Lifelong Learning for Acquiring the Wisdom of the Crowd

Ece Kamar, Ashish Kapoor, Eric Horvitz
Microsoft Research
Redmond, WA USA
{eckamar, akapoor, horvitz}@microsoft.com

Abstract

Predictive models play a key role for inference and decision making in crowdsourcing. We present methods that can be used to guide the collection of data for enhancing the competency of such predictive models while using the models to provide a base crowdsourcing service. We focus on the challenge of ideally balancing the goals of collecting data over time for learning and for improving task performance with the cost of workers' contributions over the lifetime of the operation of a system. We introduce the use of distributions over a set of predictive models to represent uncertainty about the dynamics of the world. We employ a novel Monte Carlo algorithm to reason simultaneously about uncertainty about the world dynamics and the progression of task solution as workers are hired over time to optimize hiring decisions. We evaluate the methodology with experiments on a challenging citizen-science problem, demonstrating how it balances exploration and exploitation over the lifetime of a crowdsourcing system.

1 Introduction

In recent years, crowdsourcing has emerged as a valuable approach to harness human abilities from a population of workers to solve tasks that computers cannot easily perform alone. Crowdsourcing has been used to solve such tasks as image labeling, product categorization, and handwriting recognition. In most cases, system administrators take on the burden of managing crowdsourcing tasks and making hiring decisions. This time consuming job is a barrier for wider uses of crowdsourcing. We investigate principles and algorithms for the adaptive control of crowdsourcing tasks. We describe a methodology, named CrowdExplorer, that optimizes hiring decisions by adaptively learning about tasks as it observes worker contributions. We demonstrate the operation and value of the methods with a workload drawn from efforts on a large-scale citizen science challenge.

We focus particularly on inference and decision making for the crowdsourcing of *consensus tasks*. The goal of consensus tasks is to identify a correct answer by collecting assessments from different human workers. Examples of consensus tasks include *games with a purpose* (e.g., image labeling in the ESP game) [Von Ahn and Dabbish, 2008], paid crowdsourcing systems (e.g., product categorization in Mechanical Turk) [Ipeirotis, 2010], and citizen science projects (e.g., efforts to classify birds or celestial objects). Consensus tasks can be subtasks of larger tasks. For example, a system for providing real-time traffic flow and predictions may acquire reports on traffic conditions by contacting drivers within targeted regions [Krause *et al.*, 2008].

We shall describe how successful adaptive control of consensus tasks hinges on managing the tradeoff between the exploration of consensus tasks by hiring workers and exploitation by optimizing the best action to take via the use of decision-theoretic models. We formalize decision-making in consensus tasks with a Markov decision process (MDP) with partial observability, and highlight the challenges in the adaptive control of consensus tasks when accurate models of the world are not known and need to be learned over time. We describe a methodology we refer to as CrowdExplorer for the adaptive control of consensus tasks. CrowdExplorer uses a set of linear predictive models to learn continuously about the dynamics of consensus tasks. The procedure quantifies its uncertainty over the world dynamics with probability distributions over a set of models. CrowdExplorer uses a novel Monte Carlo planning algorithm to optimize decisions about hiring workers by efficiently and simultaneously reasoning about the uncertainty over models and about the way a task may progress. We evaluate the methodology on a dataset collected from a real-world crowdsourcing effort called Galaxy Zoo. The evaluations demonstrate that the methodology can outperform existing approaches for addressing the exploration-exploitation tradeoff and that its use can significantly improve the value of crowdsourcing.

2 Consensus Tasks

A task is a *consensus task* if it centers on identifying a correct answer that is unknown to the task owner but can be correctly identified by aggregating multiple workers' predictions. Formally, a consensus task t is characterized as follows: Let A be the set of possible answers for t. There exists a mapping $t \rightarrow a^* \in A$ that assigns each task to a correct answer. $L \subseteq A$ is a subset of answers that workers are aware of, $o \in L$ is the prediction (vote) of a worker about the correct answer of the task. Each task is associated with a finite horizon (budget)

h, which determines the maximum number of workers that can be hired for a task. The task owner has a positive utility $u \in \mathbb{R}_{>0}$ for correctly identifying the correct answer of the task but hiring each worker is associated with a cost $c \in \mathbb{R}_{>0}$. Once the budget is consumed, a consensus rule f maps the sequence of worker votes $\{o_1, ..., o_h\}$ to the correct answer $a^* \in A$. A widely used example of consensus rule is the majority rule, which determines the correct answer as the answer that is predicted the most by the workers.

A crowdsourcing system usually hosts a series of consensus tasks of the same task class (e.g., image labeling and product categorization) which may arrive at arbitrary times. In a traditional workflow, the system uses the whole budget to identify the correct answer using the consensus rule and delivers the correct answer to the task owner. Once a task is completed, the system moves on to solving another task. With this workflow, the system is able to achieve perfect accuracy but hires the maximum number of workers for each task. We focus in this paper on an adaptive control methodology that can learn about consensus tasks over time and optimize the hiring of workers to maximize the overall efficiency.

2.1 Galaxy Zoo as Consensus Tasks

We motivate and evaluate our adaptive control mechanism with Galaxy Zoo. Galaxy Zoo is one of the largest citizen science efforts to date and serves as a well-known and canonical example of a consensus task. The Galaxy Zoo system was designed to engage members of the public to provide help with identifying the correct classification of millions of galaxies [gal, 2007].

Each Galaxy Zoo task is an image of a celestial object. Workers classify images into 6 possible classes (e.g., elliptical galaxy, clockwise spiral galaxy). The budget of a Galaxy Zoo task is determined by the number of workers assigned to the task by the original Galaxy Zoo system and this number varies randomly from 30 to 93 for different tasks. According to the consensus rule defined by the task owners, the correct answer of a Galaxy Zoo task is determined as the answer that is agreed upon by at least 80% of the workers once the whole budget is consumed. If such a consensus is not reached, the correct answer is called *undecidable*. Studies on the data collected from the Galaxy Zoo system showed that the answers identified through the consensus rule have high agreement with expert opinion [Lintott and others, 2008].

2.2 Modeling Consensus Tasks

We model decision making in consensus tasks as a finite-horizon MDP with partially observable rewards, represented with a tuple $< h, S, \mathbb{A}, T, R >$. h, the horizon of the task, is determined by the maximum number of workers to be hired for a task. $s_t \in S$, a state of a consensus task at time t, is composed of $\{o_1, ..., o_t\}$, the sequence of worker votes. [1] $\mathbb{A}$, the set of actions for a consensus task include H, hire a worker, and $\neg H$, terminate and deliver the most likely answer to the task owner. Making the right tradeoff between these two actions is crucial for the success of the system because when $\neg H$ is selected, the system terminates and there is no chance to reverse this decision. $T(s_t, \alpha, s_{t+1})$ is the likelihood of transitioning from state s_t to s_{t+1} after taking action α. The system transitions to a terminal state and then moves on to a new task, if the selected action is $\neg H$. If the system decides to hire a worker, the system may transition to any s_{t+1} such that $s_{t+1} = s_t \cup \{o_{t+1}\}$, where $o_{t+1} \in L$. This transition probability corresponds to the likelihood of a randomly selected worker voting o_{t+1}. Since worker votes are correlated, this likelihood is conditioned on the sequence of votes collected so far on a task.

The reward function $R(s_t, \alpha)$ represents the reward obtained by executing action α in state s_t. The reward for hiring a worker is determined by c, the cost for hiring a worker. The reward for terminating a task depends on the likelihood of correctly identifying an answer label for the task. The system can observe the sequence of votes collected for a task, and consequently the current state, but it cannot observe the correct answer. Therefore, it keeps a belief about the correct answer at each state to be able to estimate the reward for terminating. The correct answer is computed using the consensus rule at the horizon. At states earlier than the horizon, the correct answer is predicted by applying the transition model recursively until reaching the horizon. $b(s_t, a)$, the probability of correct answer being a at state s_t, is computed as given below, where f is the consensus rule:

$$b(s_t, a) = \begin{cases} [f(o_1, ..., o_t) = a] & \text{if } t = h \\ \Sigma_{s_{t+1} \in S} T(s_t, H, s_{t+1}) b(s_{t+1}, a) & \text{otherwise} \end{cases}$$

When the system terminates, it delivers the answer that is the most likely to be correct according to its belief. Let $\hat{a} = argmax_{a \in A} b(s_t, a)$, the reward for terminating at state s_t is defined as $R(s_t, \neg H) = b(s_t, \hat{a}) \times u$, where u is the utility for predicting the answer correctly.

When the parameters of the MDP are known, the Bellman equation can be used to calculate for any state s_t the value for terminating ($V^{\neg H}$) and the value for hiring (V^H):

$$V^{\neg H}(s_t) = R(s_t, \neg H)$$
$$V^H(s_t) = R(s_t, H) + \sum_{s_{t+1}} T(s_t, H, s_{t+1})\, V^{\pi^*}(s_{t+1})$$

where $V^{\pi^*}(s_t) = max(V^H(s_t), V^{\neg H}(s_t))$. The decision at any state of a consensus task is guided by Value of Information (VOI) analysis. VOI for state s_t is the difference between $V^H(s_t)$ and $V^{\neg H}(s_t)$. If VOI is computed to be positive, it is beneficial to hire an additional worker.

In practice, the number of states of a consensus task grows exponentially with the horizon, which makes exact solution approaches intractable. Previous work showed that consensus tasks can be solved efficiently using Monte Carlo planning, when accurate models of the dynamics of the world are known [Kamar and Horvitz, 2013].

Here, we investigate approaches for the adaptive control of consensus tasks in settings where accurate models of the world do not exist. Adaptive control of consensus tasks differs from other problems with inherent exploration-exploitation tradeoffs. In solving consensus tasks, a system

[1] If the system has information about the properties of the task or about characteristics of workers hired for the task, the state representation can be expanded accordingly.

needs to make decisions without receiving continuous reinforcement about its performance. Until the system reaches to the horizon, it does not receive an explicit signal about its performance. In contrast to traditional problems, in which any action helps to explore the world, the exploration of a consensus task permanently terminates once $\neg H$ action is taken.

In consensus tasks, the domains of answers and worker predictions are finite and known. The values for the horizon, utilities for correct identification of answers and for worker costs are quantified by task owners. However, both the priors on the correct answers of consensus tasks and the transition models are unknown, and they need to be learned over time. Therefore, to make hiring decisions appropriately, a successful adaptive control system needs to reason about its uncertainty about its specific models of the world as well as its uncertainty over the way a task may progress.

3 Adaptive Control Methodology

We now focus on the *CrowdExplorer* methodology. The building blocks of CrowdExplorer are an online learning module for learning a set of probabilistic models representing the dynamics of the world (i.e. state transitions), and a decision-making module that optimizes hiring decisions by simultaneously reasoning about its uncertainty about its models and the way a task may stochastically progress.

One of the key challenges is that the number of state transitions that define the dynamics of consensus tasks grows exponentially with the horizon. To address this challenge, we use a property of consensus tasks: the next state of the system is completely determined by the vote of a next worker. Using this property, we capture the transition probabilities with a set of models that predict the vote of a next worker based on the current state of the task. This implicit representation of the world dynamics significantly reduces the number of variables needed to represent consensus tasks.

Formally, we model state transitions with a set of linear models $M = \{M_1, ..., M_{|L|}\}$, where M_i predicts the likelihood of a next worker predicting the answer as $a_i \in L$. Each model takes as input a set of features describing the current state, including the ratio of number of collected votes to the horizon, and for each vote class, the ratio of number of votes collected for that class to the total number of votes collected. Let $\mathbf{x}_t$ denote k dimensional feature representation of state s_t and each model M_i is defined by k-dimensional vector of weights $\mathbf{w}_i$, then we estimate transition probabilities as below, where $s_{t+1} = s_t \cup \{o_{t+1} = a_i\}$.

$$T(s_t, H, s_{t+1}) = \frac{e^{\mathbf{w}_i^T \mathbf{x}_t}}{\Sigma_j e^{\mathbf{w}_j^T \mathbf{x}_t}}$$

The linear models are constantly updated using an online learning algorithm. Initially, the models are uninformative per the lack of training instances. As workers provide votes, the system has access to more data and consequently the models starts to provide useful transition probabilities. As these models are latent, we represent the parameters $\mathbf{w}_i$ as random variables. The online learning is implemented as a Bayesian inference procedure using Expectation Propagation [Minka, 2001]. Specifically, the inference procedure provides a Gaussian posterior distribution over the model parameters $\mathbf{w}_i$. One of the key benefits of the Bayesian treatment is that the variance of this posterior distribution captures a measure of confidence in determining the model. Intuitively, when little data observed, the inference procedure will typically return a covariance matrix with large diagonal entries, corresponding to the high degree of difficulty in determining the model from a small amount of data. This uncertainty quickly diminishes as the system sees more training instances. Reasoning about such uncertainty is crucial as it enables management of the tradeoff between exploration, learning better models by hiring more workers, and exploitation, selecting the best action based on the current models of the world.

begin
  initialize $Pr_M = \{Pr_{M_1}, ..., Pr_{M_{|L|}}\}$
  foreach *task* i **do**
    $s_t^i \leftarrow \{\}$
    repeat
      $VOI \leftarrow CalculateVOI(s_t^i, Pr_M)$
      if $VOI > 0$ **then**
        $o_{t+1} \leftarrow GetNextWorkerVote$
        $AddLabel(Pr_M, o_{t+1})$
        $s_{t+1}^i \leftarrow s_t^i \cup \{o_{t+1}\}$
        $s_t^i \leftarrow s_{t+1}^i$
      end
    until $VOI \leq 0$ *or* $t = h$
    output $s_t^i.\hat{a}$
  end
end

CalculateVOI(s_t:state, Pr_M:model distribution)
begin
  repeat
    $\{\widetilde{M_1}, ..., \widetilde{M_{|L|}}\} \leftarrow SampleModels(Pr_M)$
    $SampleExecutionPath(s_t, \{\widetilde{M_1}, ..., \widetilde{M_{|L|}}\}, h)$
  until *Timeout*
  return $VOI \leftarrow s_t.V^H - s_t.V^{\neg H}$
end

Algorithm 1: CrowdExplorer methodology

The other key component of CrowdExplorer is the decision-making module. This module is responsible for estimating the value of hiring an additional worker by taking into account the uncertainty over the models and over the stochastic transitions of the world. The foundation of the decision-making module is a Monte Carlo planning algorithm, MC-VOI, which is designed for making decisions for consensus tasks when accurate models of the world are known. MC-VOI explores the exponential space of consensus tasks efficiently by sampling future state, action transitions. Previous work has shown that MC-VOI can outperform other Monte Carlo planning algorithms for solving consensus tasks. We expand the MC-VOI algorithm in our decision-making module to reason about the model uncertainty that is inherent to adaptive control in addition to reasoning about the stochastic transitions in the world.

Details of the CrowdExplorer methodology are given in Algorithm 1. For any state s_t^i of a consensus task i, the

methodology uses sampling to estimate values of states for taking different actions as an expectation over possible models and stochastic transitions. At each iteration, the methodology first samples a set of models $(\widetilde{M_1}, ..., \widetilde{M_{|L|}})$ from the model distribution Pr_M. These sampled models are then used to sample future state transitions from s_t^i by continuously taking action H until reaching the horizon. The resulting state transitions form an execution path. Each execution path represents one particular way a consensus task may progress if the system hires workers until reaching the horizon. The aggregation of execution paths forms a partial search tree over possible states. The tree represents both the uncertainty over the models and over future transitions.

For each state s_t on the partial search tree, the methodology uses recursive search on the tree to estimate values for hiring a worker $(s_t.V^H)$ and for terminating $(s_t.V^{\neg H})$, and to predict the most likely answer for that state $(s_t.\hat{a})$ (See Algorithm 2). A worker is hired if VOI for the initial state is estimated to be positive. Once the vote of the next worker arrives, the vote is used to update the predictive models and update the state of the task. This computation is repeated for future states until the budget is consumed or VOI is estimated to be non-positive. The methodology terminates the task by delivering the predicted answer $(\hat{a})$ and moves on to the next task.

The variance of the predictive models estimated dynamically by the online learning algorithm guides the decision-making algorithm in controlling the exploitation-exploration tradeoff. When the variance is high, each sampled model provides a different belief about the way future workers will vote. Execution paths reflecting these diverse beliefs lead to high uncertainty about the consensus answer that will be received at the horizon. Consequently, this leads to more exploration by hiring workers. When the variance is low, sampled models converge to a single model. In this case, the hiring decisions are guided by exploiting the model and selecting the action with the highest expected utility.

This behavior is illustrated in Figure 1 for a toy example, in which $o_i \in \{0, 1\}$, $h = 3$ and majority rule is the consensus rule. The figure displays the partial search trees generated for initial state $s_1 = \{o_1 = 1\}$ when there is (a) high uncertainty over the models - model weights w_i having large variance—and (b) low uncertainty over the models—model weights w_i having small variance. Branches between nodes represent possible votes to be collected from workers. Each leaf is labeled with a^*, the correct answer as identified at the horizon, and N, the number of times the leaf is sampled. In (a), high uncertainty over the models leads to high uncertainty over the correct answer. Consequently, high value is associated with hiring more workers to resolve the uncertainty over the correct answer. VOI is estimated to be high. In (b), there is no uncertainty over the correct answer according to the partial search tree generated by the algorithm. Therefore, hiring a worker does not help with predicting the correct answer. VOI is estimated to be not positive.

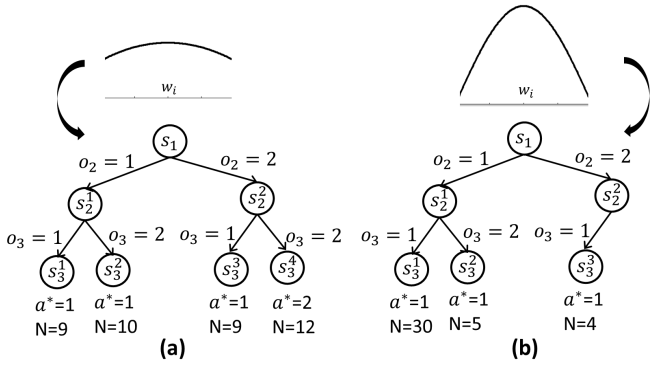

Figure 1: Search trees generated by CrowdExplorer when there is (a) high and (b) low uncertainty over models.

3.1 Sampling Execution Paths

The algorithm for sampling an execution path (p) for a sampled model $(\tilde{M})$ is presented in Algorithm 2. The detailed explanation of the algorithm is presented in [Kamar and Horvitz, 2013]. The algorithm generates execution paths by recursively sampling future votes from the predictive models until reaching the horizon. At the horizon, it uses the consensus rule to determine the correct answer corresponding to the path (a_p^*). For each path, the algorithm uses a_p^* to evaluate the utilities of each state on the path for taking actions H and $\neg H$ by taking into account c, the cost of a worker.

SampleExecutionPath(s_t:state, $\tilde{M}$:set of models, h:horizon)
begin
  if $t = h$ **then**
    $a_p^* \leftarrow ConsensusRule(s_t)$
  else
    $o_{t+1} \leftarrow SampleNextVote(s_t, \tilde{M})$
    $s_{t+1} \leftarrow s_t \cup \{o_{t+1}\}$
    $a_p^* \leftarrow SampleExecutionPath(s_{t+1}, \tilde{M}, h)$
  end
  $s_t.N[a_p^*] \leftarrow s_t.N[a_p^*] + 1$
  $s_t.N \leftarrow s_t.N + 1$
  $s_t.V^{\neg H} \leftarrow (\frac{\max_{a \in A} s_t.N[a]}{s_t.N} \times u) - (t \times c)$
  if $t < h$ **then**
    $s_t.V^H \leftarrow \frac{\sum_{s'_{t+1} \in \Phi(s_t)} (s'_{t+1}.V \times s'_{t+1}.N)}{s_t.N}$
  end
  $s_t.V \leftarrow \max(s_t.V^{\neg H}, s_t.V^H)$
  $s_t.\hat{a} \leftarrow argmax_{a \in A} s_t.N[a]$
  return a_p^*
end

Algorithm 2: Algorithm for Sampling Execution Paths

For each state s_t visited on a path, the algorithm keeps the following values: $s_t.N$ as the number of times s_t is sampled, $s_t.N[a]$ as the number of times a path visited s_t reached answer a, $s_t.N[a]/s_t.N$ as the likelihood at s_t for the correct answer being a, $s_t.\hat{a}$ as the predicted answer at s_t. $s_t.V^{\neg H}$, the value for terminating, is estimated based on the likelihood of predicting the answer correctly at that state. $\Phi(s_t)$ is the set of states reachable from s_t after taking action H. $s_t.V^H$, the value for hiring more workers, is calculated as the weighted average of the values of future states accessible from s_t.

4 Experiment Setup

We evaluated the ability of CrowdExplorer methodology to adaptively control consensus tasks on a dataset collected by Galaxy Zoo. The dataset includes 44350 votes collected for 1000 randomly selected Galaxy Zoo tasks, where each task is the labeling of a single image. The budget for each task is determined by the number of worker votes available for each task in the dataset. The budget of each task is known and varies between 30 and 93. The sequence of votes for each task is ordered randomly. The correct answer of each task is determined by the consensus rule defined by the experts of Galaxy Zoo; the answer agreed on by the 80% of workers at the horizon of a task is the correct answer. The system is rewarded $1 for correctly predicting the correct answer of a task (including predicting undecidables), the cost of hiring a worker is varied between 1 cent and 0.25 cents to observe the behavior of CrowdExplorer in trading off the cost of worker with expected benefit in different settings. The overall utility of the system over a certain number of tasks is the difference between the number of tasks answered correctly and the overall cost of workers hired.

We compare the performance of CrowdExplorer methodology with a *deterministic* baseline. This baseline continues hiring workers until there is no chance to change the system's opinion about the correct answer with the remaining number of workers to be hired.

We implemented a number of adaptive control methods to evaluate the performance of CrowdExplorer. These methods use well-known exploration policies to address the exploration-exploitation tradeoff in consensus tasks. They have access to the linear models described in Section 3. The models are dynamically updated with the votes collected from workers using the online learning algorithm. When it is best to exploit, these methods use the MC-VOI algorithm for selecting the best action to take. In selecting the best action, they do not reason about the uncertainty over the set of models, but use the single most likely set of models to represent the world dynamics.

The *constant exploration* method always explores for the first c_c tasks and then exploits for the remaining tasks. In the *epsilon-greedy* method, the probability of exploration at each state is calculated as $min(1, \frac{c_\epsilon}{n})$, where n is the number of tasks the system has seen so far. Traditional *Boltzman exploration* equation is proposed for settings in which all possible actions lead to exploring. We adapt the Boltzman method to consensus tasks as follows: If $V^H > V^{\neg H}$, the methodology hires a worker. If $V^H \leq V^{\neg H}$, the likelihood of hiring is:

$$Pr(H|s_t) = 2 \frac{e^{\frac{V^H(s_t)}{T}}}{e^{\frac{V^H(s_t)}{T}} + e^{\frac{V^{\neg H}(s_t)}{T}}}$$

where $T = \frac{c_b}{n}$. Since $Pr(H|s_t)$ is computed only when $V^H \leq V^{\neg H}$, this quantity is in [0,1]. It is close to 1 when T is large. It diminishes to 0 as more tasks are seen. Finally, the *optimistic* method estimates an optimistic value for hiring workers by adding a bound term $\frac{c_o}{n}$ to MC-VOI's estimation of V^H. It compares the optimistic estimate of V^H with $V^{\neg H}$ to select which action to take.

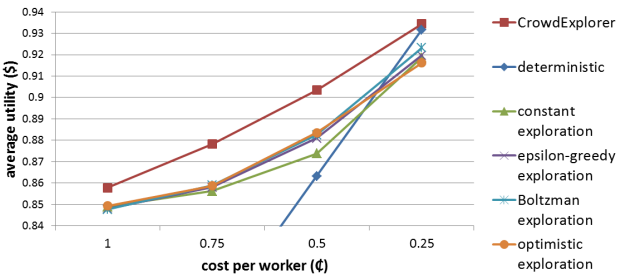

Figure 2: Average utilities of different methodologies for varying worker costs.

c_c, c_ϵ, c_b and c_o are positive constants given as input to the adaptive control methods for tuning their exploration behaviors. Similarly, the value that is used to initialize the co-variance matrices of the CrowdExplorer methodology can be adjusted to control the way the uncertainty over the models diminish with training data. In the experiments, we search over different values and report the results for the ones that provide largest overall utility. We found that the performance of CrowdExplorer was relatively robust to the initial setting of this co-variance matrix and maintained the performance advantage over the competing approaches for all reasonable values. In contrast, we found that the epsilon-greedy and optimistic exploration methods are very sensitive to their constant parameter values.

All of the examined adaptive control methodologies, including CrowdExplorer, explore the state space of consensus tasks by using the sampling approach of the MC-VOI algorithm to make decisions. The procedure can be stopped anytime. To control across all adaptive control methodologies, MC-VOI is provided the same computational resources for exploring the space of consensus tasks at every run.

5 Results

Figure 2 reports the average utilities of different adaptive control methodologies and the deterministic baseline over 1000 tasks as a function of the cost of a worker. As shown in the figure, CrowdExplorer delivers utilities higher than all other adaptive control methodologies for all worker costs. The value for adaptively controlling consensus tasks, instead of following a deterministic strategy, grows with the cost of a worker. When the cost is low (0.25 cents), the utility of CrowdExplorer is comparable to the utility of the deterministic baseline, and it is significantly better than the utilities of other adaptive control methodologies. For higher worker costs, all adaptive control methodologies perform better than the baseline.

CrowdExplorer dynamically adjusts its behavior to changing costs. When the cost is high (1 cent), it has 92% accuracy by hiring 13% of available workers. For high-medium cost (0.75 cents), it reaches to 95% accuracy with 20% of workers. Its accuracy is 97% with 30% workers when the cost is medium-low (0.5 cents). When the cost is low (0.25 cents), it reaches near perfect accuracy with 56% workers.

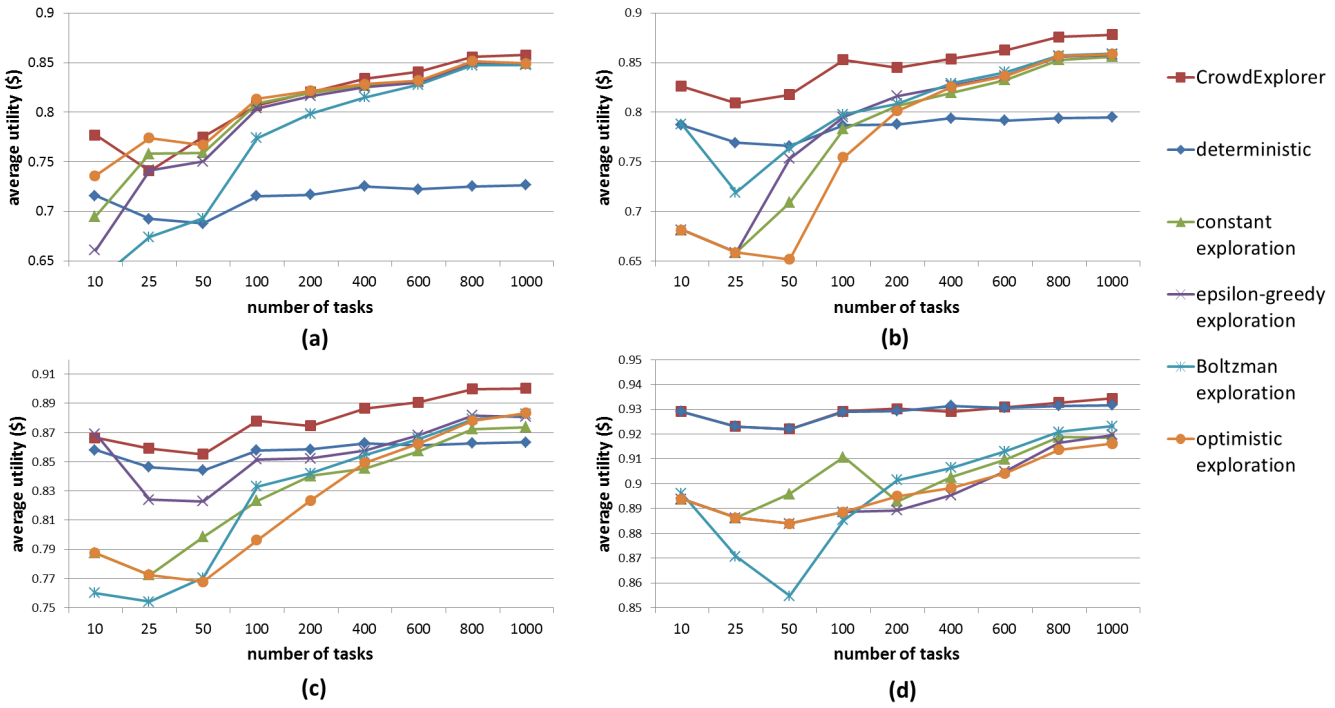

Figure 3: Average utilities of different methodologies when cost is (a) 1 cent, (b) 0.75 cents, (c) 0.5 cents and (d) 0.25 cents.

Figure 3 reports the average utilities of the methodologies with respect to the number of tasks solved for different worker costs. The performance of CrowdExplorer is better or comparable to other methodologies during the execution of the system for all worker costs. Other adaptive control methodologies often perform poorly until learning good models of the world. Their performances improve with the completion of increasing numbers of tasks. On the contrary, the performance of CrowdExplorer is consistently good (i.e., better or comparable to the deterministic baseline) throughout the execution of the system. In contrast to the other methodologies, CrowdExplorer can customize its exploration decisions to the specifics of a task and to its knowledge about the task. Based on its models at any time, CrowdExplorer can assess for which tasks it can make decisions about hiring workers confidently, and which tasks it needs to learn more about.

6 Related Work

Decision-theoretic guidance of crowdsourcing has recently come into focus as a rich research area. Dai et. al. focused on optimizing iterative workflows by separating the process into offline learning and optimization phases [Dai *et al.*, 2011]. Lin et. al. developed an agent for dynamically choosing between different iterative workflows [Lin *et al.*, 2012]. In contrast to the optimization of iterative workflows, which can be addressed with limited lookahead approaches, the optimization of consensus tasks requires reasoning about long sequences of evidence. Researchers proposed greedy and heuristic approaches for solving consensus tasks [Sheng *et al.*, 2008]. Kamar et. al. formalized consensus tasks as a sequential decision-making problem and introduced Monte Carlo planning approaches for addressing the combinatorial challenge [Kamar *et al.*, 2012; Kamar and Horvitz, 2013]. These approaches assume having perfect models of consensus tasks and did not address the challenge of simultaneously learning and optimizing for consensus tasks.

Adaptive control of consensus tasks relates to multiple lines of research for addressing the exploitation-exploration tradeoff. This tradeoff is studied under various assumptions in multi-armed bandit problems [Auer *et al.*, 2002], optimal stopping problems [Peskir and Shiryaev, 2006], active learning [Kapoor and E., 2007] and reinforcement learning research [Sutton and Barto, 1998]. Many of these assumptions are not realized on consensus tasks due to its special structure. For example, a consensus task does not receive continuous reinforcement about its performance, which prevents updating values of states based on observed rewards [Rummery and Niranjan, 1994; Silver *et al.*, 2008]. In consensus tasks, exploration is only possible by hiring more workers and exploration policies that select actions randomly may result in premature termination of tasks.

CrowdExplorer is a Monte Carlo approach for continuously learning and optimizing decisions when stochastic models of the world are unknown. This approach differs from existing Monte Carlo planning algorithms, which use sampling to explore and learn values for state, action pairs of large MDPs when generative models of the world are known [Kocsis and Szepesvári, 2006; Silver and Veness, 2010]. The statistical modeling techniques used by some Monte Carlo plan-

ning algorithms (e.g., UCT) are designed for action selection rather than for learning world models [Kocsis and Szepesvári, 2006]. The MC-VOI algorithm exploits the structure of consensus tasks to eliminate the challenge of action selection in Monte Carlo planning [Kamar and Horvitz, 2013]. Due to its superior performance in solving consensus tasks, it serves as a foundation for CrowdExplorer.

The methodology proposed has several other novel aspects: first our approach uses predictive models as implicit representations of the world. Related work to this aspect includes research in reinforcement learning where abstract representation of Q values for state-action pairs [Sutton, 1996; Džeroski *et al.*, 1998] are used. Second, our method efficiently models uncertainty over the predictive models. There has been some related research where uncertainty over the model of the world is represented as distributions over Q values or individual transition probabilities [Dearden *et al.*, 1999; Poupart and Vlassis, 2008; Jaulmes *et al.*, 2005]. The number of distributions to be represented by these approaches for consensus tasks grows exponentially with the horizon. Finally, our method incorporates a new Monte Carlo approach that can simultaneously reason about the uncertainty over models and over stochastic transitions. Existing approaches that separate sampling of models from decision-making [Dearden *et al.*, 1999; Jaulmes *et al.*, 2005] are hard to apply in our scenario as they get they quickly become intractable.

7 Conclusion and Future Work

We reviewed our efforts on the adaptive control of crowdsourcing tasks. We presented a methodology that can dynamically learn about the dynamics of crowdsourcing tasks and that can make effective decisions about when to hire more workers. We evaluated our methodology on a real-world dataset and demonstrated that the methodology can achieve significant savings in worker resources by adaptively learning about consensus tasks and optimizing hiring decisions.

We are exploring extensions of the methodology that can learn more sophisticated models of the world dynamics, including characteristics of individual workers and tasks. We are investigating extensions for controlling for the optimal timing and routing of tasks to workers and studying the applicability of the methodology for crowdsourcing tasks other than consensus tasks. We are also investigating extensions of CrowdExplorer to take into account the expected utility of learning on future tasks in making hiring decisions. Finally, we seek to study opportunities for simultaneously learning and optimizing decisions in association with other Monte Carlo planning algorithms. We believe a Monte Carlo approach that can simultaneously reason about model and transition uncertainties will be valuable for adaptive control in numerous real-world domains.

8 Acknowledgments

We thank Chris Lintott for sharing Galaxy Zoo data, Paul Koch for assistance with accessing the data, and Dan Bohus and Rich Caruana for discussions and feedback.

References

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

[Dai *et al.*, 2011] P. Dai, D.S. Weld, et al. Artificial intelligence for artificial artificial intelligence. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[Dearden *et al.*, 1999] R. Dearden, N. Friedman, and D. Andre. Model based Bayesian exploration. In *Proceedings of the fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 150–159. Morgan Kaufmann Publishers Inc., 1999.

[Džeroski *et al.*, 1998] S. Džeroski, L. De Raedt, and H. Blockeel. Relational reinforcement learning. *Inductive Logic Programming*, pages 11–22, 1998.

[gal, 2007] Galaxy Zoo, 2007.

[Ipeirotis, 2010] P.G. Ipeirotis. Analyzing the Amazon Mechanical Turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.

[Jaulmes *et al.*, 2005] R. Jaulmes, J. Pineau, and D. Precup. Active learning in partially observable markov decision processes. *Machine Learning: ECML 2005*, pages 601–608, 2005.

[Kamar and Horvitz, 2013] E. Kamar and E. Horvitz. Light at the end of the tunnel: A Monte-Carlo approach to computing value of information. In *To Appear in Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[Kamar *et al.*, 2012] E. Kamar, S. Hacker, and E. Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 467–474. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[Kapoor and E., 2007] A. Kapoor and Horvitz E. On discarding, caching, and recalling samples in active learning. In *Uncertainty in Artificial Intelligence*, 2007.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer, 2006.

[Krause *et al.*, 2008] A. Krause, E. Horvitz, A. Kansal, and F. Zhao. Toward community sensing. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 481–492. IEEE Computer Society, 2008.

[Lin *et al.*, 2012] C.H. Lin, M. Mausam, and D.S. Weld. Dynamically switching between synergistic workflows for crowdsourcing. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[Lintott and others, 2008] C.J. Lintott et al. Galaxy Zoo: Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008.

[Minka, 2001] T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence*, 2001.

[Peskir and Shiryaev, 2006] G. Peskir and A. Shiryaev. *Optimal stopping and free-boundary problems*, volume 10. Birkhäuser Basel, 2006.

[Poupart and Vlassis, 2008] P. Poupart and N. Vlassis. Model-based Bayesian reinforcement learning in partially observable domains. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics, Fort Lauderdale, FL, USA (January 2008)*. Citeseer, 2008.

[Rummery and Niranjan, 1994] G.A. Rummery and M. Niranjan. *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering, 1994.

[Sheng *et al.*, 2008] V.S. Sheng, F. Provost, and P.G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[Silver and Veness, 2010] D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. *Advances in Neural Information Processing Systems*, 23:2164–2172, 2010.

[Silver *et al.*, 2008] D. Silver, R.S. Sutton, and M. Müller. Sample-based learning and search with permanent and transient memories. In *Proceedings of the 25th international conference on Machine learning*, pages 968–975. ACM, 2008.

[Sutton and Barto, 1998] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

[Sutton, 1996] R.S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, pages 1038–1044, 1996.

[Von Ahn and Dabbish, 2008] L. Von Ahn and L. Dabbish. Designing games with a purpose. *Communications of the ACM*, 2008.