Building A Brighter Future for Crowd Work

Skill ladders may help crowd workers to "skill up" as they work. But what other technical innovations will lead to better opportunities for crowd work?

By Jeffrey P. Bigham and Kristin Williams

DOI: 10.1145/3013522

Work is changing. Crowd work, with its open calls, short contracts, and non-existent support for worker development, is proliferating. Work won't look exactly like today's Amazon Mechanical Turk in the future, but elements of it (often the least desirable) seem likely to influence the future of work if we don't figure out how to stop it.

There are many angles one could take on this problem. For instance, an on-going concern about crowd platforms is they are low-paying, so we could create better policy to protect workers. Workers generally have little power on today's platforms; we could facilitate worker organization to better enable them to protect themselves. These are valuable and necessary endeavors.

This article is about what systems builders can do to create a brighter future for crowd work.

It is tempting to believe what we need is to design a new platform from scratch that represents our new future—this may in fact be one route to a better future—but building new platforms is risky. It takes effort to build the platform, attract workers there, and in the end you might still end up with a platform that you later realize misses some core component of what you need.

We are less interested in the crowd platform than what we build on top of it. Many crowd platforms can be seen as being "crowd complete," a loose play off of "Turing complete," meant to convey the idea that given a certain set of functionality, any crowd platform could be used to create any other.

Many of the tools we have built with our collaborators use Amazon Mechanical Turk, Upwork, and other platforms, in order to take advantage of the built-in capabilities of each. Amazon Mechanical Turk makes large numbers of workers available via an API, whereas Upwork offers workers with advertised (and rated) expertise in different professional areas (e.g., programming, or visual design). But, platform affordances don't dictate how they can be used. Sometimes we find experts on Mechanical Turk (for instance, Turkers speak hundreds of languages [1]), and sometimes we use Upwork without the customary personal contact with workers via an API [2]).

Like many, we have taken inspiration from "The Future of Crowd Work" by Kittur et al [3]. In particular, we have adopted their idea of creating "crowd career ladders" and supporting worker learning into our "skill lad-

Image by Lienhard.Illustrator / Shuuterstock.com

der" concept, which involves significant building innovation and effort to make practical. The remainder of this article introduces the concept of a skill ladder, then makes it concrete with a skill ladder we have created to teach real-time captioning, and finally ends with a call for more systems builders to get involved with creating a brighter future for crowd work.

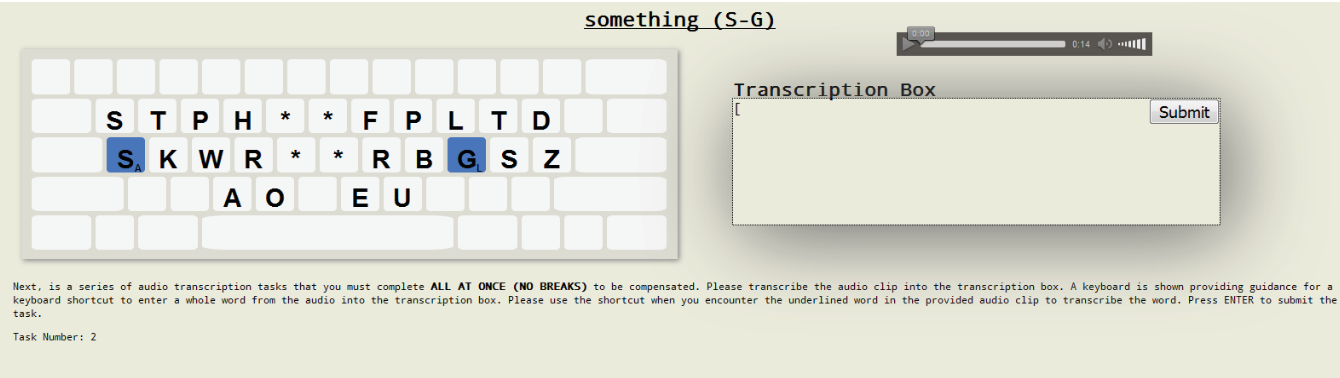
BUILDING LADDERS TO NEW SKILLS

Crowd-working platforms are often characterized as being marketplaces for "low skill" or "non-expert" work. One issue is that while workers have all kinds of expertise, that expertise is difficult to tap into. Another challenge is crowd work on existing platforms does not support workers who would like to "skill up" (gain expertise that would qualify them for "better" work) while working. We use the term "better" to mean a variety of things, but think of it as a proxy for improved pay, enjoyment, etc.; the things people might want to have in their work.

Many workers cannot simply take time off from working to learn new skills. Some workers rely on the income from crowd work to get by. The schedules of traditional education may not fit their working schedule. (One of the reasons why they might have been attracted to the piecemeal jobs on crowd platforms to begin with!) As a result, it is important to consider building opportunities for improving one's skills into the tasks that workers are already doing.

We refer to this building-in of opportunities for improved work as a "ladder." A skill ladder is a path to move up in the skill chain. Skill ladders are embodied in the socio-tech-

Figure 1. While completing a transcription task, Scopist suggests chords to workers to help them complete tasks faster and learn stenography.



nical processes that allow workers to develop and improve. This is not a new concept. Traditional companies often provide (and even require) skill ladders for employees to go through. For instance, technology companies routinely place fresh undergraduate hires on skill or career ladders in their organizations that will teach them what they need to know about the infrastructure and culture of the organization in order to qualify them for higher levels of responsibility at the organization. Many organizations even see it as a requirement to work various positions in the company before qualifying for higher-level roles. It would be odd for a research manager to never have been a researcher, or for someone to be a professor without ever having been a student.

Ladders are currently absent from crowd work. But, ladders could be facilitated by system or task design. They could potentially exist between many different job types (e.g., editing to writing, data labeler to data producer, etc.), although some may be better candidates than others. It may be easier to move between work that is similar in type than it would be to transition from work of different types. For instance, it may make sense to ladder-up from transcribing short audio files to more challenging transcription tasks such as transcribing longer recordings, or those with specialized vocabulary, or those with more stringent real-time demands.

A good ladder has a few different qualities. One is that a ladder allows workers to transition. Good ladders for a particular platform work on tasks that exist on their target platform. A great ladder works on tasks that are common on that platform. A good ladder does not require the worker to spend more time per task than they otherwise would (thus, reducing their pay rate). A great ladder makes the worker incrementally better at the task as they learn to take on new tasks at a different level (thus incrementally improving their pay as they go).

We have created a concrete example of a (potentially) great ladder that transitions workers from audio transcription tasks to real-time captioning. Audio transcription is one of the most common types of tasks on many different crowd platforms, including Mechanical Turk. It involves the conversion of aural speech to text. Computers can't yet do this reliably, and so people do the bulk of this work. Almost anyone can do this (slowly), and pay generally averages from $2-5 per hour (USD).

A parallel task requiring higher skill (and demanding higher wages,

Figure 2. Example of the transcriptionist's QWETY input and Scopist prediciting the typist's intentions.

Input Prompt

```
miracles now drnI use opterm miracle I don't
```

Key Sequence

```
mimriraaccllees s_nonwow _drnnrdii _uussee _oppottererrmm _mimirraaccllee  ii  dodonntt
uu u  u u u u qq  uu q  q uuu   q q u u q q uu  u uu  q q uu  u u u u q q q q uu  u u
```

Possible Chords

```
mimriraaccllees s_nonwow _drnnrdii _uussee _oppottererrmm _mimirraaccllee  ii  dodonntt
```

Hypotheses

0.75 miracles now drni use opterm miracle I don't
0.89 miracles now drni use the term miracle I don't
0.89 miracles now when I use opterm miracle I don't
1.00 **miracles now when I use the term miracle I don't**

on the order of $100-300 per hour (USD) is real-time captioning. Real-time captioning also involves the conversion of speech to text, but it adds an additional brutal constraint. The conversion has to happen within five seconds of the speech. The reason it needs to be so fast is that it's used for live performances (lectures, live sports, etc.), and any longer than five seconds of delay makes it too difficult for people who rely on the captioning to follow along. Currently, real-time captioning can only be done by professional stenographers who train for a year or two to be able to type at real-time speeds. They do this by learning to type "chords" involving multiple simultaneous key-presses, which correspond to parts of words or even whole words. The best real-time captionists can type at a sustained pace of more than 300 words per minute.

We have built a system called Scopist. It is a ladder for workers doing regular audio transcription work to gradually "skill up" to real-time captioning. A primary reason this was difficult to do before was real-time stenography had previously assumed stenographers would train for a period of time (generally a year or two) until they had completely skilled up to real-time stenography. That is, they had to master everything before the skill was useful or usable. Instead we created a JavaScript application that allows workers to type QWERTY and steno into the same textbox, allowing them to type chords as they learn them. This was non-trivial because often chords appear to be QWERTY and vice versa, e.g., the two keys "af" could mean the first two letters of the word "affinity" or the entire chord for the word "something."

Solving this problem involved creating an algorithm that observes the typing sequence, and makes a good guess as to the typist's likely intention—for those cases in which it wasn't clear, was it more likely that the typist was typing a chord or meant to type individual letters? We leveraged differences between the two methods of typing, i.e., in steno one never uses spaces because they are automatically determined. We settle ties with a language model. Overall, this algorithm is able to differentiate between QWERTY and steno with nearly perfect accuracy.

Over time, we plan to deploy this system to workers on Mechanical Turk, and maybe we'll see some are able to eventually "graduate" into real-time captioning. But this is, of course, only one skill ladder. We chose it because it seems especially well-suited for the kind of work that is being done on Mechanical Turk now. It provides a parallel feel between two types of jobs, and workers can improve incrementally over time and benefit right away from increased efficiency.

It's interesting to think about what other kinds of ladders might make sense to build on this or other platforms. Could workers start on a path to becoming a medical doctor by labeling images of cancer growths? Could they start on their path toward becoming a journalist by writing or proofreading blog posts? These are big questions, but part of the answer is technical HCI. Can we build the systems that would facilitate such skilling-up over time?

A CALL TO BUILDERS

Creating a brighter future for crowd workers will require contributions from diverse fields. We need platforms for workers to come together, we need to engage more diverse platforms, and we need policy advances to make sure labor laws keep up with changes to work. Builders will be vital for this new future because a lot of what we would like to achieve with policy, or even socially minded innovation in this space, will not be possible without innovation.

This is the example Scopist was meant to provide. It's great to talk about creating ladders that allow workers to advance as they work, but it's not always clear how to do that. In the case of Scopist, it involved creating an algorithm that could differentiate QWERTY text input from chorded text input. Once that technical innovation was there, we could create a textbox that could accept both, allowing workers to gradually learn to chord as they do their normal work. This technical innovation allows us to start thinking about how to scaffold learning into the tasks. That may involve more technical innovation as we attempt to model what is best to teach when, and draw more and more on speech recognition to recommend chords to learn based on what is likely to be useful.

If we look at "The Future of Crowd Work" [3], we see a lot of other opportunities for technical innovation. What does reputation look like on a platform like Mechanical Turk? Can we automatically discover expertise, or associate good work on third party tasks to the correct kind of skills? To build expertise and reputation, workers will likely want to specialize. What systems do we need to allow them to better specialize? It seems that at a minimum we need to build better support for workers finding tasks in their area of specialization. There may even be opportunities for workers to specialize in finding work for others. What does a management role look like on crowd platforms, and what will be needed to support it? In our opinion, we've barely started to scratch the surface of what technical innovation needs to happen to support this brighter future many of us our hoping for.

References

[1] Bloodgood, M. and Callison-Burch, C. Using Mechanical Turk to build machine translation evaluation sets. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk* (Los Angeles, June, 2010). Association for Computational Linguistics, 208-211.

[2] Huang, T.H.K., Lasecki, W.S., and Bigham, J.P. Guardian: A crowd-powered spoken dialog system for web APIs. In *Third AAAI conference on human computation and crowdsourcing* (San Diego, November 8-11).

[3] Kittur, N., Nickerson, J.V., Bernstein, M., Gerber, E., Shaw, A., Zimmerman, J., Lease, M., and Horton, J. The future of crowd work. In *Proceedings of the 2013 conference on computer supported cooperative work* (San Antonio, Feb. 23-27). ACM, New York, 2013,1 301-1318.

Biographies

Jeffrey P. Bigham is an associate professor in the School of Computer Science at Carnegie Mellon University. When he's not tweeting, he's building systems that combine crowdsourcing and AI to improve accessibility and reimagine work.

Kristin Williams is a Ph.D. student in the School of Computer Science at Carnegie Mellon University. By building interaction techniques, she creates new ways to functionally communicate in contexts demanding a high degree of linguistic skill.

1528-4972/16/12 $15.00