Tasker: Safely Serving Verifiable Microtasks for Researchers

Monique Jeffrey[1,4], Imauri Motorin[1,4], Sophie Kozolan[1,4], Samuel Osei-Afriyie[2,4], Mark Whiting[3,4]
[1]University of Maryland Baltimore County (UMBC), Catonsville, USA
[2]SUNY College at Oneonta, Oneonta, USA
[3]Carnegie Mellon University, Pittsburgh, USA
[4]iSchool Inclusion Institute, Pittsburgh, USA

Abstract. Paid crowdsourcing removes many traditional boundaries in conducting participant based research, however with this new tool, new instrumentation challenges have arisen for researchers. Three common challenges include: the difficulty in creating large numbers of high quality and novel tasks, verifying results of the tasks without relying on manual cheat mitigation techniques, and ensuring that the tasks adhere to the latest visual and instructional design to get high quality results. These circumstances endanger current and future research on Amazon Mechanical Turk and can result in compromised data. We introduce Tasker, a secure system architecture for serving unique tasks supported by usability principles to workers, and providing verification information concerning their completion and accuracy to researchers. This poster discusses insights from our pilot study and explorations toward methods that demonstrate a marked improvement for speed, security and robustness in developing tasks for research leveraging Amazon Mechanical Turk.

Keywords: Crowdsourcing, Amazon Mechanical Turk, Research tools

Acknowledgements: Thank you to Mike Depew, Kayla Booth and the iSchool Inclusion Institute
Contact: Monique Jeffrey monijef1@aumbc.edu and/or Imauri Motorin bp70291@umbc.edu

1 Introduction

Amazon Mechanical Turk (MTurk), a popular crowdsourcing platform, allows requesters to outsource human intelligence tasks (HITs) [2] for pay. MTurk removes traditional boundaries to research such as time, cost, and access to participants by centrally combining a low-cost compensation system, a large subject pool, the collection of data [3], and effectively reducing the time between theory development and experimentation [16]. However, three issues undermine the use of MTurk by researchers. The difficulty of creating large numbers of high quality and novel tasks, verifying results of the tasks without

relying on manual cheat mitigation techniques and ensuring that the tasks adhere to the latest visual and instructional design standards to get high quality results.

Creating large numbers of high quality and novel tasks allow researchers to meet their specific data collection goals, but often researchers must "defensively design HITs" [6] or come up with an "alternative approach by designing HITs that are less attractive for cheaters" [10]. In an attempt to quickly make the most money, some workers submit "generic, non-reflected answers" [10], which along with spammers and bots [11] flood the marketplace and negatively affects the quality of data leading researchers to extensively verify result quality. "High quality gold standard data or inter-annotator agreement ratios" [10], are only two of the ways that researchers have to manually apply their own cheat mitigation techniques. This is not only time-and-resource-consuming but also risky due to the possibility of workers gaming the system and weakening the integrity of their research [10, 16]. For multiple HITs, such thorough manual verification is not always feasible [10]. Designing high quality tasks that are clear and easily understood takes some level of usability knowledge. MTurk task instructions can be unclear and the task itself ambiguous [9]. Daniel and Farhad [7] presented findings from a focus group involving 28 researchers that named "ease of use", meaning a user-friendly interface layout, as an agreed upon requirement of crowdsourcing platform design. The persistence of these interconnected problems displays the need for a solution with a balance of high quality and low expense.

We introduce Tasker, a system that delivers verifiable tasks from a secure database, avoiding nefarious workers, safeguarding the integrity of the researchers experiments and incorporating the best usability practices for high quality designs from current literature. This work explores this situation through the lense of three questions: 1) Could we make using MTurk easier for researchers with a task database? 2) Is it possible to mitigate cheating with verification? 3) Could we integrate task design best practices to improve performance?

2 Tasker

2.1 Pilot Study

The goal of the study was to understand the needs and potential benefits of Tasker for researchers who use crowdsourcing platforms. We sent out a survey to researchers who have previously used MTurk, and collected the responses two weeks after. We asked questions about their research background, the platform they usually use, the kind of tasks they perform on those platforms, and the time and resources they spend in order to get a feel of their environment.

According to our results, most of our participants have published at venues like the ACM Computer Supported Cooperative Work (CSCW) conference and The ACM Conference on Human Factors in Computing Systems (CHI). Some of their research topics include crowdsourcing and the use of platforms

like MTurk. For most of our participants, building a task is like creating a set of requirements for a software. Some of their challenges are making the requirements clear enough for someone to understand what is needed for them to do. Some of the researchers agree that they would like to have templates for common tasks and will appreciate having some feedback from the system.

Overall, the researchers concerns were similar to the problems we found in the literature. The system would need to be easy to use reducing multiple steps and ambiguity for researchers, safe for researchers because many have sensitive data, and adhere to proven usability designs so that the researchers will avoid laborious design considerations when generating tasks. We took all these aspects into consideration in order to build a system that will be useful for researchers using MTurk.

2.2 System Design

We present Tasker: a service accessed through an API or application programming interface that is aimed at improving the quality of worker responses by acting as a verification system for researchers. The system helps determine the validity of worker responses to HITs by providing pre-developed, advanced engineered tasks based on the researcher's request. The infrastructure for Tasker enables researchers to engage in rapid crowd research on-boarding of research participants (workers). Predeveloped HIT and rubrics are used as assessments to validate workers. Validity of worker responses to those tasks are determined by a rubric unique to each task type. After the tasks are distributed to workers, their responses are scored against a rubric and results are sent to the researcher upon completion of the tasks.

The system provides: tasks that follow the best practices in task design, task security through verification, and a robust database of multiple task types held in a secure server. The database of tasks were designed through analyzing tasks on the MTurk website and both "cognitively inspired features" [10] and "motivation aspects for human computation" [11] from 5 highly cited papers[1]. The researcher can render tasks from the database through an API call. The worker interface for each tasks front-end component is modeled after the best practices in task design and a compilation of HTML and CSS components.

When the desired task types are selected by the researcher, source code is generated. The researcher then embeds code into an MTurk HIT request to be issued to workers. A task from the Tasker database is then rendered in place of the embed code. To ensure a task is not duplicated, a security verification ledger based on the MTurk worker ID is used. After tasks are distributed and workers response to the task, their responses are scored against a unique rubric based on the task type. Depending on the task type selected by the researcher one of two different validation methods or types of rubrics are used to score worker's validity: ground truth validation and open-ended validation. With the ground truth validation method, the scoring of the task is a simple 'valid' or 'not valid' response. For the open-ended validation, a score is given based off

[1] See citations [13], [14], [15], [16], and [17]

rubric criteria. Upon completion of the task by workers, a 'validation score' is sent to the researcher through a private response with the corresponding MTurk 'worker ID' from the worker who completed those specific tasks.

3 Evaluating Tasker

We will study MTurk workers and researchers experience with Tasker through two surveys in the near future. Survey 1 will consist of workers rating task quality while using Tasker and without Tasker. Survey 2 will measure the researchers ease of use and will include questions that will help us compare alternative methods for the mitigation of cheating and verification of data. This will allow us to establish a baseline for Tasker for both the worker and researcher while also learning if Tasker meets its goal in implementation. After analyzing our results from the two studies, we hope to dive deeper into usability principles by conducting user experience testing; as we have currently only used practices based off current literature. User experience testing will also help us evaluate workers and how well our verification system mitigates cheating. Furthermore, studying how to incorporate machine learning into Tasker for task creation and task verification will be an ongoing priority.

4 Conclusion

To fully answer the three guiding questions, (1) Could we make using MTurk easier for researchers with a task database? (2) Is it possible to mitigate cheating with verification? (3) Could we integrate task design best practices to improve performance? We must continue our research. We see the potential through templates, a robust database of tasks, a secure verification system and usability principles and will continue to explore these methods. Moreover, we envision the integration of other features like machine learning to offer more robust support. Potentially, Tasker will not just support researchers on MTurk but all participants of the MTurk community and across other crowdsourcing domains.

References

[1] Association for Computational Linguistics. Meeting (45th : 2007 : Prague, C.R. and Association for Computational Linguistics., C. 2007. *ACL 2007 : proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. Association for Computational Linguistics.

[2] Behrend, T.S., Sharek, D.J., Meade, A.W. and Wiebe, E.N. 2011. The viability of crowdsourcing for survey research. *Behavior research methods*. 43, October 2015 (2011), 800–813. DOI:https://doi.org/10.3758/s13428-011-0081-0.

[3] Buhrmester, M., Kwang, T. and Gosling, S.D. 2011. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*. 6, 1 (2011), 3–5. DOI:https://doi.org/10.1177/1745691610393980.

[4] Cao, C.C., She, J., Tong, Y. and Chen, L. 2012. Whom to Ask? Jury Selection for Decision Making Tasks on Micro-blog Services. *VLDB*. 5, 11 (2012), 1495–1506.

[5] Chandler, D. and Kapelner, A. 2013. Breaking monotony with meaning: Motivation in crowdsourcing markets. *Journal of Economic Behavior and Organization*. 90, (2013), 123–133. DOI:https://doi.org/10.1016/j.jebo.2013.03.003.

[6] Chen, J.J., Menezes, N.J. and Bradley, A.D. 2011. Opportunities for Crowdsourcing Research on Amazon Mechanical Turk. *Human Factors*. 5, (2011), 3. DOI:https://doi.org/10.1145/1357054.1357127.

[7] Daniel, S. and Farhad, D. 2014. User Requirements of a Crowdsourcing Platform for Researchers: Findings From a Series of Focus Groups. *PACIS 2014 Proceedings*. (2014).

[8] Dow, S., Kulkarni, A., Klemmer, S. and Hartmann, B. 2012. Shepherding the Crowd Yields Better Work. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*. (2012), 1013–1022. DOI:https://doi.org/10.1145/2145204.2145355.

[9] Dredze, M. and Callison-Burch, C. 2010. Creating Speech and Language Data With Amazon's Mechanical Turk. June (2010), 1–12.

[10] Eickhoff, C. and de Vries, A.P. 2013. Increasing cheat robustness of crowdsourcing tasks. *Information Retrieval*. 16, 2 (2013), 121–137. DOI:https://doi.org/10.1007/s10791-011-9181-9.

[11] Huang, S. and Fu, W. 2013. Don't Hide in the Crowd! Increasing Social Transparency Between Peer Workers Improves Crowdsourcing Outcomes. *CHI 2013: Changing Perspective, Paris, France*. (2013), 621–630. DOI:https://doi.org/10.1145/2470654.2470743.

[12] Kulkarni, A., Can, M. and Hartmann, B. 2011. Turkomatic : Automatic Recursive Task and Workflow Design for Mechanical Turk. *Aaai*. (2011), 2053–2058. DOI:https://doi.org/10.1145/1979742.1979865.

[13] Marcus, A. and Parameswaran, A. 2013. Crowdsourced Data Management: Industry and Academic Perspectives. *Foundations and Trends R in Databases*. 6, 12 (2013), 1–161. DOI:https://doi.org/10.1561/1900000044.

[14] Marcus, A., Wu, E., Karger, D.R., Madden, S. and Miller, R.C. 2013. Crowdsourced Databases : Query Processing with People. (2013).

[15] Marge, M.R., Banerjee, S. and Rudnicky, A.I. 2010. Using the Amazon Mechanical Turk for Transcription of Spoken Language. (2010).

[16] Mason, W. and Suri, S. 2012. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*. 44, 1 (2012), 1–23. DOI:https://doi.org/10.3758/s13428-011-0124-6.

[17] Moussawi, S. and Koufaris, M. 2013. The Crowd on the Assembly Line: Designing Tasks for a Better Crowdsourcing Experience. *Proceedings of the 34th International Conference on Information Systems*. (2013), 1–17.