Human Computation and Crowdsourcing: Works in Progress and Demonstration Abstracts
AAAI Technical Report CR-13-01

Frenzy: A Platform for Friendsourcing

Lydia B. Chilton, Felicia Cordeiro, Daniel S. Weld, James A. Landay
University of Washington
{hmslydia, felicia0,weld,landay}@cs.uw.edu

Abstract

Although micro-task crowdwork was popularized by the Mechanical Turk (MTurk) labor-market platform, it is useful in many other contexts and communities as well. Unfortunately, several MTurk design choices, such as worker anonymity and isolation, are problematic in other environments. This paper introduces Frenzy, a platform for friendsourcing. Frenzy helps people who are closely connected in a community complete crowdsourcing tasks on a platform that affords more user control and freedom than traditional crowdsourcing platforms. Our evaluation of Frenzy shows that friendworkers can find meaningful work to do, effectively use discussion to reduce uncertainty, and finish the task. Most importantly, they are happy with the quality of the output and indicate a desire to use Frenzy for their own tasks such as: creating wedding albums, coding social science data, and creating sessions of accepted conference papers.

Introduction

Crowdsourcing has been used to solve a diverse set of problems too hard for computers to perform. In particular, workflows comprised of micro-jobs allow people to contribute to a large task without making a big time commitment. While micro-work has become popular because of the success of the Amazon Mechanical Turk (MTurk) labor market [1], the idea has merit for other communities (e.g., friends and coworkers) who are motivated by other incentives (e.g., a sense of community or desire for fun rather than economic payment). We term this type of distributed labor "friendsourcing" and call those performing labor "friendworkers."

Unfortunately, several MTurk design choices, such as worker anonymity and isolation, are problematic in friendsourcing environments. Furthermore, in the process of translating a workflow and interface from MTurk workers (turkers) to friendworkers is surprisingly challenging. Friends are different than paid workers and one must design for them differently. Specifically, we note three key differences between turkers and friendworkers.

First, turkers get paid to use whatever interface you make for them. Friends, however, want interfaces that afford user control and freedom.

Second, often tasks that we ask the crowd to do are subjective. Workers don't know that they are doing it correctly. On Turk, workers get the feedback of payment. If they are paid, then they assume they are doing the tasks well enough to continue doing them. Friends, however, do not get paid and thus need some other mechanism to let them know they are doing tasks well and making valuable contributions.

Third, for turkers, we construct short tasks that can be done massively in parallel. However, we don't have enough friends to merit massively parallel work. Friends still benefit from short tasks so they are not overwhelmed by hairy problems. However, making tasks massively parallel no longer the primary goal.

Friendsourcing Challenges

Designing a platform for friendworkers is a challenge for the field of human computation. Friendworkers want a platform that is more open and has more user control and freedom and communication. But there are dangers of designing for those desires:

1. If workers have complete freedom to do what they want, then there is no guarantee that all needed tasks will complete. How can friendworkers be encouraged to do the necessary jobs?
2. Friendworkers must be able to easily find work they enjoy. They shouldn't have to spend their time hunting for items that need attention.
3. If workers are allowed to chat, we must make sure the discussion is productive and does not thwart progress.

Frenzy Design

In response to this challenge we are building Frenzy, a platform for friendsourcing that draws design inspiration from existing online social systems such as Twitter [5], Facebook [2], Google Spreadsheets [3] and MediaWiki [4] and is capable to enabling a number of typical crowdsourcing tasks: both independent tasks and tasks that result in the global understanding of all the data items. We now discuss how features of Frenzy address each of the challenges described above.

Displaying tasks On most crowdsourcing platforms, workers see only one task at a time: one image to caption, one paragraph to shorten, etc. HITs sometimes batch a few tasks together, but they never let workers see all the tasks because multiple people might work on them at the same time. In contrast, Frenzy displays a feed of tasks. The data item for each task is the subject of the feed, and the workers answer will go in the discussion section to the right. This way, friendworkers can easily search or scroll through all the tasks. Two mechanisms prevent workers from duplicating each other's work. First, Frenzy gives strong visual cues as to the presence of other workers, a technique inspired by the way collaborative spreadsheets use colored cursors to represent other users. Second, Frenzy shows which items have been recently updated, and workers can quickly see the most recent contributions.

Finding work Frenzy allows friendworkers to find tasks though a mechanism we call *actionable feedback*, which provides feedback on how close you are to satisfying the system goals and gives you clickable items that allow one to search and sort the tasks. The owner/administrator provides completion conditions. For example, she can require that all images have at least one hashtag, or all paragraphs are 15% shorter, etc. Frenzy computes statistics and its feedback gives workers affordances to select tasks by their completion status.

Resolving uncertainty User agreement can be a powerful sign to workers that their answer is correct. While communication is problematic in systems like MTurk and the ESP game, since it allows cheating, it helps and motivates friendworkers. Frenzy implements two communication channels: a real-time chat client, a task-embedded discussion.

References

1. von Ahn, L. and Dabbish, L. Labeling images with a computer game. *CHI 2004*.

2. Amazon Mechanical Turk. https://www.mturk.com/mturk/.

3. The Berkeley Segmentation Dataset and Benchmark. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/.

4. Bernstein, M., Little, G., Miller, R.C., Hartmann, B., Ackerman, M.S., Karger, D.R., Crowell, D., and Panovich, K. Soylent: a word processor with a crowd inside. *UIST 2010*.

5. Chilton, L., Little, G., Edge, D., Weld, D.S., and Landay, J.A. Cascade: Crowdsourcing Taxonomy Creation. *CHI 2013*.

6. Dai, P., Mausam, and Weld, D. Decision-theoretic control of crowd-sourced workflows. *AAAI 2010*.

7. Dow, S., Kulkarni, A., Klemmer, S.R., and Hartmann, B. Shepherding the crowd yields better work. *CSCW 2012*.

8. Facebook. Facebook. http://www.facebook.com.

9. Fan, S.B., Robison, T., and Tanimoto, S.L. CoSolve: A system for engaging users in computer-supported collaborative problem solving. *VLHCC 2012*.

10. Galaxy Zoo. http://www.galaxyzoo.org/.

11. Google Spreadsheets. http://www.google.com/drive/apps.html.

12. Heimerl, K., Gawalt, B., Chen, K., Parikh, T.S., and Harmann, B. Communitysourcing : Engaging Local Crowds to Perform Expert Work Via Physical Kiosks. *CHI 2012*.

13. Kittur, A., Smus, B., and Kraut, R.E. Crowdforge: Crowdsourcing complex work. *UIST 2011*.

14. Kriplean, T., Morgan, J., Freelon, D., Borning, A., and Bennett, L. Supporting reflective public thought with considerit. *CSCW 2012*.

15. Kulkarni, A., Can, M., and Hartmann, B. Collaboratively crowdsourcing workflows with turkomatic. *CSCW 2012*.

16. Little, G., Chilton, L., Goldman, M., and Miller, R.C. Turkit: human computation algorithms on mechanical turk. *UIST 2010*.

17. MediaWiki. http://www.mediawiki.org/wiki/MediaWiki.

18. MobileWorks. https://www.mobileworks.com/.

19. Starbird, K. and Stamberger, J. Tweak the tweet: Leveraging microblogging proliferation with a prescriptive syntax to support citizen reporting. *ISCRAM 2010*.

20. Twitter. http://www.twitter.com/.

21. Ushahidi. Ushahidi. http://www.ushahidi.com/.

22. Zhang, H., André, P., Chilton, L., Kim, J., Dow, S.P., Miller, R.C., Mackay, W., and Beaudouin-Lafon, M. Cobi: communitysourcing large-scale conference scheduling. *CHI Extended Abstracts 2013*.

23. Zhang, H., Horvitz, E., Miller, R.C., and Parkes, D.C. Crowdsourcing General Computation. *CHI 2011*.

24. Zhang, H., Law, E., Miller, R.C., Gajos, K.Z., Parkes, D.C., and Horvitz, E. Human computation tasks with global constraints. *CHI 2012*. Amazon Mechanical Turk. https://www.mturk.com/mturk/.

25. Michael S. Bernstein, Desney S. Tan, Greg Smith, Mary Czerwinski, Eric Horvitz: Personalization via friendsourcing. ACM Trans. Comput.-Hum. Interact. 17(2) (2010)