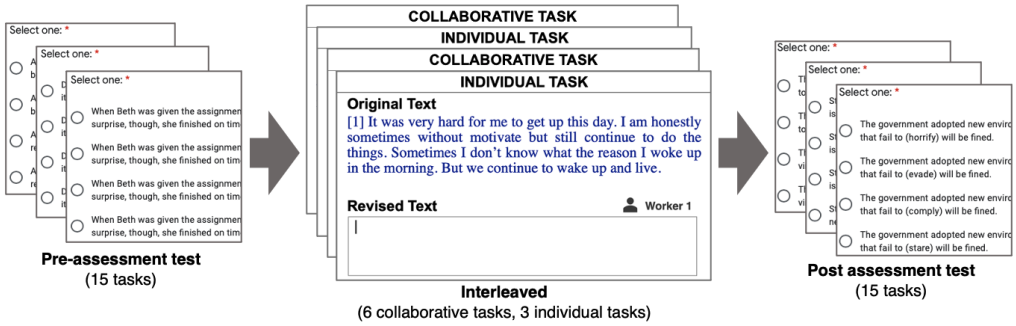

Fig. 12. Flow of Collaborative Task Learning Experiment (Interleaved Collaborative and Individual tasks)

We invited novice-intermediate workers to complete tasks using the *CTs only* strategy (Figure 11) or the *interleaved* strategy (Figure 12) and received 70 valid responses (35 per strategy). All workers had a HIT approval rate of 100%. Workers who used the CTs only strategy completed the tasks for an estimated duration of 22 minutes and were paid a total of $2.20. Those who used the interleaved strategy completed the tasks for an estimated duration of 27 minutes and were paid a total of $2.70.

We recorded task throughput, average quality, and skill improvement. To measure answer quality, we used an online grammar checking tool, Grammarly[3] and recorded the overall score given by the tool. Skill improvement is computed as in the Exp. 1.

Exp. 2 Results. The results for Exp. 2. are summarized on Table 5. The observations on skill improvement and throughput are statistically significant based on a one-way ANOVA, where p = 0.10. While the **quality** observed in both cases are not significantly different, the quality score was higher in the *CTs only*. We examined this further and noted that the quality of CTs in both cases are similar. However, in the *interleaved* case, the individual tasks have lower quality scores that affected the overall quality of the *interleaved* case.

For example, we look at *Worker 1* who performed tasks in the *interleaved* case. The average quality of the 6 CTs performed by *Worker 1* is 0.97. However, the average quality of the 3 individual tasks he/she performed is only 0.70. As a result, his/her overall quality is only 0.88.

[3]www.grammarly.com

Table 5. Comparison of Collaborative Tasks Only vs. Interleaved Collaborative and Individual Tasks

Learning Strategy	Quality	Throughput (tasks/min)	Skill Improvement
Collaborative tasks only	0.93	0.28	0.06
Interleaved collaborative and individual tasks	0.88	**0.34**	**0.42**

On the other hand, **skill improvement** is significantly higher in the *interleaved* case compared to the *CTs only* case. *Worker 1* is one of the workers whose skill improved. Initially, he/she obtained a score of 0.60 in the pre-assessment test then eventually obtained a score of 0.93 in the post-assessment test. Using the same formula for skill improvement in Exp. 1, *Worker 1*'s skill improvement score is 0.56.

The higher skill improvement in the *interleaved* case may be attributed to the fact that individual tasks are similar to CTs, which may have contributed to the learning of the worker. Moreover, in the case of *CTs only*, since there are already answers from expert workers, the novice-intermediate workers may have become under-challenged, resulting in a lower skill improvement.

Throughput is also observed to be higher in the *interleaved* case. We can conjecture that as skills improve, workers become more proficient and faster.

In the future, we need a further investigation comparing to individual tasks only or collaborative tasks only with the same number of tasks. We will also need to investigate an impact of different types of individual tasks on learning when interleaved with collaborative tasks.

6 CONCLUSION AND PERSPECTIVES

In this paper, we developed task assignment and completion strategies that implement learning strategies that are grounded in existing theories. We showed that optimizing task assignment toward workers' learning has a positive effect on skill improvement, task throughput and contribution quality in image classification and text editing tasks. In particular, strategies that alternate task difficulties achieve the best balance between worker-centric and platform-centric goals.

6.1 Generalization to other task types

We have shown that our results are beneficial for citizen sciences, where we often see long-term commitment of workers. While our experiments focused on image classification and text editing tasks, they are applicable to some of the tasks in Bloom's "Knowledge and Comprehension classes" such as labeling, counting, transcription, and spelling and grammar [5], because they have a ground truth where workers can improve their skills by completing individual tasks, or learn from each other in collaborative tasks.

It is however unclear whether our framework can be applied to tasks in Bloom's "Evaluation and Synthesis class" such as design, creation, examination, and critique. As those tasks require more advanced skills than knowledge and comprehension class tasks, rich interaction is expected, and is likely to be necessary to improve skills [2, 25, 36, 59]. Since we cannot avoid additional cost required by rich interaction, how to balance platform-centric and worker-centric needs for such tasks is challenging.

6.2 Toward a more expressive framework

We are currently exploring several avenues to make our framework more expressive. Regarding learning potential, our main assumption was that learning depends on a worker's performance factor which is modeled as the difference between expected and observed performance. In our new formulation, we seek to design holistic workflows that balance learning and productivity. In [56], a workflow called CrowdSCIM provides high learning gain but completion time is significantly longer (10 min. more) than other methods. As a result, deciding which workflow is better could be formulated as an optimization objective that combines learning and productivity.

Hettiachchi proposed assessing worker's cognitive skills and subsequent assignment based on strong cognitive skills improves contribution quality [26]. This could potentially be combined to further improve both worker satisfaction and task outcomes. This formulation can be leveraged to study long-term effects of learning by deploying tasks during an extended period, and measuring worker satisfaction, retention, and performance over time.

We would also like to study the impact of revealing other workers' contributions on the learner's response [42]. This priming can improve crowdwork quality for task types with objective answers [20], but in tasks that require subjective answers, this may introduce a selection bias and reduce diversity. In this context, we would like to investigate how enabling self-correction in which workers are allowed to change their contribution after seeing other responses [50], impacts learning.

To enable the above, we need to revisit our formulation to assign tasks to several workers at a time. This raises new computational problems [48] that need to account for the evolution of skills of multiple workers at a time. In particular, this will impact collaborative task assignment that will need to be adaptive and account for a holistic treatment of workers.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work was partially supported by JST CREST under Grant No.: JPMJCR16E3 and AIP challenge program, and JSPS KAKENHI under Grant No.: JP21H03552.

REFERENCES

[1] Maha Alsayasneh, Sihem Amer-Yahia, Éric Gaussier, Vincent Leroy, Julien Pilourdault, Ria Mae Borromeo, Motomichi Toyama, and Jean-Michel Renders. 2018. Personalized and Diverse Task Composition in Crowdsourcing. *IEEE Trans. Knowl. Data Eng.* 30, 1 (2018), 128–141.

[2] Michael Anderson. 2011. Crowdsourcing higher education: A design proposal for distributed learning. *MERLOT Journal of Online Learning and Teaching* 7, 4 (2011), 576–590.

[3] Frank B Baker and Seock-Ho Kim. 2004. *Item response theory: Parameter estimation techniques.* CRC Press.

[4] Ashok R Basawapatna, Alexander Repenning, Kyu Han Koh, and Hilarie Nickerson. 2013. The zones of proximal flow: guiding students through a space of computational thinking skills and challenges. In *Proceedings of the ninth annual international ACM conference on International computing education research.* 67–74.

[5] Benjamin S Bloom. 1956. Taxonomy of educational objectives. Vol. 1: Cognitive domain. *New York: McKay* (1956), 20–24.

[6] Kenneth A Bruffee. 1999. *Collaborative learning: Higher education, interdependence, and the authority of knowledge.* ERIC.

[7] Alec Burmania, Srinivas Parthasarathy, and Carlos Busso. 2015. Increasing the reliability of crowdsourcing evaluations using online quality assessment. *IEEE Transactions on Affective Computing* 7, 4 (2015), 374–388.

[8] Carrie J Cai, Shamsi T Iqbal, and Jaime Teevan. 2016. Chain reactions: The impact of order on microtask chains. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.* ACM, 3143–3154.

[9] Chandra Chekuri and Sanjeev Khanna. 2006. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM J. COMPUT* 38, 3 (2006), 1.

[10] Chun-Wei Chiang, Anna Kasunic, and Saiph Savage. 2018. Crowd Coach: Peer Coaching for Crowd Workers' Skill Growth. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–17.

[11] Derrick Coetzee, Seongtaek Lim, Armando Fox, Bjorn Hartmann, and Marti A Hearst. 2015. Structuring interactions for large-scale synchronous peer learning. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. 1139–1152.

[12] Mihaly Csikszentmihalyi. 1975. *Beyond boredom and anxiety: The experience of play in work and games.* Jossey-Bass.

[13] Peng Dai, Jeffrey M Rzeszotarski, Praveen Paritosh, and Ed H Chi. 2015. And now for something completely different: Improving crowdsourcing workflows with micro-diversions. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 628–638.

[14] Dan Davis, Guanliang Chen, Claudia Hauff, and Geert-Jan Houben. 2018. Activating learning at scale: A review of innovations in online learning strategies. *Computers & Education* 125 (2018), 327–344.

[15] Leo J De Vin, Lasse Jacobsson, JanErik Odhe, and Anders Wickberg. 2017. Lean Production Training for the Manufacturing Industry: Experiences from Karlstad Lean Factory. *Procedia Manufacturing* 11 (2017), 1019–1026.

[16] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, and Philippe Cudré-Mauroux. 2014. Scaling-up the crowd: Micro-task pricing schemes for worker retention and latency improvement. In *Second AAAI Conference on Human Computation and Crowdsourcing*.

[17] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2016. Scheduling human intelligence tasks in multi-tenant crowd-powered systems. In *Proceedings of the 25th international conference on World Wide Web*. 855–865.

[18] Mira Dontcheva, Robert R Morris, Joel R Brandt, and Elizabeth M Gerber. 2014. Combining crowdsourcing and learning to improve engagement and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3379–3388.

[19] Shayan Doroudi, Ece Kamar, and Emma Brunskill. 2019. Not Everyone Writes Good Examples but Good Examples Can Come from Anywhere. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 7. 12–21.

[20] Steven Dow, Anand Kulkarni, Scott Klemmer, and Björn Hartmann. 2012. Shepherding the crowd yields better work. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*. 1013–1022.

[21] Ryan Drapeau, Lydia B Chilton, Jonathan Bragg, and Daniel S Weld. 2016. Microtalk: Using argumentation to improve crowdsourcing accuracy. In *Fourth AAAI Conference on Human Computation and Crowdsourcing*.

[22] Mohammadreza Esfandiari, Dong Wei, Sihem Amer-Yahia, and Senjuti Basu Roy. 2019. Optimizing Peer Learning in Online Groups with Affinities. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1216–1226.

[23] Ujwal Gadiraju and Stefan Dietze. 2017. Improving learning through achievement priming in crowdsourced information finding microtasks. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*. ACM, 105–114.

[24] Ujwal Gadiraju, Besnik Fetahu, and Ricardo Kawase. 2015. Training workers for improving performance in crowdsourcing microtasks. In *Design for Teaching and Learning in a Networked World*. Springer, 100–114.

[25] Michael D Greenberg, Matthew W Easterday, and Elizabeth M Gerber. 2015. Critiki: A scaffolded approach to gathering design feedback from paid crowdworkers. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*. 235–244.

[26] Danula Hettiachchi, Niels Van Berkel, Vassilis Kostakos, and Jorge Goncalves. 2020. CrowdCog: A Cognitive skill based system for heterogeneous task assignment and recommendation in crowdsourcing. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW2 (2020), 1–22.

[27] Kazushi Ikeda and Keiichiro Hoashi. 2017. Crowdsourcing Go: Effect of worker situation on mobile crowdsourcing performance. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1142–1153.

[28] Nicolas Kaufmann, Thimo Schulze, and Daniel Veit. 2011. More than fun and money. Worker Motivation in Crowdsourcing-A Study on Mechanical Turk.. In *AMCIS*, Vol. 11. Detroit, Michigan, USA, 1–11.

[29] Juho Kim. 2015. *Learnersourcing: improving learning with collective learner activity.* Ph.D. Dissertation. Massachusetts Institute of Technology.

[30] Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S Bernstein. 2017. Mechanical novel: Crowdsourcing complex work through reflection and revision. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. 233–245.

[31] Masaki Kobayashi, Hiromi Morita, Masaki Matsubara, Nobuyuki Shimizu, and Atsuyuki Morishima. 2018. An empirical study on short-and long-term effects of self-correction in crowdsourced microtasks. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*.

[32] Masaki Kobayashi, Hiromi Morita, Masaki Matsubara, Nobuyuki Shimizu, and Atsuyuki Morishima. 2021. Empirical Study on Effects of Self-Correction in Crowdsourced Microtasks. *Human Computation* 8 (2021).

[33] David A Kolb. 1984. *Experiential learning: Experience as the source of learning and development.* Englewood Cliffs.

[34] Martijn Koops and Martijn Hoevenaar. 2013. Conceptual change during a serious game: Using a Lemniscate model to compare strategies in a physics game. *Simulation & Gaming* 44, 4 (2013), 544–561.

[35] David R Krathwohl and Lorin W Anderson. 2009. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives.* Longman.

[36] Markus Krause, Tom Garncarz, JiaoJiao Song, Elizabeth M Gerber, Brian P Bailey, and Steven P Dow. 2017. Critique style guide: Improving crowdsourced design feedback with a natural language model. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 4627–4639.

[37] Katsumi Kumai, Masaki Matsubara, Yuhki Shiraishi, Daisuke Wakatsuki, Jianwei Zhang, Takeaki Shionome, Hiroyuki Kitagawa, and Atsuyuki Morishima. 2018. Skill-and-stress-aware assignment of crowd-worker groups to task streams. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*.

[38] Suna Kyun, Slava Kalyuga, and John Sweller. 2013. The effect of worked examples when learning to write essays in English literature. *The Journal of Experimental Education* 81, 3 (2013), 385–408.

[39] Susanne P Lajoie and Alan Lesgold. 1989. Apprenticeship training in the workplace: Computer-coached practice environment as a new form of apprenticeship. *Machine-mediated learning* 3, 1 (1989), 7–28.

[40] Thomas D LaToza, Arturo Di Lecce, Fabio Ricci, W Ben Towne, and André Van Der Hoek. 2015. Ask the crowd: Scaffolding coordination and knowledge sharing in microtask programming. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 23–27.

[41] Jean Lave and Etienne Wenger. 1991. *Situated learning: Legitimate peripheral participation*. Cambridge university press.

[42] Lena Mamykina, Thomas N Smyth, Jill P Dimond, and Krzysztof Z Gajos. 2016. Learning from the crowd: Observational learning in crowdsourcing communities. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2635–2644.

[43] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*. 897–908.

[44] Jacob Mincer. 1962. On-the-job training: Costs, returns, and some implications. *Journal of political Economy* 70, 5, Part 2 (1962), 50–79.

[45] Nilesh Padhariya and Kshama Raichura. 2014. Crowdlearning: An incentive-based learning platform for crowd. In *2014 Seventh International Conference on Contemporary Computing (IC3)*. IEEE, 44–49.

[46] Participants of Shonan Future-of-Work Workshop. 2020. Data Management Research and More: Making AI Machines Work for Humans in FoW. In *ACM SIGMOD Record (to appear)*.

[47] Julien Pilourdault, Sihem Amer-Yahia, Dongwon Lee, and Senjuti Roy. 2017. Motivation-aware task assignment in crowdsourcing. In *EDBT*.

[48] Julien Pilourdault, Sihem Amer-Yahia, Senjuti Basu Roy, and Dongwon Lee. 2018. Task relevance and diversity as worker motivation in crowdsourcing. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 365–376.

[49] Jeffrey M Rzeszotarski, Ed Chi, Praveen Paritosh, and Peng Dai. 2013. Inserting micro-breaks into crowdsourcing workflows. In *First AAAI Conference on Human Computation and Crowdsourcing*.

[50] Nihar Shah and Dengyong Zhou. 2016. No oops, you won't do it again: Mechanisms for self-correction in crowdsourcing. In *International conference on machine learning*. 1–10.

[51] SurveyMonkey. [n.d.]. Calculating the Number of Respondents You Need. https://help.surveymonkey.com/articles/en_US/kb/How-many-respondents-do-I-need.

[52] Ryo Suzuki, Niloufar Salehi, Michelle S Lam, Juan C Marroquin, and Michael S Bernstein. 2016. Atelier: Repurposing expert crowdsourcing tasks as micro-internships. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2645–2656.

[53] Mohammad Taheri, Nasser Sherkat, Nick Shopland, Dorothea Tsatsou, Enrique Hortal Nicholas Vretos, Christos Athanasiadis, and Penny Standen. 2017. Adaptation and Personalization principles based on MaThiSiS findings. In *Public report on Managing Affective-learning THrough Intelligent atoms and Smart InteractionS project*.

[54] Kazutoshi Umemoto, Tova Milo, and Masaru Kitsuregawa. 2020. Toward Recommendation for Upskilling: Modeling Skill Improvement and Item Difficulty in Action Sequences. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 169–180.

[55] Lev Vygotsky. 1987. Zone of proximal development. *Mind in society: The development of higher psychological processes* 5291 (1987), 157.

[56] Nai-Ching Wang, David Hicks, and Kurt Luther. 2018. Exploring trade-offs between learning and productivity in crowdsourced history. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–24.

[57] Joseph Jay Williams, Juho Kim, Anna Rafferty, Samuel Maldonado, Krzysztof Z Gajos, Walter S Lasecki, and Neil Heffernan. 2016. Axis: Generating explanations at scale with learnersourcing and machine learning. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*. 379–388.

[58] Amy S Wu, Rob Farrell, and Mark K Singley. 2002. Scaffolding group learning in a collaborative networked environment. (2002).

[59] Alvin Yuan, Kurt Luther, Markus Krause, Sophie Isabel Vennix, Steven P Dow, and Bjorn Hartmann. 2016. Almost an expert: The effects of rubrics and expertise on perceived value of crowdsourced design critiques. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. 1005–1017.

[60] Haiyi Zhu, Steven P Dow, Robert E Kraut, and Aniket Kittur. 2014. Reviewing versus doing: Learning and performance in crowd assessment. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. 1445–1455.

Received October 2020; revised April 2021; revised July 2021; accepted July 2021