# Towards Integrating Real-Time Crowd Advice with Reinforcement Learning

**Gabriel V. de la Cruz Jr.,**
**Bei Peng**
School of EECS
Washington State University
gabriel.delacruz@wsu.edu,
bei.peng@wsu.edu

**Walter S. Lasecki**
Computer Science Department
University of Rochester
wlasecki@cs.rochester.edu

**Matthew E. Taylor**
School of EECS
Washington State University
taylorm@eecs.wsu.edu

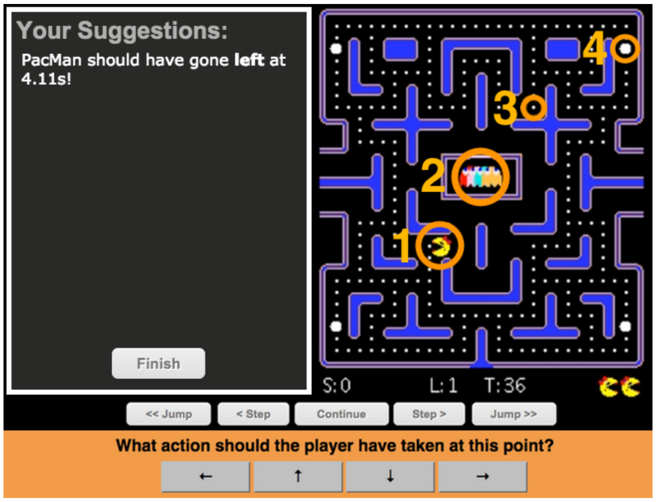
## ABSTRACT

Reinforcement learning is a powerful machine learning paradigm that allows agents to autonomously learn to maximize a scalar reward. However, it often suffers from poor initial performance and long learning times. This paper discusses how collecting on-line human feedback, both in real time and *post hoc*, can potentially improve the performance of such learning systems. We use the game Pac-Man to simulate a navigation setting and show that workers are able to accurately identify both when a sub-optimal action is executed, and what action should have been performed instead. Demonstrating that the crowd is capable of generating this input, and discussing the types of errors that occur, serves as a critical first step in designing systems that use this real-time feedback to improve systems' learning performance on-the-fly.

## INTRODUCTION

Reinforcement learning [7] is a very flexible, robust approach to solving problems. However, early in the training process much of the problem space is unexplored, often resulting in poor performance because reasonable policies are only discovered after a considerable amount of trial-and-error. In this paper, we propose the idea of using on-demand human intelligence, available via crowdsourcing platforms such as Amazon Mechanical Turk, to provide immediate feedback to reinforcement learning systems based on the intuition and experience of the human observer.

To test whether crowd workers are able to accurately provide such advice, we perform a set of experiments that measure the crowd's ability to generate just-in-time warnings to an agent playing Pac-Man. First, we establish that the crowd can collectively identify the correct point at which an error occurs with over 91% accuracy. Second, we demonstrate that not only can this mistake identification be done in real time with a mean latency of just 0.39 seconds, but also that workers are able to identify what the optimal move *would* have been been.



PacMan should have gone **left** at 4.11s!

What action should the player have taken at this point?

**Figure 1: This screenshot shows the web interface of the user study with game layout, and components of the Pac-Man game: 1) Pac-Man, 2) 4 Ghosts, 3) Pills, and 4) Power Pills.**

Third, we compare the crowd's performance in this real-time setting with an offline "review" setting where game playback can be controlled and replayed. In this setting, mistakes can be better estimated, with a mean distance from the correct position of just 0.15 seconds.

This work is the first research to establish the crowd's ability to react to mistakes made by an intelligent agent in real time, and provide accurate guidance on a preferred alternative action. Our work informs the design of future systems that use human intelligence to guide untrained systems through the learning process, without limiting systems to only learn from their mistakes far after they make them.

The contributions of this paper are to:

- Present the idea that on-line crowds can provide very accurate assistance to learning agents by using real-time data.
- Demonstrate that crowd workers can respond quickly and accurately enough to provide just-in-time feedback.
- Show that workers can also improve their accuracy in *post hoc* review settings for use in future situations.

*IUI'15 Companion*, Mar 29 - Apr 01, 2015, Atlanta, GA, USA
ACM 978-1-4503-3308-5/15/03.
http://dx.doi.org/10.1145/2732158.2732180

## BACKGROUND AND RELATED WORK

Reinforcement learning has a history of succeeding on difficult problems with little information. This paper leverages the on-policy learning algorithm Sarsa [7]. A Sarsa agent learns to estimate Q-values, representing the estimated total reward the agent would receive in a state $s$, execute action $a$, and then follows the current policy until the episode is terminated. Over time, this type of temporal difference learning allows the agent to learn a (near) optimal policy that collects as much reward as possible, in expectation.

While autonomous methods like Sarsa have many empirical successes and sound theoretical underpinnings, if a human is available to provide useful information, the agent is often able to learn much faster. For example, a user could make judgements about the agent's performance by providing human reward [3, 6] or providing demonstrations. Finally, when a human can temporarily control the agent to perform the correct behavior, learning from demonstration techniques [1].

Most related to this paper is existing work done on crowdsourcing control and recognition tasks. Human control of robots has been previously explored in the context of a robot Ouija board and navigation setting [2]. The Robot Management System [10] (RMS) also uses on-line contributors to crowdsource human-robot interaction studies. RMS used groups of participants working from their home computers to practice controlling a robot using camera views and a web-based control interface.

Legion [4] explored using crowd workers to collaboratively control a robot in real time. This was the first work to show that on-demand human intelligence could be used to control a robot when an automatic system is unable to proceed. Legion:AR [5] showed that an active learning approach could be used in an activity recognition setting to call on crowd support for an action-labeling task only when needed. In both systems, low-latency responses were achieved by keeping the crowd continuously engaged with a task for a period of time. However, complete crowd control does not let the system effectively evaluate its own policy. In this work, we explore if and how we can use real-time crowds in an advisory role, without needing the crowd to directly control the Pac-Man avatar, while still maintaining exceptional response speeds.

## EXPERIMENTAL DESIGN

Our Pac-Man agent (see Figure 1) used Sarsa to learn a near-optimal policy to win the game while earning as many points as possible using an existing open learning implementation [8]. Due to the large state space, the agent uses seven high-level features for function approximation to learn a continuous Q-value function. Pac-Man code is available from `http://www.eecs.wsu.edu/~taylorm/13PacMan.zip`.

To generate the videos used in the user study, we recorded Pac-Man being controlled by a human who intentionally made different types of mistakes. Then, we picked 10–14 seconds which contained one (and only one) mistake. Q-values for the agent's trajectory were also recorded, confirming that the human-created mistakes had lower Q-values than the "correct" action. We create four videos where each contained a mistake: Video 1) moving so that Pac-Man is trapped by one or more ghosts, Video 2) not moving towards an edible ghost after eating a power pill, Video 3) taking an empty path instead of going for pills when they are no risk, and Video 4) not going for all edible ghosts that are within close range.

To study the hypothesis that crowd workers can provide information useful to reinforcement learning agents, we consider four settings. First, a video of Pac-Man is played only once (*real-time*) or the worker can view it multiple times (*review*). Second, the worker may be asked to identify the time at which the mistake is made (*Mistake Identification*), or asked to identify both the mistake time as well as suggest the optimal action (*Action Suggestion*).

We want to measure the performance of users in identifying the point at which a mistake is made and suggesting optimal action Pac-Man should have executed. To evaluate worker actions, we can compare to recorded Q-values.

## USER STUDIES

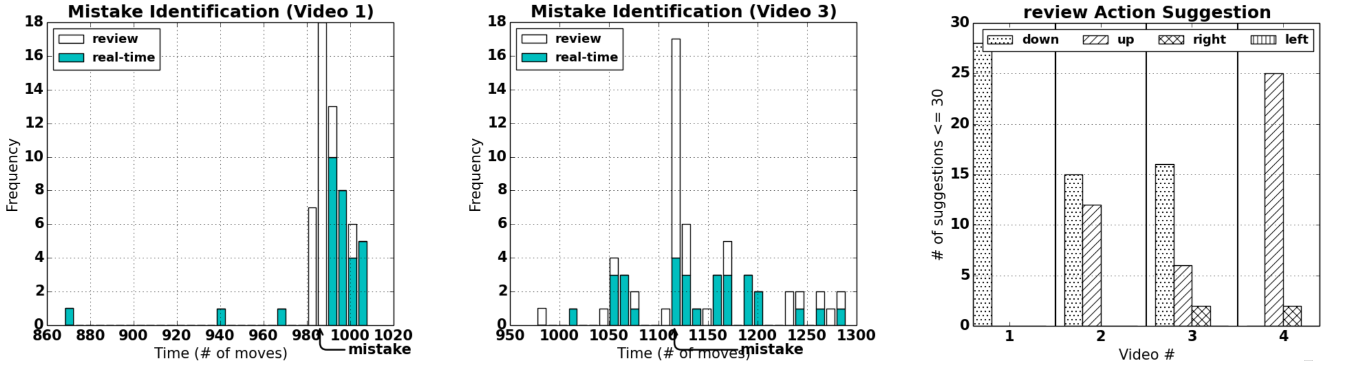
Workers were first shown instructions describing the task, as well as the rules of Pac-Man. During a preliminary test of the web interface, we found that workers would sometimes identify mistakes *before* the sub-optimal action was executed, anticipating the mistake. We provided explicit instructions to workers to encourage them to identify the exact time at which a mistake was made. Workers were then directed to a tutorial which asked them to complete an example task using the marking interface. After the tutorial, workers will watch a new video and must press a button (see Figure 1) as soon as they observe a mistake.

We recruited crowd workers from Amazon Mechanical Turk (AMT) for our experiments. While AMT provides immediately, programmatic access to crowds, it also poses a number of challenges, including that workers: 1) are unlikely to be experts, 2) may not take the task seriously and not read the instructions, and 3) may intentionally select incorrect times/actions. Our methods need to be robust to these challenges, unlike in Learning from Demonstration, where demonstrations are typically assumed to be optimal.

16 Human Intelligence Tasks (HITs) on AMT encompassed our four different types of experiments. Each experiment was tested with 4 different videos. We collected data from 30 unique workers per HIT and every worker was paid 25 cents.

## RESULT ANALYSIS

This section presents the results of our study in three parts. First, we establish that the crowd can identify the mistake with high accuracy. Second, we demonstrate that not only can Mistake Identification be done in real-time but that workers can also successfully identify what the optimal "correct" move would have been. Third, we compare the crowd's performance in the real-time setting with offline "review" setting and show that if additional time is available, even more accurate performance can be achieved.



(a) A histogram of workers' suggestions using Video 1 (mistake time: 986).

(b) A histogram of workers' suggestions using Video 3 (mistake time: 1116).

(c) Number of times each action was suggested by workers (Optimal: 1–down, 2–down, 3–down, 4–up).

**Figure 2: Selected exemplar results from our 16 Amazon Mechanical Turk experiments.**

### Mistake Identification

Our performance measure is based on how many workers can correctly identify and suggest a time that is close to the correct mistake time. Histograms provide a visual representation of the accuracy of workers in different settings. The mistake times are reported as game move numbers, which are 986, 1809, 1116 and 334, for Videos 1–4, respectively. These video clips are 10 to 14 seconds long, corresponding to 250–350 total game moves, and the mistakes located roughly three quarters of the way through the clip. However, because Pac-Man moves continually, it is difficult for workers to identify the exact frame when the mistake was executed.

To quantify how accurate the workers were, we calculated the difference between the actual mistake time and the identified mistake time, where zero corresponds to a perfect answer. We selected a threshold of 30 actions, roughly 1 second, so that any answer within ±1 second will be counted as correctly identifying the mistake. Figure 2(a) shows the distribution of workers' answers where responses within the 956–1016 moves range are considered to be correct, showing only two errant responses.

To compute the mean difference between the time reported by a worker and the real error time $\mu_{diff}$, we use: $\mu_{diff}(AMT_k) = \frac{\sum_{i=1}^{n} |t_{w_i} - t_m|}{n}$, where $k$ is the group number, $n$ is the total number of workers per group that are within the threshold, $t_{w_i}$ is the $i$th worker's suggested time, and $t_m$ is the correct mistake time. The standard deviation is also computed using: $\sigma_{diff}(AMT_k) = \sqrt{\frac{\sum_{i=1}^{n} (\mu_{diff}(AMT_k) - |t_{w_i} - t_m|)^2}{n}}$, where a low value indicates suggestions are tightly clustered.

To establish that workers can correctly identify where and when mistakes occur in our game, we count the number of people who correctly identified the mistake. Video 1's review setting collective percentage of correct events has the highest over all four videos with 98.3%. This is followed by Videos 4 and 2, with 88.0% and 86.6% respectively. And Video 3 has the lowest accuracy with 68.4%. This observed high percentage of correct events from the three videos suggests that the crowd can identify a mistake in many cases.

It is also important to point out that there are instances in which the mistakes are more subtle, making it harder to identify. Video 3 has the lowest accuracy, and the Mistake Identification experiment has the least percentage of correct answers at 56.7%. However, the sparsity of the data as shown in Figure 2(b) suggests that the mistake was harder to find.

In summary, these results established that, in most cases, workers can identify a mistake in the Pac-Man game with an overall accuracy for review Mistake Identification at 80% and an accuracy of 91% for review Action Suggestion.

### Optimal Action Identification

Given that workers can correctly identify mistakes, we next consider whether they can also accurately provide the action that should have been taken. To do this, we first have to verify that majority of the workers within the threshold suggested the same action, and second, the suggested action has the maximum Q-value in the recorded video's game state.

Figure 2(c) shows that all workers' suggested actions that are within the 30-move threshold, in both the real-time and review cases, meaning that a majority of workers do suggest similar actions.

Knowing that the crowd reaches consensus on a single action, we can now compare the crowd's advice to the recorded Q-values of the game to verify if it is the correct (near-optimal) action. The maximum Q-value of the 4 possible Pac-Man actions determines what action Pac-Man should perform. In Video 1, a step before 986 moves should suggest the Q-values for the next move. At move 985, the Q-values are: *up* $=$ 1729, *right* $=$ 1621, *down* $=$ 1768, and *left* $=$ 1621. In the human-controlled game in Video 1, Pac-Man went right at this time when it should have gone down (the maximum Q-value). And as shown in Figure 2(c), workers did suggest for Pac-Man should move downward in Video 1. Similar in the other three videos demonstrate that workers can identify that a mistake has been made but as well as provide an advice that is useful and near-optimal.

### Real-time vs. Review

We expected the real-time setting to be considerably harder than review setting. This assumption can be verified by considering the mean difference for each setting — the average mean difference for real-time setting is 9.1 moves ($\approx 0.36$ seconds) while review case is of 4.5 moves ($\approx 0.18$ seconds). The lower mean difference in review experiments shows that if additional time is available, even closer estimates of the point of the mistake can be gathered.

We performed a $4 \times 2$ Between Subjects Factorial ANOVA test of all Action Suggestion experiments shows that the difference of suggested mistake time by workers between subjects real-time and review setting was statistically significant ($F = 5.10, p < .05, \eta^2 = .023$). This difference between real-time and review setting in all Mistake Identification experiments is also significant ($F = 5.02, p < .05, \eta^2 = .022$). This indicates that the different mistakes in each video can also affect worker's ability to identify them.

Interestingly, there is only a small difference between the mean difference of real-time and review setting in Mistake Identification for Video 2. This indicates that the mistake in Video 2 was harder for workers to identify than the mistakes in the other three videos.

It is notable here that the average of mean differences in the real-time setting for Mistake Identification results to 9.8 moves ($\approx 0.39$ seconds), and with Action Suggestion at 8.8 moves ($\approx 0.35$ seconds), which are both very close to the human response for tasks with no high-level reasoning needed (e.g., clicking a button in response to a visual stimulus). This suggests that crowd advice for tasks, such as navigation, can be collected nearly as fast as people can physically respond. This quickly-available input can, in turn, be used to improve real-time learning of virtual and physical agents.

## FUTURE WORK

Future work will focus on developing learning algorithms that to leverage the unique strengths of human input on-the-fly without being detrimentally affected by incorrect advice. Although others [9] have incorporated advice from multiple demonstrators in past work, errors from crowdsourced workers are a unique challenges and opportunities to scale these systems. Furthermore, we plan to continue to improve our interfaces to further reduce the latency of worker responses. One potential method to do this is to leverage workers' ability to predict when mistakes might be made, which we initially observed, to collectively decrease latency below the best after-the-fact response speed possible. We are also interested in studying how the number of examples during the tutorial affects participants' accuracy. Finally, we are interested in eliciting a confidence measure from workers, potentially allowing us to weight different pieces of advice.

## CONCLUSION

Reinforcement learning algorithms often suffer from poor early-stage performance since agents have to experience considerable amount of trial-and-error before learning an effective policy. Our approach uses real-time crowds to provide immediate assistance to the learning agent to help improve its performance. We ran a set of user studies to show that crowd workers from Amazon Mechanical Turk can respond quickly and accurately enough to provide just-in-time feedback to an agent playing Pac-Man. We show that workers can correctly identify the point at which a mistake is made by Pac-Man and the optimal action Pac-Man should have executed. We also showed that higher performance could be achieved by workers in *post hoc* review settings.

Our results demonstrated that 1) crowd workers are able to accurately choose the mistake time in real-time with a mean latency of just 0.39s, and 2) latency does not increase if workers must also suggest an action. By leveraging the crowd, we present an effective, scalable means of providing during-task assistance to learning agents.

## ACKNOWLEDGEMENTS

The authors would like to thank our anonymous reviewers and Leah Zulas for helpful comments and suggestions. This work was funded in part by NSF IIS-1319412, and a Microsoft Research Ph.D. Fellowship.

## REFERENCES

1. Argall, B. D., Chernova, S., Veloso, M., and Browning, B. A survey of robot learning from demonstration. *Robot. Auton. Syst. 57*, 5 (May 2009), 469–483.

2. Goldberg, K., Chen, B., Solomon, R., Bui, S., Farzin, B., Heitler, J., Poon, D., and Smith, G. Collaborative teleoperation via the internet. In *Proc. of ICRA* (2000).

3. Knox, W. B., and Stone, P. Combining manual feedback with subsequent MDP reward signals for reinforcement learning. In *Proc. of AAMAS* (2010).

4. Lasecki, W. S., Murray, K. I., White, S., Miller, R. C., and Bigham, J. P. Real-time crowd control of existing interfaces. In *Proc. of UIST* (2011).

5. Lasecki, W. S., Song, Y. C., Kautz, H., and Bigham, J. P. Real-time crowd labeling for deployable activity recognition. In *Proc. of CSCW* (2013).

6. Loftin, R., Peng, B., MacGlashan, J., Littman, M. L., Taylor, M. E., Huang, J., and Roberts, D. L. A strategy-aware technique for learning behaviors from discrete human feedback. In *Proc. of AAAI* (2014).

7. Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction*, vol. 28. MIT press, 1998.

8. Taylor, M. E., Carboni, N., Fachantidis, A., Vlahavas, I., and Torrey, L. Reinforcement learning agents providing advice in complex video games. *Connection Science 26*, 1 (2014), 45–63.

9. Taylor, M. E., Suay, H. B., and Chernova, S. Integrating reinforcement learning with human demonstrations of varying ability. In *Proc. of AAMAS* (2011).

10. Toris, R., Kent, D., and Chernova, S. The robot management system: A framework for conducting human-robot interaction studies through crowdsourcing. *Journal of Human-Robot Interaction 3*, 2 (2014), 25–49.