Bolt: Instantaneous Crowdsourcing via Just-in-Time Training

Alan Lundgard[1], **Yiwei Yang**[1], **Maya L. Foster**[2], **Walter S. Lasecki**[1]
Computer Science and Engineering, University of Michigan – Ann Arbor[1]
Neuroscience, Johns Hopkins University[2]

ABSTRACT

Real-time crowdsourcing has made it possible to solve problems that are beyond the scope of artificial intelligence (AI) within a matter of seconds, rather than hours or days with traditional crowdsourcing techniques. While this has led to an increase in the potential application domains of crowdsourcing and human computation, problems that require machine-level speeds—on the order of milliseconds, not seconds—have remained out of reach because of the fundamental bounds of human perception and response time. In this paper, we demonstrate that it is possible to exceed these bounds by combining human and machine intelligence. We introduce the *look-ahead approach*, a hybrid intelligence workflow that enables *instantaneous crowdsourcing* systems (i.e., those that can return crowd responses within mere milliseconds). The look-ahead approach works by exploring possible future states that may be encountered within a short time horizon (e.g., a few seconds into the future) and prefetching crowd worker responses to these states. We validate the efficacy and explore the limitations of our approach on the *Bolt* system, which consists of an arcade-style game (Lightning Dodger) that we formally model as a Markov Decision Process (MDP). When the MDP reward function is unspecified—as in many real-world tasks—the look-ahead approach enables just-in-time (JIT) training of the agent's policy function. Through a series of crowd worker experiments, we demonstrate that the look-ahead approach can outperform the fastest individual worker by approximately two orders of magnitude. Our work opens new avenues for hybrid intelligence systems that are as smart as people, but also far faster than humanly possible.

ACM Classification Keywords
H.5.m Information Interfaces and Presentation (e.g., HCI): Miscellaneous

Author Keywords
Instantaneous crowdsourcing; real-time crowdsourcing; continuous crowdsourcing; interactive crowdsourcing; real-time systems; human computation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
CHI 2018, April 21-26, 2018, Montreal, QC, Canada.
© 2018 Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-5620-6/18/04...$15.00
DOI: https://doi.org/10.1145/3173574.3174041

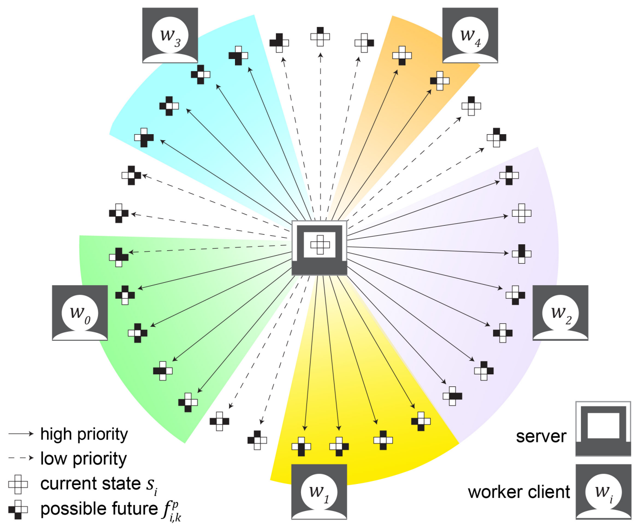

Figure 1. The look-ahead approach prefetches crowd worker responses to possible future states based on the current state and the agent's transition function. Each possible future state is prioritized by the likelihood of encountering it in the near future. Each crowd worker "covers" as much of the future state space as they can within a short time horizon.

INTRODUCTION

In the past decade, some of the largest advances in the capabilities of crowd-powered systems have come from significant decreases in latency. A task initially requiring hours or days could later be completed in minutes [4], and eventually seconds [15, 13, 3]. With each advance in latency reduction, it became feasible to solve entirely new types of problems using crowdsourcing and human computation. However, throughout all of these advances, one fundamental constraint held true: system latency was lower-bounded by how quickly a worker could complete their task upon arrival. This paper introduces a hybrid intelligence approach that aims to break this speed barrier and demonstrates that, for problems that can be modeled as MDPs, we can lower overall system response latency to just a couple of *milliseconds*.

Crowds in Two Milliseconds

To achieve this latency reduction, we leverage continuous real-time crowdsourcing, which keeps workers engaged throughout an ongoing task to get responses as quickly as half a second [15, 9], and introduce the look-ahead approach, which elicits worker responses to states that may be encountered in the near future. In this way, the look-ahead approach is akin to *prefetching* in OS-level memory management [16], allowing us to "teach" the computer what to do in a specific state immediately before it is encountered. More formally, this can be thought of as just-in-time (JIT) training of an MDP agent's policy (i.e., the function that determines which action the agent should take in response to a given state). Typically, an optimal (or near-optimal) policy is learned over repeated game trials, but JIT policy training elicits and aggregates crowd worker responses to generate a policy on-the-fly. This avoids the need for trial-and-error learning, which may not be possible in critical real-world control settings, such as autonomous vehicle navigation [1]. When a given state is encountered, the computer "instantaneously" returns the cached policy response—a process that only takes a couple of milliseconds.

Overview

Our hybrid intelligence approach opens a new frontier for crowd-powered systems, with potential applications to real-world problems that require human responses at machine-level speeds. In short, this paper contributes the following:

- The idea of *instantaneous crowdsourcing*, a new class of crowdsourcing system that delivers human responses at machine-level speeds.
- The first instantaneous crowdsourcing workflow, the *look-ahead approach*, enabling just-in-time training of an agent's policy by prefetching responses to possible future states.
- *Bolt*, an instantaneous crowdsourcing system for empirically evaluating just-in-time policy training using the look-ahead approach on the Lightning Dodger testbed environment.

BACKGROUND AND RELATED WORK

The look-ahead approach builds most directly upon the literature in prefetching and recruiting, real-time crowdsourcing, and hybrid intelligence workflows.

Prefetching and Recruiting

Our approach is analogous to cache prefetching in OS-level memory management. Prefetching achieves significant on-average latency reductions by moving operation results from main memory to the cache—not because these results are needed immediately, but because they are expected to be needed in the near future [16]. Similarly, the look-ahead approach achieves a two order of magnitude latency reduction by prefetching and caching crowd worker responses in advance. Within the crowdsourcing literature, significant latency reductions have been achieved by recruiting workers to wait in a retainer prior to task arrival [3, 4]. The retainer model can also be thought of as a form of prefetching where workers are prefetched, or "pre-recruited," before they are needed. Pre-recruiting workers makes them available to start a task as soon as the end-user request is made, but their responses will still be delayed by the time it takes to complete the task itself. In this paper, we introduce the idea of having workers complete the task ahead of time—before the responses are needed—so that the output can be returned to the end-user in an instant.

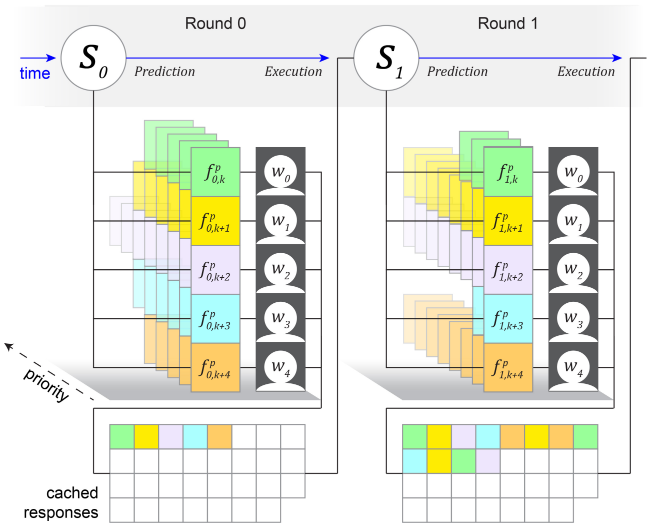

Figure 2. Look-ahead Approach: During each round, workers (w) first respond to as many possible future states (f^p) as they can (the prediction phase). Then, their responses are cached and later returned in response to subsequently encountered states (the execution phase).

Real-time Crowdsourcing

Prior work in real-time crowdsourcing has demonstrated the feasibility of incorporating real-time, collaborative input into the execution of control tasks. Most relevant to our work, real-time crowdsourcing and various aggregation methods have been successfully applied to remote-control driving tasks [15], aerial navigation simulations [19], and collaborative interface control [20, 22]. In general, these methods enable the crowd to behave as a single, high-performing actor [5], in some cases achieving on-demand, real-time responses in less than three seconds [13].

Hybrid Intelligence Workflows

Active learning and human-in-the-loop machine learning methods are well-known forms of semi-supervised learning [24, 23, 7, 18]. Our approach is similar in that it integrates human and machine intelligence, resorting to human computation where automatic processes fall short, and vice versa [6, 8]. It differs, however, in that it does not only use human input as a superior groundtruth for subsequent model training [12, 11] (although it permits this as well), but as JIT policy training, where human input is collected, cached, and utilized by the system to generate a policy on-the-fly.

LOOK-AHEAD APPROACH

We introduce the *look-ahead approach* to facilitate the creation of *Bolt*, the first *instantaneous crowdsourcing* system. In this section, we present a high-level view, which is later grounded in an arcade-style game environment (Lightning Dodger) and formalized as a Markov Decision Process (MDP), a popular and broadly applicable formalism that is common to the AI literature. The look-ahead approach works by "looking ahead" to near-future states (e.g., by predicting what moves might be made in the upcoming rounds of a game), and prefetching crowd worker responses to these states. We call this the *prediction phase*. During this phase, possible future states are

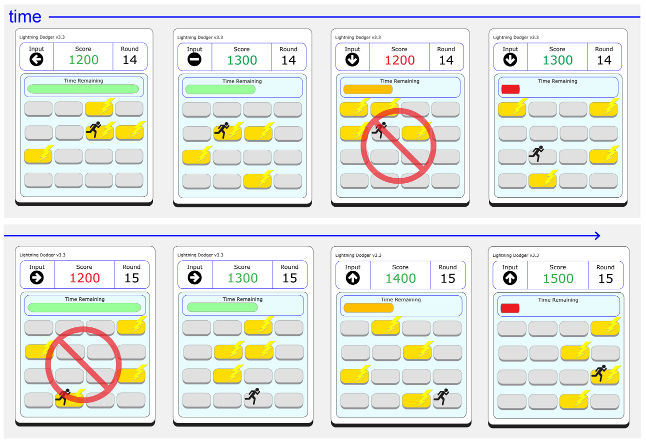

Figure 3. The Lightning Dodger game interface. During the prediction phase, workers submit responses to as many possible future states as they can and receive a reward for successfully dodging the lightning.

generated and sent to crowd workers who submit responses that get cached by the system. Then, when the system actually encounters one of these possible future states, it simply returns the cached response. We call this the *execution phase*. In other words, the look-ahead approach has crowd workers complete a simulated task (the prediction phase) a few time steps ahead of the actual task (the execution phase), allowing the system to return responses to the actual task state as quickly as it can retrieve the corresponding response from memory (Figure 2).

Additionally, responses from multiple workers can be aggregated during the prediction phase in order to improve the accuracy of the response returned during the execution phase. In general, faster, more accurate workers will improve the performance of any crowdsourcing system, but the look-ahead approach is unique in that it can achieve performance gains from slower workers as well. As each worker explores more of the upcoming state space, the system can prioritize their responses based on their real-time performance. For example, in our experiments, we give priority to responses from slower workers (who tend to give more accurate, careful responses).

In principle, the look-ahead approach will be applicable to task environments that fit the MDP formalism. This is because the approach requires that a set of *possible* future states can be generated for any encountered state. In this paper, we apply the approach to tasks where the state space, agent action set, and transition function are all specified (i.e., available to the system), but the reward function is left unspecified. Before we formally define these terms, we first introduce an example of one such task environment: the Lightning Dodger game.

LIGHTNING DODGER GAME

As a proof-of-concept testbed for just-in-time training via the look-ahead approach, we developed a controlled task environment: a gridworld game called Lightning Dodger. Lightning Dodger is flexible enough to test instantaneous crowd systems on tasks of variable complexity. The goal of players in the game is to avoid getting struck by lightning by dodging it once it begins to strike (Figure 3)—a task that requires considerable speed! In this section, we detail the mechanics of the game.

The Lightning Dodger gridworld is a $N \times M$–cell rectangular grid with a single-player agent (the dodger) and B adversary players (lightning bolt locations). At each cell, the agent can move `up`, `down`, `left`, `right`, or `stay`. The world "wraps-around," meaning that moving off one edge of the grid results in the agent appearing on the opposite edge. The agent's goal is to avoid getting caught in the same cell as an adversary (lightning bolt). Effectively dodging the lightning results in a reward of r^+ (+100 in our case), while failing to dodge (or failing to input any action) results in a reward of r^- (−100) and the display of a large red "incorrect" indicator. Task complexity is varied by adjusting input parameters, such as grid dimension, round duration, the presence of obstacles, the observability of the grid, and the number of adversaries and their behavior.

TASK ENVIRONMENT SPECIFICATION

In the last section, we detailed the game mechanics of our Lightning Dodger testbed. In this section, we model the game as a Markov Decision Process (MDP), and make a few additions to the canonical notation. The MDP formalism is common to the reinforcement learning (RL) literature, which has recently seen frequent use of arcade-style games as testbed environments for state-of-the-art methods [2, 17]. Although these games are far from real-world settings in terms of complexity, they are used because they are non-trivial yet tractable for current methods. Similarly, for the purpose of introducing and validating our method, we demonstrate the look-ahead approach on a tractable Lightning Dodger setup. In principle, any game (or task environment) capable of being modeled similarly will be amenable to our approach.

The MDP is a discrete-time, single-player game where the agent takes an action at each time step. We borrow from the canonical notation [21]:

$\mathcal{S}$ a set of states: $s \in \mathcal{S}$
$\mathcal{A}$ a set of agent actions: $a \in \mathcal{A}$
$P_a(s, s')$ the transition probability from s to s' given a
$R_a(s, s')$ the reward on transition from s to s' given a

At each time step t, the agent observes state s_t and takes action $a_t \in \mathcal{A}(s_t)$ where $\mathcal{A}(s_t)$ is the set of actions possible in state s_t. Supposing that the agent is in $s_t = s$, the action a will lead to $s_{t+1} = s'$ with probability $P_a(s, s')$, the value of which will depend on whether the agent's transition function is deterministic or stochastic. In our experiments, we consider the simplest Lightning Dodger setup: no gridworld obstacles, a deterministic transition between states, and stochastic lightning bolt locations. In this setup, $\mathcal{A}(s_t)$ is the set containing `up`, `down`, `left`, `right`, `stay` for all s_t (since each move is available to the agent at each cell), and $P_a(s, s') = 1$ (since there is no indeterminacy about the resulting state given the agent's intended action). In a typical RL problem, the action taken at each state is determined by a policy that tries to maximize its reward from R and is learned over repeated trials. However, our approach differs from typical RL in that it "learns" (generates) a policy by having human crowds provide responses to never-before-seen future states. This approach doesn't require that R be specified at all, so long as workers implicitly understand the potential outcomes of their actions.

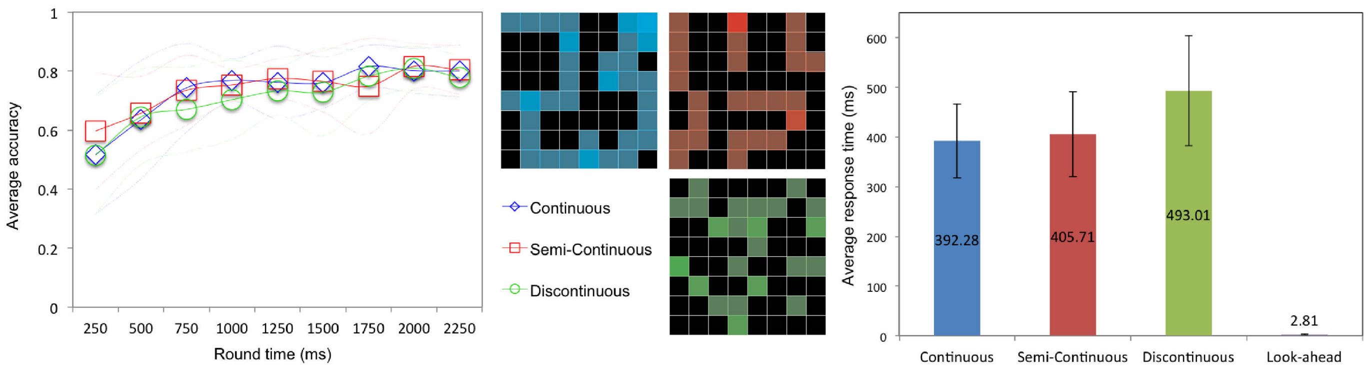

Figure 4. Left: Breaking path continuity did not result in any significant performance decline. Center: A heat map of agent movement across multiple rounds for each path type on an 8×8 gridworld (n.b., the heat maps depict agent movement across multiple rounds, so the agent may end and start a round in two adjacent cells). Right: Average response times for each path type (n.b., the look-ahead approach in the far right column).

Therefore, responses from crowd workers will be most beneficial in cases where manually specifying a reward function is difficult, as in many real-world tasks such as autonomous vehicle navigation [1]. While the look-ahead approach does not learn a policy in the traditional sense, prefetching worker responses to possible future states can be seen as JIT policy generation. We introduce the following notation:

$\mathcal{F}^p$ a set of possible future states: $f^p \in \mathcal{F}^p$
$\mathcal{W}$ a set of workers: $w \in \mathcal{W}$

At each time step t, the set of possible future states will depend on the actions possible in state s_t given the agent's transition function; formally, $f_t^p \in \mathcal{F}^p(\mathcal{A}(s_t))$. The possible future states are then distributed to workers depending on the number available (cardinality of $\mathcal{W}$) and a chosen priority metric (Figure 2). In our setup, the simplest, optimal priority metric is a straightforward breadth-first search (BFS) of the state space centered at the agent's current position. Workers submit as many responses as they can during the prediction phase, making it possible (with enough workers) to look ahead to more possible future states. However, looking further into the future forgoes an opportunity to increase robustness to individual worker error by having the system prefetch and aggregate multiple responses to each possible future state. In the next section, we assess these trade-offs empirically by evaluating the relative performance of various group sizes and aggregation methods.

EVALUATION

How should the look-ahead approach prioritize possible future states for prefetching, and in what order should they be shown to workers? Does this ordering affect worker performance? In this section, we answer these questions by reporting on two experiments conducted via Amazon Mechanical Turk (MTurk) in which crowd workers played the Lightning Dodger game. For each experiment, workers first completed a 3-round training phase (to familiarize themselves with the game controls) before completing a 20-round testing phase. We compensated workers 30-40 cents for a task taking approximately 100 seconds resulting in an effective base rate of \$10-14 per hour, with 100% bonusing for exceptional performance. We configured our Lightning Dodger testbed as a 4×4 grid with four random adversary (lightning bolt) positions (Figure 3). As previously noted, this task environment is scalable to more complex state spaces by introducing various obstacles and behaviors. However, for these preliminary experiments, we chose to make the adversary behavior stochastic, the agent's transition function deterministic, and the state space entirely observable with no obstacles. In order to further reduce complexity and cost, we used an *agent* state representation that included only those grid cells *relevant* to the agent's current position and action set—specifically, the vicinity of five cells reachable from the agent's current location, and all possible combinations of four lightning bolts within these cells (Figure 1). Given our Lightning Dodger setup, this agent state representation includes all the information needed to successfully dodge lightning, and would be applicable even to a grid-world of infinitely large dimension. This setup gives us a sufficiently complex representation of the problem at hand, but one that is also tractable in terms of state space size.

Impact of Path Type on Worker Performance

We first evaluate crowd worker performance while varying the order in which possible future states are shown to workers. This is important because the system may need to show workers a variety of future states that could have little or no relation to one another. Understanding if this impacts worker performance guides the design of our final approach.

We say that a state ordering maintains **path continuity** if the next state shown to a worker is what the worker expects as the outcome of their given input (i.e., if the agent follows the path expected by the worker). In other words, if the worker inputs `left` at time step t, does the agent (dodger) appear in the cell to the left of its current position at time step $t + 1$? This may not be the case if the agent has a stochastic transition function (i.e., if in state s, the worker inputs action a and the agent takes this action to reach s' with some probability $P_a(s, s') < 1$), or if the look-ahead approach sends possible future states to workers by doing a BFS traversal of the state space. A BFS traversal would result in a path continuity break that may be disorienting from the worker's perspective (i.e., the agent may appear to arbitrarily "teleport" to other parts of the grid). Such a disconnect between the agent's *actual* transition function and the *apparent* transition function (observed by the worker during gameplay) may negatively affect worker performance.

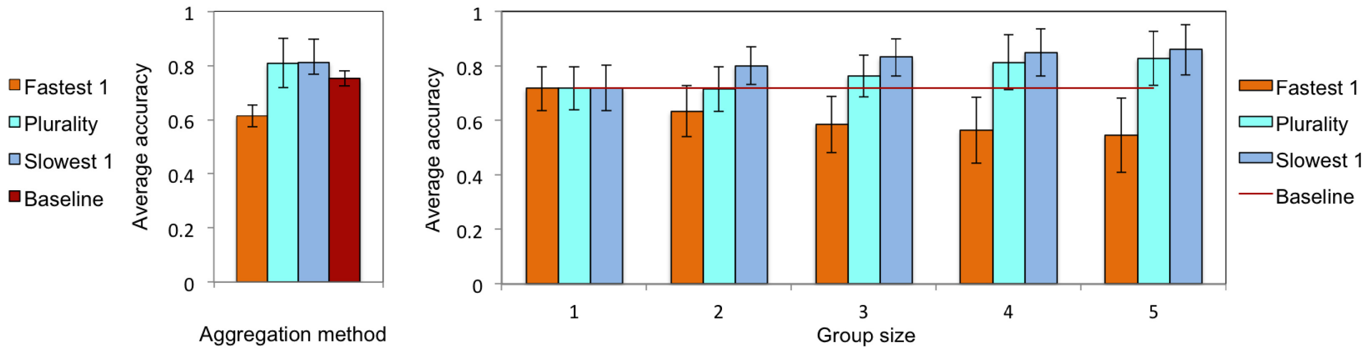

Figure 5. Left: Average accuracy by aggregation method. Right: Averagge accuracy by group size, for each aggregation method.

To test this hypothesis, we considered three path types (i.e., methods of distributing possible future states to workers) with varying levels of path continuity:

Continuous Path : Maintains path continuity across both the prediction phase and the execution phase. Workers observe no path breaks between any state transitions.

Semi-Continuous Path : Maintains path continuity during the prediction phase, but workers observe a path break when the system returns a response during the execution phase.

Discontinuous Path : Does not maintain path continuity during either phase. Workers observe path breaks between every state transition.

We hypothesized that maintaining path continuity will benefit performance because workers will spend less time situating themselves on the grid than they would if the agent were to appear at a different position at the start of each new round (Figure 4-center). We tested each of the three path types with ten unique crowd workers, across nine prediction phase durations, for a total of 270 unique tasks.

Results

Unexpectedly, there was no statistically significant performance difference between the three path types ($p > 0.5$) for all comparisons between types (Figure 4, left). However surprising, this result bodes well for the look-ahead approach. Maintaining path continuity for each individual worker comes at the expense of covering more of the highest-priority state space since it requires allowing workers to follow their own branch down the state space tree (when a straightforward BFS traversal would be optimal in our case). Since there is no observable difference between path types, workers are directed in a manner that allows the system to cover the entirety of the state space more quickly than if the system attempted to maintain path continuity for each worker.

As expected, there was a significant performance difference between short and fast prediction phase durations (Figure 4, left). Worker accuracy increased by 16.33% ($p < 0.0001$) between a 500ms prediction phase with an average of 64.56% (SD=3.77) and a 2000ms prediction phase with an average of 80.89% (SD=0.17). This confirms our intuition that workers are more accurate when given more time to respond.

Live Evaluation of the Look-Ahead Approach

In addition to testing varying levels of path continuity, we evaluated the efficacy of the look-ahead approach by conducting ten real-time experiments via MTurk, each with five worker participants for a total of 50 unique participants. We recruited workers to each game instance using the retainer model [3] in LegionTools [14, 10], ensuring that worker participation was synchronized for the duration of the task. Worker responses cached during the prediction phase were aggregated using plurality vote in real time, and returned during the execution phase. We used the discontinuous path type, allowing the system to do a straightforward BFS traversal of the state space.

Results

There was no significant latency reduction between the three path types ($p > 0.1$) for any comparison between types (Figure 4, right). However, the look-ahead approach outperforms the fastest workers by approximately *two orders of magnitude*. Discarding outliers that are more than two standard deviations from the mean (two iterations), the look-ahead approach achieves a median response time of 2ms, the time it takes for the system to aggregate and return cached responses. Crowd responses of anywhere near this speed (even on a small state space) were not possible prior to this work.

Post-Hoc Aggregation

Having achieved a response time reduction that demonstrates the efficacy of instantaneous crowdsourcing, we also explored whether the look-ahead approach maintains levels of accuracy comparable to individual workers. Using the data collected during our real-time experiment, we tested the effects of varying crowd size (number of responses collected per state) and aggregation method, over all possible adversary configurations.

Slower responses are more accurate

For small task environments, the state space is quickly covered making it possible to collect many worker responses to the same possible future state. Aggregating these responses gives the look-ahead approach robustness to individual worker error. We assessed the performance of three aggregation methods:

Fastest 1 : the cached action with minimal response time
Slowest 1 : the cached action with maximal response time
Plurality : the most-frequently-occurring cached action

As a baseline, the unaggregated average accuracy was 75.23% (SD=2.89). *Slowest 1* resulted in an 6.0% increase over the baseline ($p < 0.0001$) with an average accuracy of 81.23% (SD=7.88), while *Plurality* resulted in an 5.86% increase over the baseline ($p < 0.01$) with an average accuracy of 81.09% (SD=8.2). Notably, *Fastest 1* resulted in a significant decrease in average accuracy of 13.8% in comparison with the baseline ($p < 0.0001$), suggesting that workers submitting most quickly may not be making careful judgments about the outcome of their actions (Figure 5, left).

Larger groups are more accurate

In addition to varying aggregation methods, we also tested the effects of varying group size. Average accuracy improved significantly between a group of size five and a group of size one under the *Slowest 1* aggregation method ($p < 0.01$). Average accuracy for *Plurality* and *Slowest 1* consistently improved with each additional worker (Figure 5, right). However, *Fastest 1*'s performance further decreased with each added worker, suggesting that workers are more likely to be faster by way of inaccuracy, rather than skill.

LIMITATIONS AND DISCUSSION

We have demonstrated that the Bolt system can deliver crowd responses at machine-level speeds when the search space is small enough to be explored exhaustively. In principle, our reported performance (2ms response time; accuracy ~ 80%) will scale to larger state spaces so long as available resources (i.e., budget and worker availability) are scaled proportionately. However, this may not be feasible for "real-world" state spaces. To what extent, then, can the system maintain instantaneous speeds and high accuracy for increasingly large and realistic state spaces? Prior work has engaged up to 60-70 crowd workers simultaneously via a retainer [14], and in our experiments, a single worker processes 1-2 states per second. Additionally, processing a single state costs approximately 0.25-1 cents on average. These benchmarks suggest that the Bolt system—as currently implemented—could process over 500 states in 4 seconds using 65 workers at cost of roughly $3. Further performance gains may be achieved by using more sophisticated response aggregation methods and possible future sampling techniques.

CONCLUSION AND FUTURE WORK

We have introduced the *look-ahead approach* to facilitate the creation of *Bolt*, the first *instantaneous crowdsourcing* system. These systems use real-time crowdsourcing in a hybrid intelligence workflow to provide just-in-time training of automated agents, achieving final response latencies on the order of milliseconds (median of 2ms), instead of seconds, hours, or days. Our experiments with crowd workers demonstrate that a two order of magnitude speedup from the baseline response time is possible via our approach, and accuracy improvements are attainable via aggregation. While these results demonstrate the promise of the look-ahead approach and open avenues for instantaneous crowd-powered systems, scaling to larger state spaces using probabilistic sampling and queuing techniques remains an exciting and open problem for future work to explore.

ACKNOWLEDGMENTS

The authors would like to acknowledge Patrick Baudisch and Scott Hudson for thought-provoking discussions on primordial versions of this idea. This work was supported in part by the MCubed program and MCity at the University of Michigan, and by the Toyota Research Institute.

REFERENCES

1. Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML '04)*. ACM, New York, NY, USA. DOI:http://dx.doi.org/10.1145/1015330.1015430

2. Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. 2015. The Arcade Learning Environment. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 4148–4152. http://dl.acm.org/citation.cfm?id=2832747.2832830

3. Michael S. Bernstein, Joel Brandt, Robert C. Miller, and David R. Karger. 2011. Crowds in Two Seconds: Enabling Realtime Crowd-powered Interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 33–42. DOI: http://dx.doi.org/10.1145/2047196.2047201

4. Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. 2010. VizWiz: Nearly Real-time Answers to Visual Questions. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. ACM, New York, NY, USA, 333–342. DOI: http://dx.doi.org/10.1145/1866029.1866080

5. Jeffrey P. Bigham, Walter S. Lasecki, and Jake Wobbrock. 2015. Target Acquisition and the Crowd Actor. *Human Computation* 1 (2015), 101–131. DOI: http://dx.doi.org/10.15346/hc.v1i1.2

6. Aleksey Boyko and Thomas Funkhouser. 2014. Cheaper by the Dozen: Group Annotation of 3D Data. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 33–42. DOI: http://dx.doi.org/10.1145/2642918.2647418

7. J. Costa, C. Silva, M. Antunes, and B. Ribeiro. 2011. On Using Crowdsourcing And Active Learning To Improve Classification Performance. In *2011 11th International Conference on Intelligent Systems Design and Applications*. 469–474. DOI: http://dx.doi.org/10.1109/ISDA.2011.6121700

8. Peng Dai, Mausam, and Daniel S. Weld. 2011. Artificial Intelligence for Artificial Artificial Intelligence. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI'11)*. AAAI, 1153–1159. http://dl.acm.org/citation.cfm?id=2900423.2900606

9. Gabriel V. de la Cruz, Bei Peng, Walter S. Lasecki, and Matthew E. Taylor. 2015. Towards Integrating Real-Time Crowd Advice with Reinforcement Learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces Companion (IUI '15)*. ACM, New York, NY, USA, 17–20. DOI: http://dx.doi.org/10.1145/2732158.2732180

10. Mitchell Gordon, Jeffrey P. Bigham, and Walter S. Lasecki. 2015. LegionTools: A Toolkit + UI for Recruiting and Routing Crowds to Synchronous Real-Time Tasks. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15 Adjunct)*. ACM, New York, NY, USA, 81–82. DOI: http://dx.doi.org/10.1145/2815585.2815729

11. Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L. Isbell, and Andrea Thomaz. 2013. Policy Shaping: Integrating Human Feedback with Reinforcement Learning. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 2625–2633. http://dl.acm.org/citation.cfm?id=2999792.2999905

12. W. Bradley Knox and Peter Stone. 2009. Interactively Shaping Agents via Human Reinforcement: The TAMER Framework. In *Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP '09)*. ACM, New York, NY, USA, 9–16. DOI: http://dx.doi.org/10.1145/1597735.1597738

13. Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. 2012. Real-time Captioning by Groups of Non-experts. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 23–34. DOI:http://dx.doi.org/10.1145/2380116.2380122

14. Walter S. Lasecki, Mitchell Gordon, Danai Koutra, Malte F. Jung, Steven P. Dow, and Jeffrey P. Bigham. 2014. Glance: Rapidly Coding Behavioral Video with the Crowd. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 551–562. DOI: http://dx.doi.org/10.1145/2642918.2647367

15. Walter S. Lasecki, Kyle I. Murray, Samuel White, Robert C. Miller, and Jeffrey P. Bigham. 2011. Real-time Crowd Control of Existing Interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 23–32. DOI: http://dx.doi.org/10.1145/2047196.2047200

16. Jaekyu Lee, Hyesoon Kim, and Richard Vuduc. 2012. When Prefetching Works, When It Doesn't, and Why. *ACM Transactions on Architecture and Code Optimization* 9, 1, Article 2 (March 2012), 29 pages. DOI: http://dx.doi.org/10.1145/2133382.2133384

17. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *Computing Research Repository* 1312.5602 (2013). http://arxiv.org/abs/1312.5602

18. Bei Peng, James MacGlashan, Robert Loftin, Michael L. Littman, David L. Roberts, and Matthew E. Taylor. 2016. A Need for Speed: Adapting Agent Action Speed to Improve Task Learning from Non-Expert Humans. In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '16)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 957–965. http://dl.acm.org/citation.cfm?id=2936924.2937065

19. Elliot Salisbury, Sebastian Stein, and Sarvapali Ramchurn. 2015. Real-time Opinion Aggregation Methods for Crowd Robotics. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 841–849. http://dl.acm.org/citation.cfm?id=2772879.2773261

20. D. Song, A. Pashkevich, and K. Goldberg. 2003. Sharecam Part II: Approximate And Distributed Algorithms For A Collaboratively Controlled Robotic Webcam. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '03)*, Vol. 2. 1087–1093. DOI: http://dx.doi.org/10.1109/IROS.2003.1248789

21. Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning* (1st ed.). MIT Press, Cambridge, MA, USA.

22. Saiganesh Swaminathan, Raymond Fok, Fanglin Chen, Ting-Hao (Kenneth) Huang, Irene Lin, Rohan Jadvani, Walter S. Lasecki, and Jeffrey P. Bigham. 2017. WearMail: On-the-Go Access to Information in Your Email with a Privacy-Preserving Human Computation Workflow. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 807–815. DOI: http://dx.doi.org/10.1145/3126594.3126603

23. Sudheendra Vijayanarasimhan and Kristen Grauman. 2014. Large-Scale Live Active Learning: Training Object Detectors with Crawled Data and Crowds. *International Journal of Computer Vision* 108, 1-2 (May 2014), 97–114. DOI:http://dx.doi.org/10.1007/s11263-014-0721-9

24. Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G. Dy. 2011. Active Learning from Crowds. In *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML'11)*. Omnipress, USA, 1161–1168. http://dl.acm.org/citation.cfm?id=3104482.3104628