An Online Learning Approach to Improve the Quality of Crowdsourcing

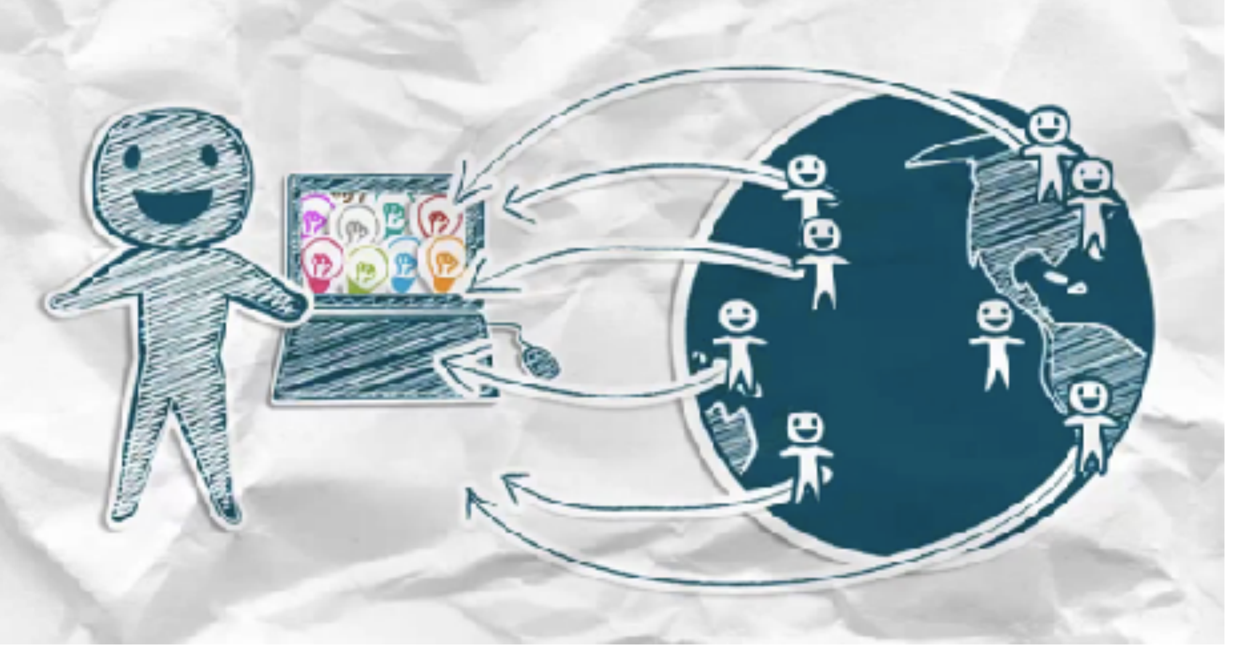
Yang Liu, Mingyan Liu

Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor

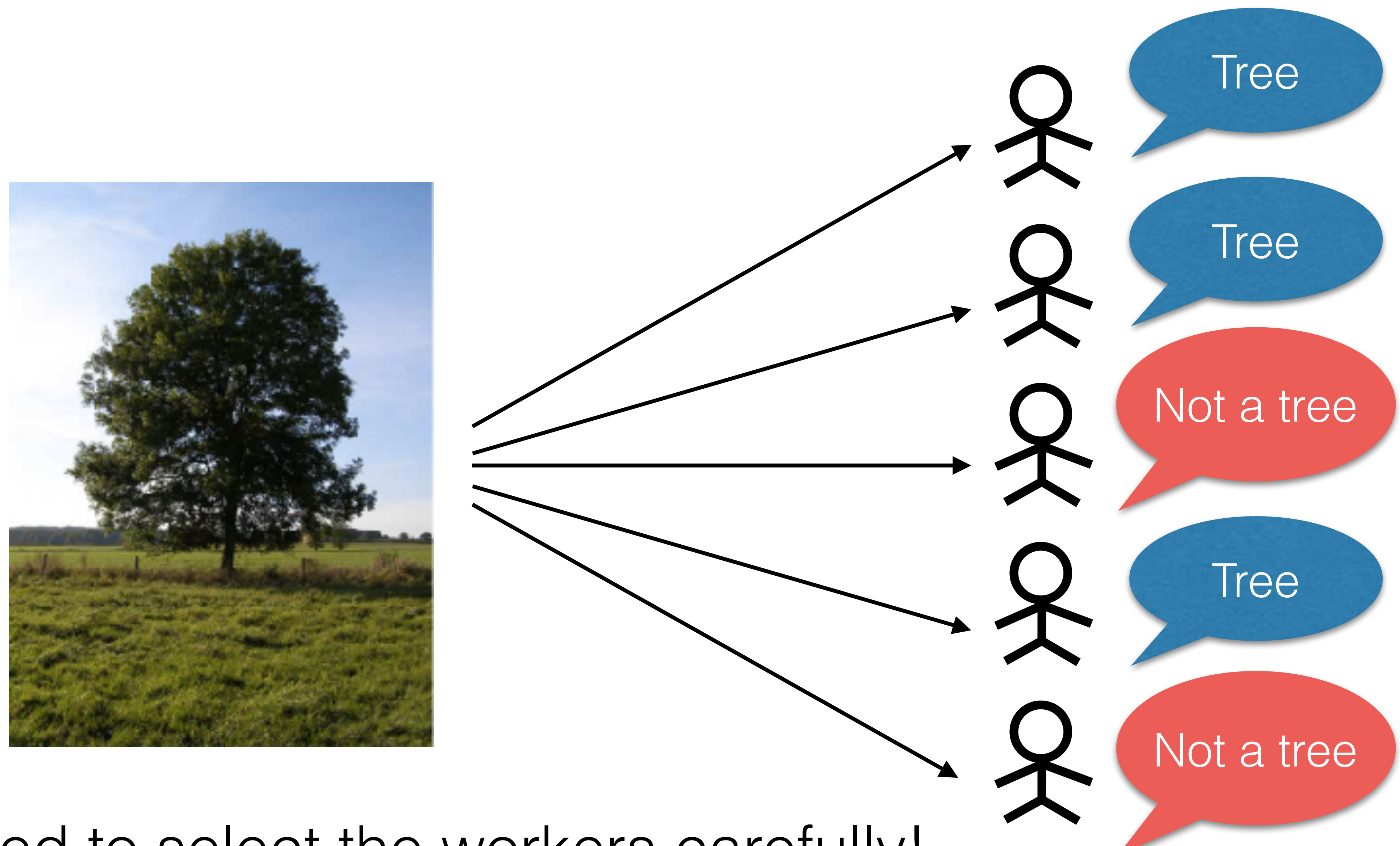
SIGMETRICS' 15

Crowdsourcing

- A requester post micro-tasks on crowdsourcing platform
  - E.g., determine whether an image has a tree
- A diverse population of workers give labels for a subset of tasks in exchange for payment



Diverse Qualities of Workers



- Need to select the workers carefully!
- Can we use MAB?

Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Pull a set of arms
  - Receive a reward

Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Pull a set of arms
  - Receive a reward

Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Pull a set of arms
  - Receive a reward

Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Pull a set of arms
  - Receive a reward

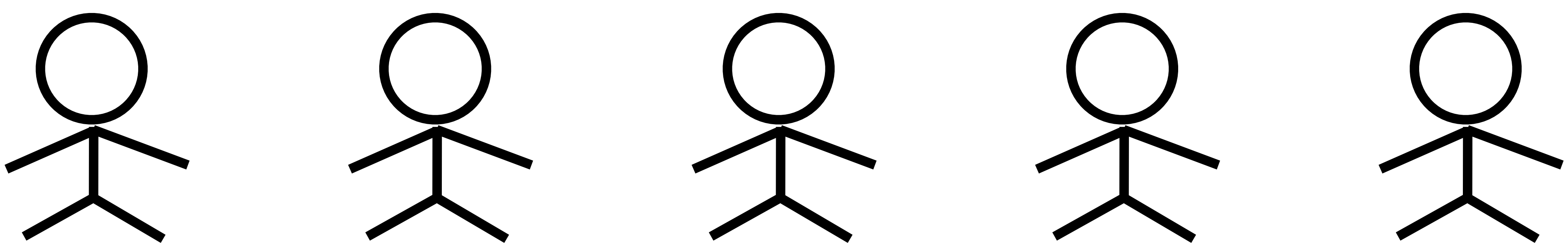
Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Pull a set of arms
  - Receive a reward

How to find an optimal set of arms?

Multi-armed Bandit (MAB) Problem

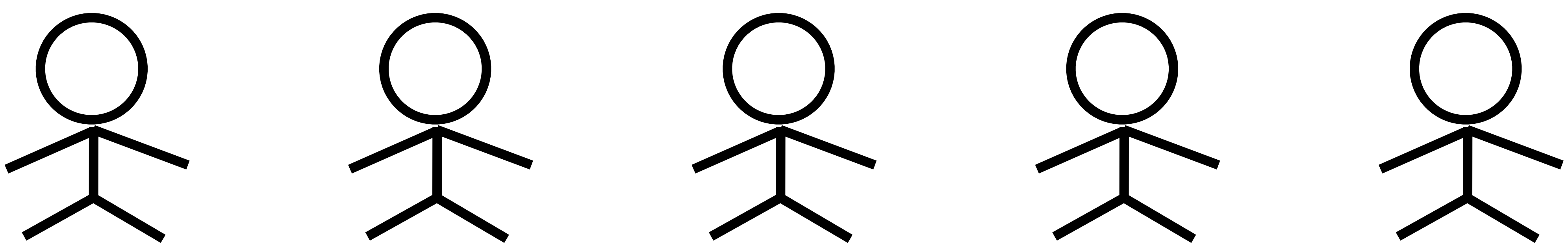
- In each time step $t = 1, 2, ..., T$:
  - Select a set of workers
  - Receive a reward



How to find an optimal set of workers?

Multi-armed Bandit (MAB) Problem

- In each time step $t = 1, 2, ..., T$:
  - Select a set of workers
  - ~~Receive a reward~~



How to find an optimal set of workers?

Crowdsourcing vs. MAB

- In crowdsourcing:
  - Data is unlabelled to begin with
  - A particular choice of workers leads to unknown quality (reward) of their labelling outcome
- In MAB:
  - A reward is known instantaneously following a selection of arms

The Worker Model

- Set of workers: $\mathfrak{M} = \{1, 2, ..., M\}$
- Probability that worker i gives a true label: p_i

Quality of worker

- Assumptions:

1. $p_i \neq p_j, \forall i \neq j$ — Workers are different
2. $\bar{p} := \sum_{I=1}^{M} \frac{p_i}{M} > \frac{1}{2}$ — Workers are good on average
3. $M > \frac{\log 2}{2(\bar{p} - 1/2)^2}$ — Justify it later

The Crowdsourcing Model

- In each time step $t = 1, 2, ..., T$:
  - A task $k \in \mathfrak{K}$ arrives
  - The user selects a subset of workers $S_t \subseteq \mathfrak{M}$
  - Worker $i \in S_t$ a label $L_i(t) \in \{0, 1\}$ for task k

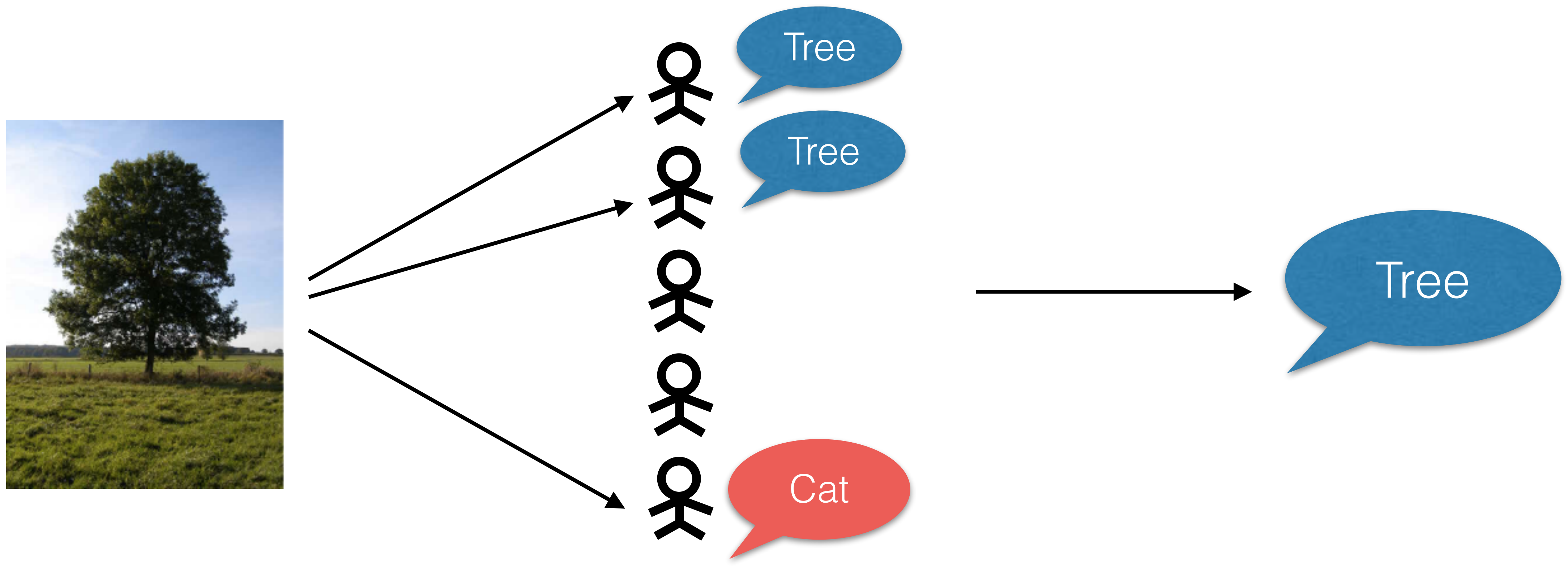
The Crowdsourcing Model

- In each time step $t = 1, 2, ..., T$:
  - A task $k \in \mathfrak{K}$ arrives
  - The user selects a subset of workers $S_t \subseteq \mathfrak{M}$
  - Worker $i \in S_t$ a label $L_i(t) \in \{0, 1\}$ for task k
  - How to aggregate the set of labels, $\{L_i(t)\}_{i \in S_t}$?

The Crowdsourcing Model

- Use majority vote to aggregate labels

$$L^*(t) = \text{argmax}_{l \in \{0,1\}} \sum_{i \in S_t} I_{L_i(t)=l}$$



The Crowdsourcing Model

- Probability of obtaining the correct label

$$\pi(S_t) = \underbrace{\sum_{S:S\subseteq S_t, |S|\geq \lceil \frac{|S_t|+1}{2} \rceil} \prod_{i\in S} p_i \cdot \prod_{j\in S_t\setminus S} (1-p_j)}_{\text{Majority gives the correct label}}$$

$$+ \underbrace{\frac{\sum_{S:S\subseteq S_t, |S|=\frac{|S_t|}{2}} \prod_{i\in S} p_i \cdot \prod_{j\in S_t\setminus S} (1-p_j)}{2}}_{\text{Ties broken equally likely}}$$

The Crowdsourcing Model

- Probability of obtaining the correct label

$$\pi(S_t) = \sum_{S: S \subseteq S_t, |S| \geq \lceil \frac{|S_t|+1}{2} \rceil} \prod_{i \in S} p_i \cdot \prod_{j \in S_t \setminus S} (1 - p_j)$$

Majority gives the correct label

$$+ \frac{\sum_{S: S \subseteq S_t, |S| = \frac{|S_t|}{2}} \prod_{i \in S} p_i \cdot \prod_{j \in S_t \setminus S} (1 - p_j)}{2}$$

Ties broken equally likely

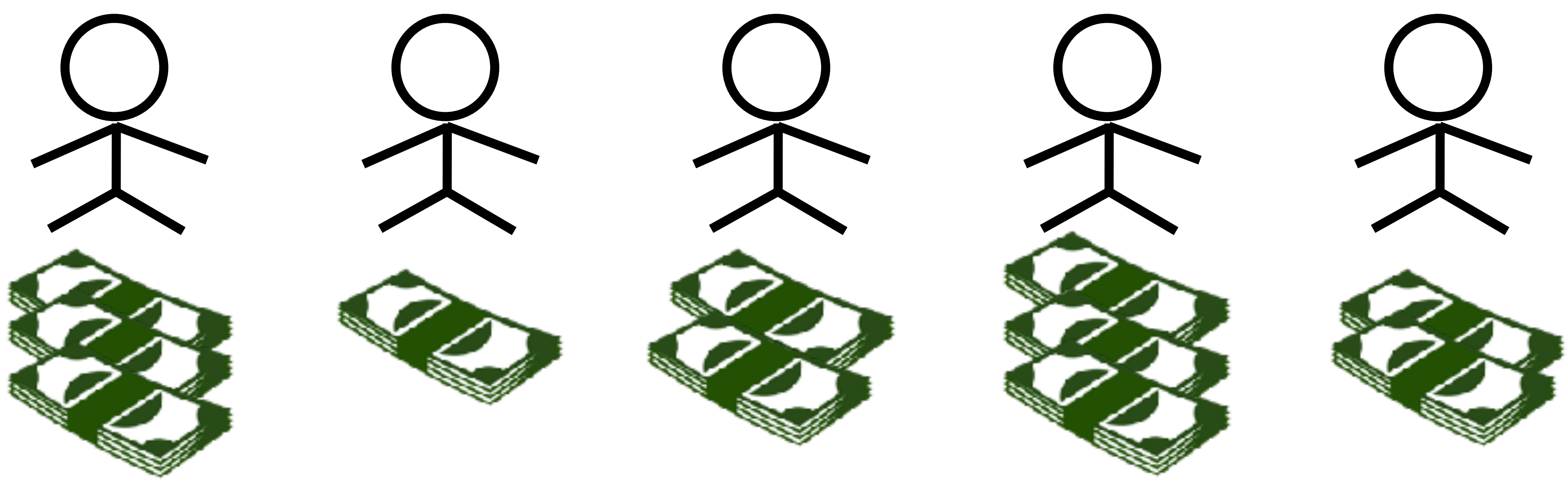
- Optimal selection of workers given worker qualities

$$S^* = \mathrm{argmax}_{S \subseteq \mathfrak{M}} \pi(S)$$

The Crowdsourcing Model

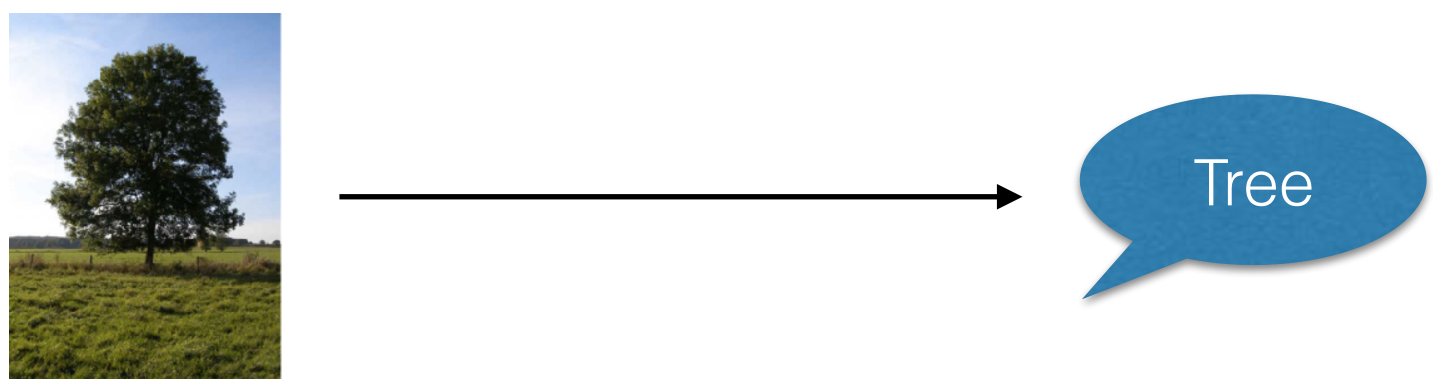
- Cost per task of worker i : c_i
- Cost per task for a set of workers:

$$\mathfrak{C}(S) = \sum_{i \in S} c_i, S \subseteq \mathfrak{M}$$



The Crowdsourcing Model

- Goal:
  - Obtaining high quality labels



  - Keeping the cost low

Definition of Regret

$$R(T) = T \cdot \underbrace{U(S^*)}_{\text{Quality of the optimal set}} - E[\sum_{t=1}^{T} \underbrace{U(S_t)}_{\text{Quality of the selected set}}]$$

$$R_{\mathfrak{C}}(T) = E[\sum_{t=1}^{T} \underbrace{\mathfrak{C}(S_t)}_{\text{Cost of the selected set}}] - T \cdot \underbrace{\mathfrak{C}(S^*)}_{\text{Cost of the optimal set}}]$$

- Goal: minimise these two regrets

Offline Optimal Selection of Workers

- Suppose we are given all worker qualities
- Select an optimal set of workers by solving

$$S^* = \text{argmax}_{S \subseteq \mathfrak{M}} \pi(S)$$

Offline Optimal Selection of Workers

- Suppose we are given all worker qualities
- Select an optimal set of workers by solving

$$S^* = \mathrm{argmax}_{S \subseteq \mathfrak{M}} \pi(S)$$

- However,
  - The number of possible selections is combinatorial
  - Cannot be solved in polynomial time

Offline Optimal Selection of Workers

THEOREM 1. *Under the simple majority vote rule, the optimal number of labelers* $s^* = |S^*|$ *must be an odd number.*

THEOREM 2. *The optimal set* S^* *is monotonic, i.e., if we have* $i \in S^*$ *and* $j \notin S^*$ *then we must have* $p_i > p_j$.

Offline Optimal Selection of Workers

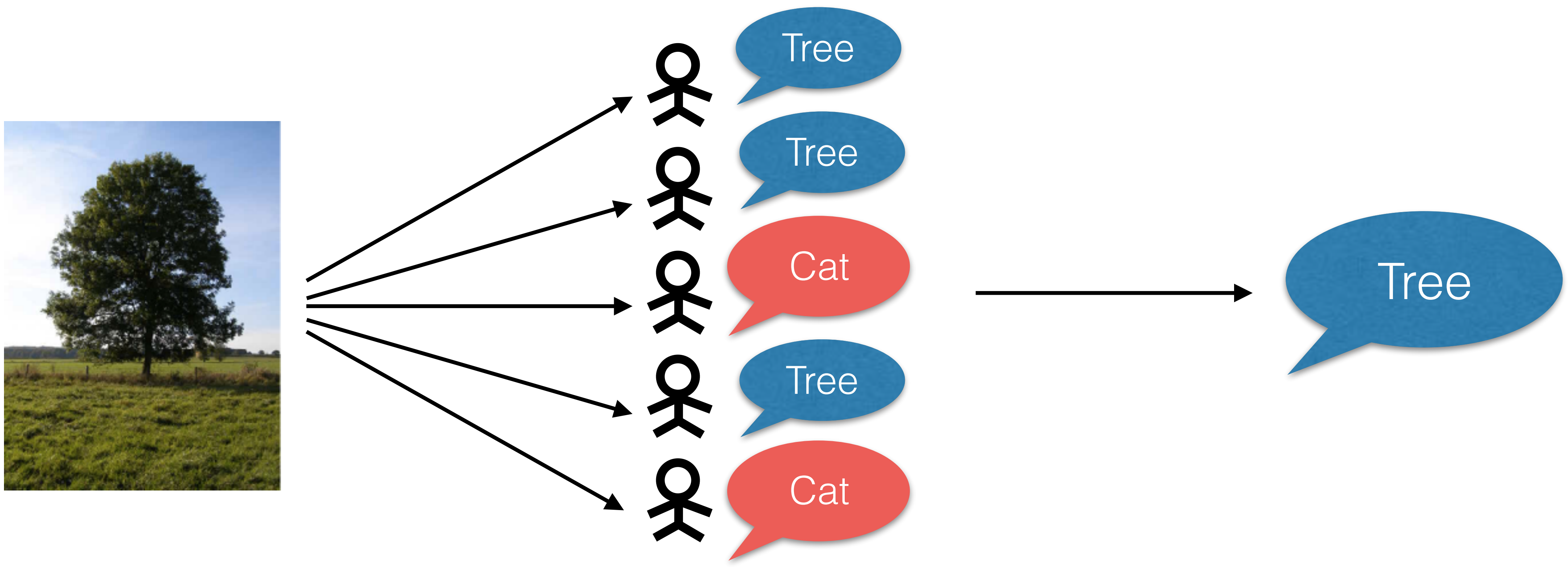
THEOREM 1. *Under the simple majority vote rule, the optimal number of labelers $s^* = |S^*|$ must be an odd number.*

THEOREM 2. *The optimal set S^* is monotonic, i.e., if we have $i \in S^*$ and $j \notin S^*$ then we must have $p_i > p_j$.*

- Optimal selection of workers consists of the top s^* workers
- Only need to compute s^*
- Only need a linear search from 1 to M

Lack of Ground Truth

- Assign a task to all workers
- Learn the ground truth label by majority vote
- How about the accuracy?



Lack of Ground Truth

- Worker i's outcome on a given task: $x_i \sim Bin(p_i, 1)$
- $x_i = 1$ if her label is correct; $x_i = 0$ otherwise
- Prob. that majority vote over M workers is correct:

$$
\begin{aligned}
P(\frac{\sum_{i=1}^{M} x_i}{M} > \frac{1}{2}) &= 1 - P(\frac{\sum_{i=1}^{M} x_i}{M} \leq \frac{1}{2}) \\
&= 1 - P(\frac{\sum_{i=1}^{M} x_i}{M} - \bar{p} \leq \frac{1}{2} - \bar{p}) \\
&\geq 1 - \exp(-2M(\bar{p} - 1/2)^2)
\end{aligned}
$$

Lack of Ground Truth

$$P(\frac{\sum_{i=1}^{M} x_i}{M} > \frac{1}{2}) \geq 1 - \exp(-2M(\bar{p} - 1/2)^2)$$

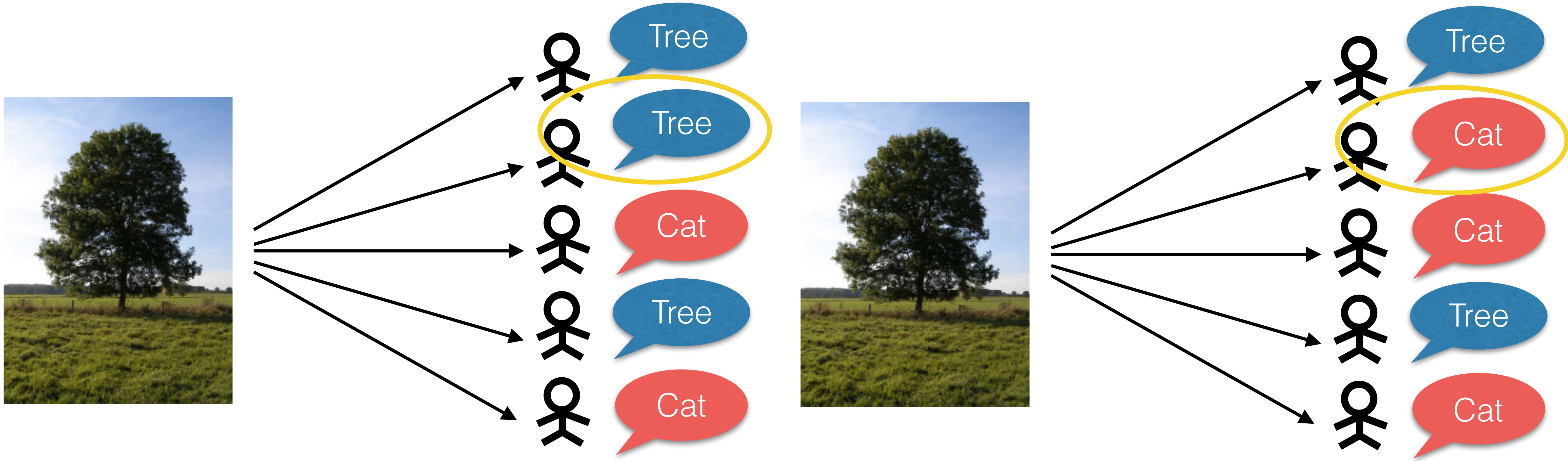
- Since we assume that
- $\bar{p} := \sum_{I=1}^{M} \frac{p_i}{M} > \frac{1}{2}$ and $M > \frac{\log 2}{2(\bar{p} - 1/2)^2}$
- Then $P(\frac{\sum_{i=1}^{M} x_i}{M} > \frac{1}{2}) > \frac{1}{2}$
- Majority vote over M workers will be correct most of the time

Exploration vs. Exploitation

- Exploitation:
  - Assign to an optimal set of worker based on estimated worker qualities
- Exploration:
  - Assign a task to some suboptimal workers to estimate worker qualities
- Need to balance this trade-off

Exploration

- Repeatedly assigning a task (tester) to **all** workers
- Testing whether a worker answers questions randomly or consistently



Exploration

- n-th voting outcome for a tester task k: $y_k(n)$

Exploration

- Denote by $y_k^*(N)$ the label obtained using majority vote over N label outcomes $y_k(1), y_k(2), \ldots, y_k(N)$:

$$y_k^*(N) = \begin{cases} 1, & \frac{\sum_{n=1}^{N} I_{y_k(n)=1}}{N} > 0.5 \\ 0, & \text{otherwise} \end{cases}$$

- This majority label after N tests on a test task will be used to analyse worker qualities

Exploration vs. Exploitation

- Determine whether we should explore or exploit

$$\mathscr{O}(t) = I_{|E(t)| \leq D_1(t) \text{ or } \exists k \in E(t) \text{ s.t. } \hat{N}_k(t) \leq D_2(t)}$$

tasks used for exploration — # times k has been assigned

- where

$$D_1(t) = \frac{1}{\left(\frac{1}{\max_{m:m \text{ odd}} m \cdot n(S^m)} - \alpha\right)^2 \cdot \varepsilon^2} \cdot \log t$$

$$D_2(t) = \frac{1}{(a_{\min} - 0.5)^2} \cdot \log t$$

Online Learning Algorithm

- Initialise all worker quality to some value in $[0, 1]$ uniformly at random

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathscr{M}}$ to some value in $[0, 1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathscr{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathscr{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathscr{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right) .$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathscr{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|} .$$

3: Else if $\mathscr{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathscr{M}} \tilde{\pi}(S) ,$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

- At time t, a new task k arrives:
  - Determine whether we should explore or exploit
  - If $\mathcal{O}(t) = 1$, we explore
  - Else, we exploit

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathcal{M}}$ to some value in $[0,1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathcal{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathcal{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathcal{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right).$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathcal{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|}.$$

3: Else if $\mathcal{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathcal{M}} \tilde{\pi}(S),$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\bar{p}_i\}_{i \in \mathcal{M}}$ to some value in $[0,1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathcal{O}(t) = 1$, the algorithm explores.

Exploration

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathcal{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathcal{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right) .$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathcal{M}$:

$$\bar{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|} .$$

3: Else if $\mathcal{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathcal{M}} \tilde{\pi}(S) ,$$

which is solved using the linear search property, but with the current estimates $\{\bar{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

- If all the tester tasks have been assigned sufficiently
  - Assign the incoming task to all workers
- Else if there exist tester tasks have been assigned insufficiently
  - Randomly select one of them
  - Assign it to all workers

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathscr{M}}$ to some value in $[0, 1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathscr{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathscr{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathscr{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right) .$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathscr{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|} .$$

3: Else if $\mathscr{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathscr{M}} \tilde{\pi}(S) ,$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

- Majority vote over $\hat{N}_k(t)$ label outcomes

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right)$$

Estimated ground truth label of task k

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathcal{M}}$ to some value in $[0, 1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathcal{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathcal{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathcal{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right).$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathcal{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|}.$$

3: Else if $\mathcal{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathcal{M}} \tilde{\pi}(S),$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

- Update workers' quality estimation

labels consistent with estimated ground truth

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|}$$

labels given by worker i

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathcal{M}}$ to some value in $[0,1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathcal{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathcal{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathcal{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5) .$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathcal{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|} .$$

3: Else if $\mathcal{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathcal{M}} \tilde{\pi}(S) ,$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathscr{M}}$ to some value in $[0,1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathscr{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathscr{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathscr{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right) .$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathscr{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|} .$$

3: Else if $\mathscr{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathscr{M}} \tilde{\pi}(S) ,$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

Exploitation

4: Set $t = t + 1$ and go to Step 2.

Online Learning Algorithm

- Find an optimal set of worker

  $$S_t = \text{argmax}_{S \subseteq \mathfrak{M}} \underbrace{\tilde{\pi}(S)}$$

  Estimated quality of worker set S

- Search a set with highest estimated quality

- Assign task k to S_t

1: Initialization at $t = 0$: Initialize the estimated accuracy $\{\tilde{p}_i\}_{i \in \mathcal{M}}$ to some value in $[0, 1]$; denote the initialization task as k, set $E(t) = \{k\}$ and $\hat{N}_k(t) = 1$.

2: At time t a new task arrives: If $\mathcal{O}(t) = 1$, the algorithm explores.

2.1: If there is no task $k \in E(t)$ such that $\hat{N}_k(t) \leq D_2(t)$, then assign the new task to $\mathcal{M}$ and update $E(t)$ to include it and denote it by k; if there is such a task, randomly select one of them, denoted by k, to $\mathcal{M}$. $\hat{N}_k(t) := \hat{N}_k(t) + 1$; obtain the label $y_k(\hat{N}_k(t))$;

2.2: Update $y_k^*(\hat{N}_k(t))$ (using the alternate indicator function notation $I(\cdot)$):

$$y_k^*(\hat{N}_k(t)) = I\left(\frac{\sum_{\hat{t}=1}^{\hat{N}_k(t)} y_k(\hat{t})}{\hat{N}_k(t)} > 0.5\right).$$

2.3: Update labelers' accuracy estimate $\forall i \in \mathcal{M}$:

$$\tilde{p}_i = \frac{\sum_{k \in E(t), k \text{ arrives at time } \hat{t}} I(L_i(\hat{t}) = y_k^*(\hat{N}_k(t)))}{|E(t)|}.$$

3: Else if $\mathcal{O}(t) = 0$, the algorithm exploits and computes:

$$S_t = \text{argmax}_m \tilde{U}(S^m) = \text{argmax}_{S \subseteq \mathcal{M}} \tilde{\pi}(S),$$

which is solved using the linear search property, but with the current estimates $\{\tilde{p}_i\}$ rather than the true quantities $\{p_i\}$, resulting in estimated utility $\tilde{U}()$ and $\tilde{\pi}()$. Assign the new task to those in S_t.

4: Set $t = t + 1$ and go to Step 2.

Regret Bounds

$$R(T) \leq \frac{U(S^*)}{\left(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha\right)^2 \cdot \varepsilon^2 \cdot (a_{\min} - 0.5)^2} \cdot \log^2(T)$$

$$+ \Delta_{\max}\Big(2 \sum_{\substack{m=1 \\ m\ odd}}^{M} m \cdot n(S^m) + M\Big) \cdot \Big(2\beta_2 + \frac{1}{\alpha \cdot \varepsilon}\beta_{2-z}\Big) ,$$

- The regret is in an order of $\log^2 T$
- The regret converge as $T \to \infty$

Regret Bounds

$$R(T) \leq \frac{U(S^*)}{\left(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha\right)^2 \cdot \varepsilon^2 \cdot (a_{\min} - 0.5)^2} \cdot \log^2(T)$$

$$+ \Delta_{\max}\left(2 \sum_{\substack{m=1 \\ m\ odd}}^{M} m \cdot n(S^m) + M\right) \cdot \left(2\beta_2 + \frac{1}{\alpha \cdot \varepsilon}\beta_{2-z}\right),$$

- Inversely proportional to a_{min}

$$a_{min} := P\left(\frac{\sum_{i=1}^{M} x_i}{M} > 1/2\right)$$

- Prob. that majority vote of all workers give a correct label

Regret Bounds

$$R(T) \leq \frac{U(S^*)}{\left(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha\right)^2 \cdot \varepsilon^2 \cdot (a_{\min} - 0.5)^2} \cdot \log^2(T)$$

$$+ \Delta_{\max}\left(2 \sum_{\substack{m=1 \\ m\ odd}}^{M} m \cdot n(S^m) + M\right) \cdot \left(2\beta_2 + \frac{1}{\alpha \cdot \varepsilon}\beta_{2-z}\right),$$

- Inversely proportional to a_{min}

$$x_i \sim Bin(p_i, 1)$$

$$a_{min} := P\left(\frac{\sum_{i=1}^{M} x_i}{M} > 1/2\right)$$

- Prob. that majority vote of all workers give a correct label

Regret Bounds

$$R(T) \leq \frac{U(S^*)}{\left(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha\right)^2 \cdot \varepsilon^2 \cdot (a_{\min} - 0.5)^2} \cdot \log^2(T)$$

$$+ \Delta_{\max}\left(2 \sum_{\substack{m=1 \\ m\ odd}}^{M} m \cdot n(S^m) + M\right) \cdot \left(2\beta_2 + \frac{1}{\alpha \cdot \varepsilon}\beta_{2-z}\right),$$

- Inversely proportional to a_{min}

$$x_i \sim Bin(p_i, 1)$$

quality

$$a_{min} := P\left(\frac{\sum_{i=1}^{M} x_i}{M} > 1/2\right)$$

- Prob. that majority vote of all workers give a correct label

Regret Bounds

$$R_{\mathfrak{C}}(T) \leq \frac{\sum_{i \notin S^*} c_i}{(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha)^2 \cdot \epsilon^2} \cdot \boxed{\log T}$$

$$+ \frac{\sum_{i \in \mathfrak{M}} c_i}{(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha)^2 \cdot \epsilon^2 \cdot (a_{min} - 0.5)^2} \cdot \boxed{\log^2(T)}$$

$$+ (M - |S^*|) \cdot (2\sum_{m=1}^{M} m \cdot n(S^m) + M) \cdot (2\beta_2 + \frac{1}{\alpha \cdot \epsilon}\beta_{2-z})$$

- The regret is in an order of $\log^2 T$
- The regret converge as $T \to \infty$

Regret Bounds

$$R_{\mathfrak{C}}(T) \leq \frac{\sum_{i \notin S^*} c_i}{(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha)^2 \cdot \epsilon^2} \cdot \log T$$
$$+ \frac{\sum_{i \in \mathfrak{M}} c_i}{(\frac{1}{\max_{m:m\ odd} m \cdot n(S^m)} - \alpha)^2 \cdot \epsilon^2 \cdot (a_{min} - 0.5)^2} \cdot \log^2(T)$$
$$+ (M - |S^*|) \cdot (2 \sum_{m=1}^{M} m \cdot n(S^m) + M) \cdot (2\beta_2 + \frac{1}{\alpha \cdot \epsilon} \beta_{2-z})$$

- Inversely proportional to a_{min}

Weighted Majority Vote

- Determine the mostly likely label of the task by:

$$\text{argmax}_{l \in \{0,1\}} \underbrace{P(L^*(t) = l | L_1(t), \ldots, L_M(t))}$$

Posterior probability of true label given labels from workers

Weighted Majority Vote

- Determine the mostly likely label of the task by:

$$\operatorname{argmax}_{l \in \{0,1\}} P(L^*(t) = l | L_1(t), \ldots, L_M(t))$$

- Prob. that the true label is 1 given the labels from workers:

$$\begin{aligned} P(L^*(t) = 1 | L_1(t), \ldots, L_M(t)) &= \frac{P(L_1(t), \ldots, L_M(t), L^*(t) = 1)}{P((L_1(t), \ldots, L_M(t))} \\ &= \frac{P(L_1(t), \ldots, L_M(t) | L^*(t) = 1) \cdot P(L^*(t) = 1)}{P((L_1(t), \ldots, L_M(t))} \\ &= \frac{P(L^*(t) = 1)}{P((L_1(t), \ldots, L_M(t))} \cdot \prod_{i:L_i(t)=1} p_i \cdot \prod_{i:L_i(t)=0} (1 - p_i) \end{aligned}$$

Weighted Majority Vote

- Prob. that the true label is 0 given the labels from workers:

$$P(L^*(t)=0|L_1(t),\ldots,L_M(t))$$
$$=\frac{P(L^*(t)=0)}{P((L_1(t),\ldots,L_M(t))}\cdot\prod_{i:L_i(t)=0}p_i\cdot\prod_{i:L_i(t)=1}(1-p_i)$$

Weighted Majority Vote

- Suppose the true label for task k is 1
- Assume equal prior $P(L^*(t) = 1) = P(L^*(t) = 0)$
- A true label is produced if

$$\prod_{i:L_i(t)=1} p_i \cdot \prod_{i:L_i(t)=0} (1 - p_i) > \prod_{i:L_i(t)=0} p_i \cdot \prod_{i:L_i(t)=1} (1 - p_i)$$

- Take log(.) on both sides

$$\underbrace{\sum_{i:L_i(t)=1} \log \frac{p_i}{1 - p_i}}_{\text{weighted votes that label is 1}} > \underbrace{\sum_{j:L_j(t)=0} \log \frac{p_j}{1 - p_j}}_{\text{weighted votes that label is 0}}$$

Weighted Majority Vote

- Weight of a set of workers S

$$W(S) = \sum_{i \in S} \log \frac{p_i}{1-p_i}, \forall S \subseteq \mathfrak{M}$$

- The estimated version using estimated worker qualities

$$\tilde{W}(S) = \sum_{i \in S} \log \frac{\tilde{p}_i}{1-\tilde{p}_i}, \forall S \subseteq \mathfrak{M}$$

- Use this weighted majority voting scheme to aggregate labels in the algorithm

Regret Bound

$$R(T) \le \frac{U(S^*)}{\left(\frac{1}{\max_m \max\{4C\cdot m, m\cdot n(S^m)\}} - \alpha\right)^2 \cdot \varepsilon^2 \cdot (a_{\min} - 0.5)^2} \log^2 T$$
$$+ \Delta_{\max}\left(2 \cdot \sum_{m=1}^{M} m \cdot n(S^m) + M + \frac{M^2}{2}\right) \cdot \left(2\beta_2 + \frac{1}{\alpha \cdot \varepsilon} \beta_{2-z}\right).$$

- Also in an order of $\log^2 T$
- Has a smaller constant than the regret of majority vote
- Converge to a lower regret

Simulation Study

- Setup:
  - 5 workers
  - Generate p_i uniformly at random between $[0.6, 1]$
- Baseline:
  - Majority vote over all workers

Simulation Study

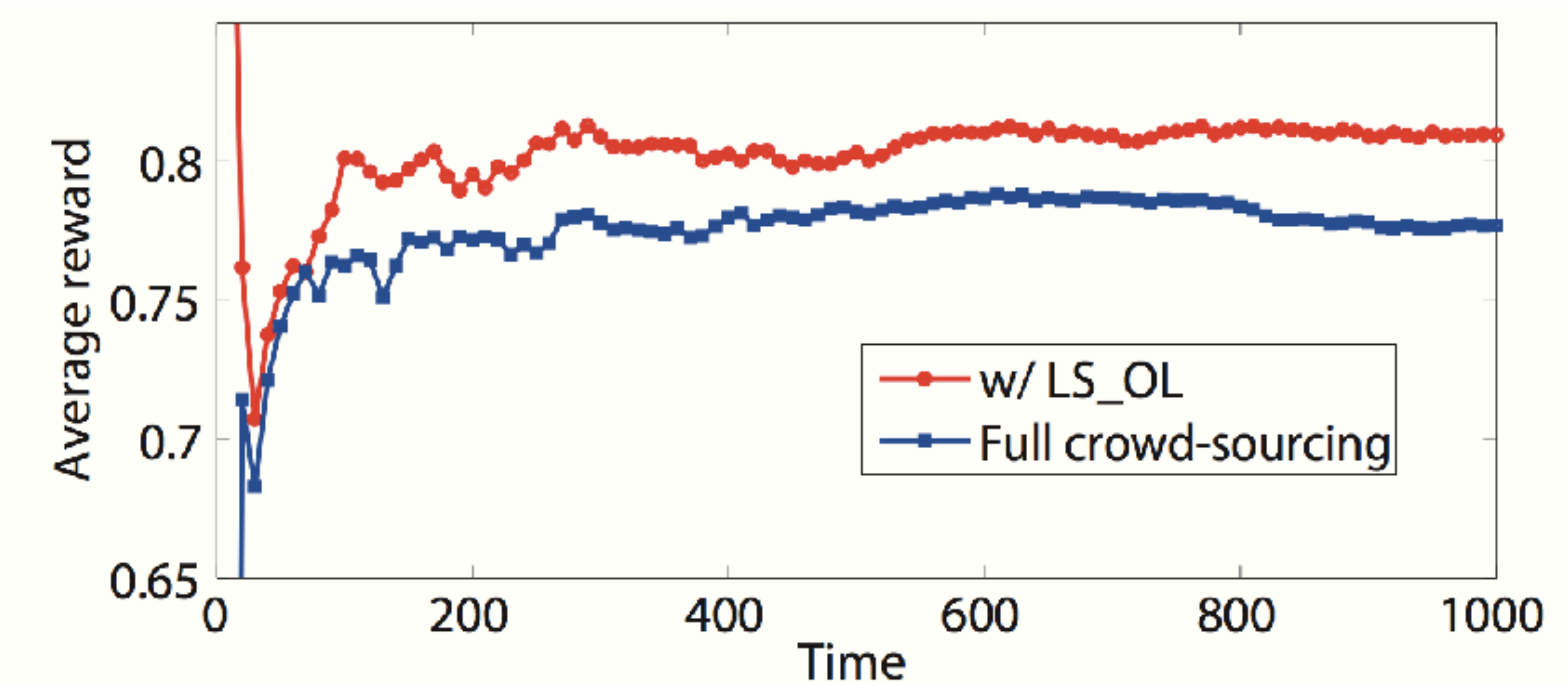


Figure 3: Performance comparison: online labeler selection v.s. full crowd-sourcing (majority vote)

Simulation Study

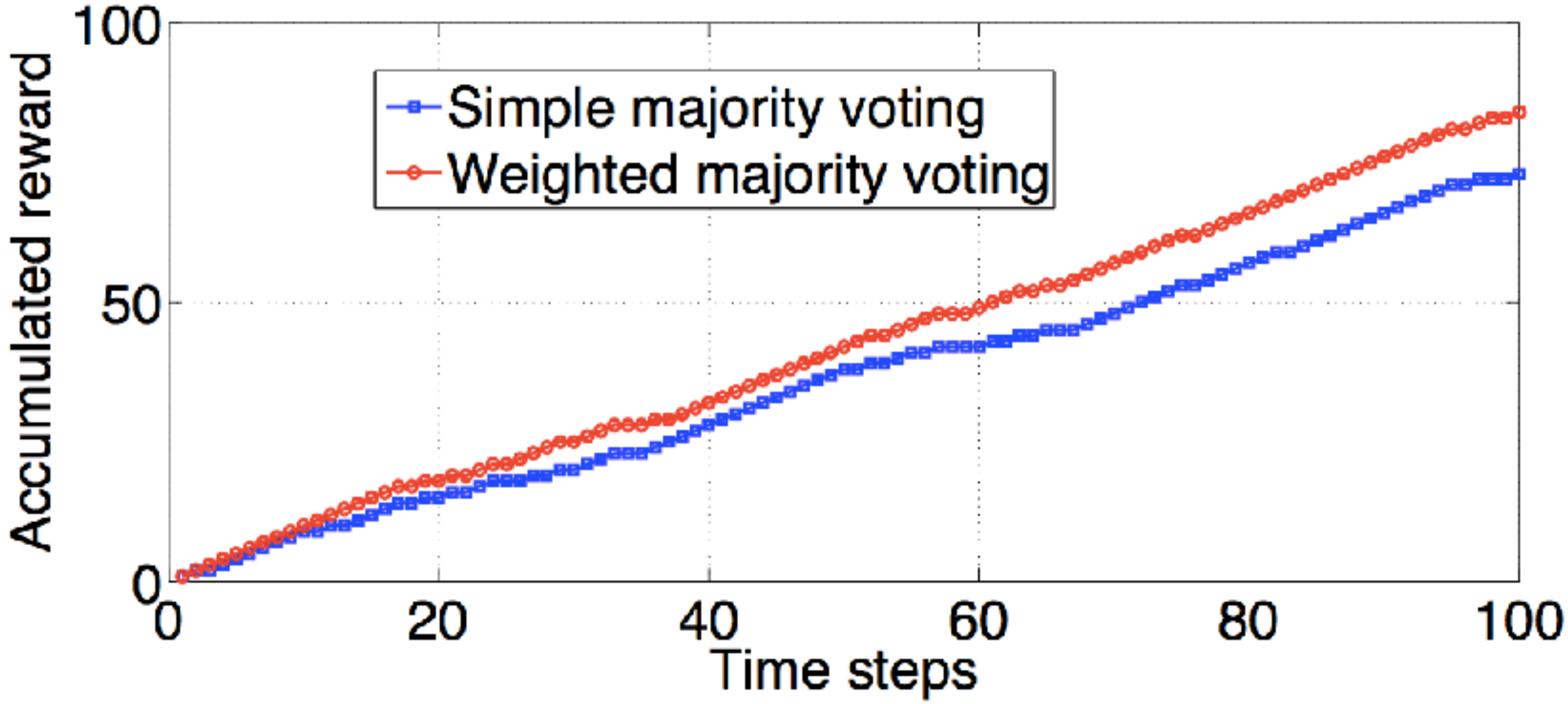


Figure 5: Comparing weighted and simple majority voting within LS_OL.

Study on Real Data

- Dataset:
  - Collected from Amazon Mechanical Turk
  - 1,000 images each labeled by 5 workers
- Baseline:
  - Majority vote over all workers

Study on Real Data

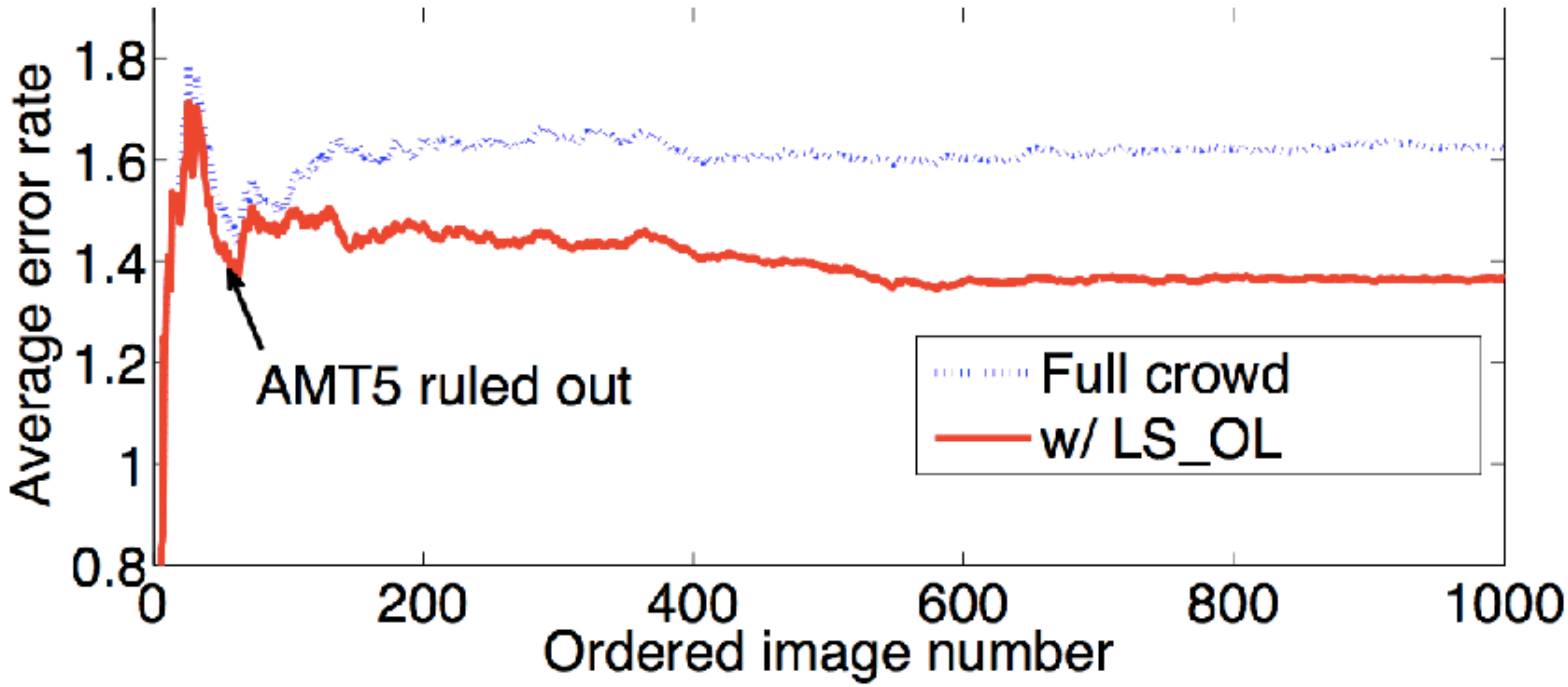


Figure 7: Average error comparison: online labeler selection v.s. full crowd-sourcing.

Conclusion

- Purpose two online learning algorithms to:
  - Learn worker qualities
  - Select the best set of workers
  - Aggregate labels from workers
- Conduct a theoretical analysis of the proposed algorithm