MobileWorks: Designing for Quality in a Managed Crowdsourcing Architecture

Anand Kulkarni[1,2], Philipp Gutheim[1,3,4], Prayag Narula[1,4], David Rolnitzky[1], Tapan Parikh[4], Björn Hartmann[3]
[1]MobileWorks, Inc., University of California, Berkeley
[2]Department of IEOR, [3]Division of Computer Science, [4]School of Information

ABSTRACT
Online labor marketplaces offer the potential to automate a variety of tasks too difficult for computers, but present requesters with significant difficulties in obtaining accurate results. We share experiences from building MobileWorks, a crowd platform that departs from the marketplace model to provide robust, high-quality results. Three architectural contributions yield measurably improved accuracy on input tasks. A *dynamic work routing system* identifies expertise in the crowd and ensures that all work posted into the system is completed with bounded completion times and at fair worker prices. A *peer management* system ensures that experienced members of the crowd prevent wrong answers. Last, *social interaction techniques* give the best workers the ability and incentives to manage, teach and supervise other members of the crowd, as well as to clarify tasks. This process allows the crowd to collaboratively learn how to solve unfamiliar tasks.

Author Keywords
crowdsourcing, human computation, development, marketplace

INTRODUCTION
Human computation platforms are online services that allow programmatic access to a large crowd of human workers on demand. Software systems that leverage human computation have become increasingly popular in recent years, and new applications are discovered on an ongoing

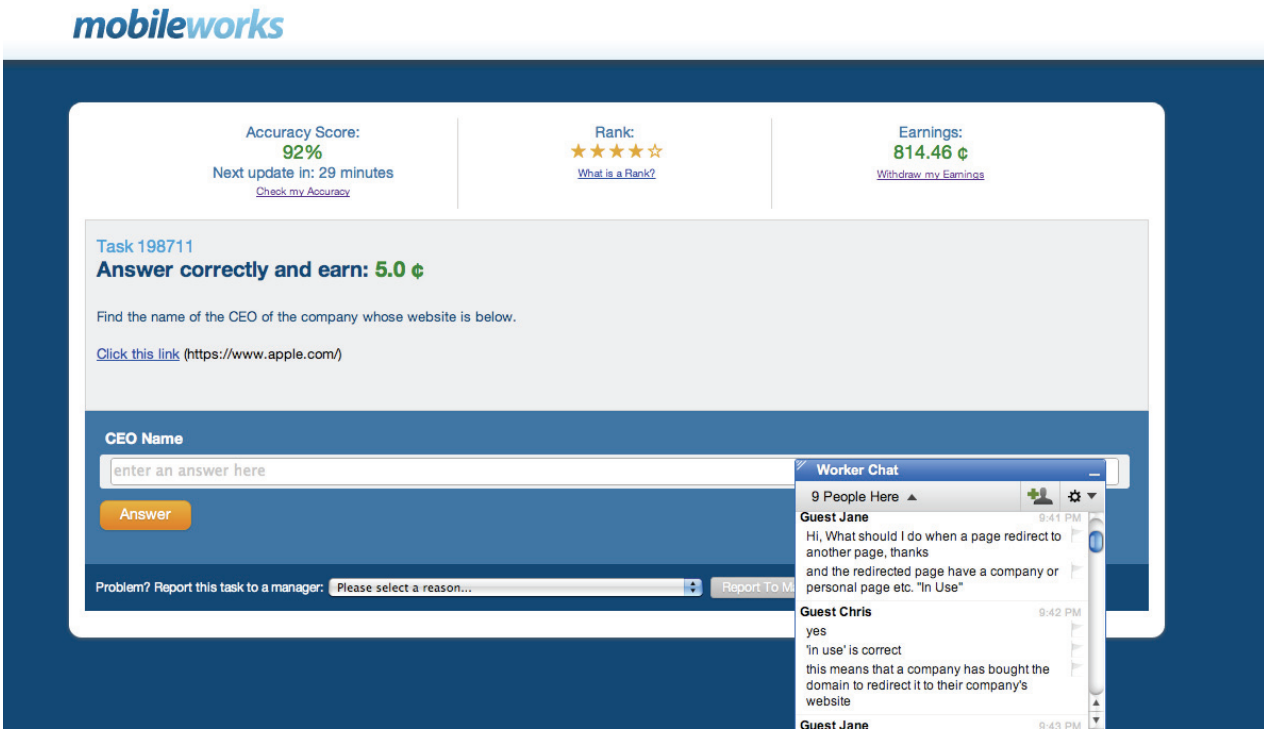

Figure 1. The MobileWorks interface serves a stream of appropriate tasks to workers, interleaving qualification tests, tasks and training material. Workers are presented with immediate feedback on performance and earnings. Worker-to-worker chat is used to debug tasks on the fly, and workers can escalate tasks for managerial review.

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

basis in problem domains ranging from vision and artificial intelligence to software development and business process outsourcing. Because human computation platforms involve algorithmic control of work for hire, they provide an interesting set of challenges at the intersection of economics, HCI, and theoretical computer science.

Human computation systems are typically assessed on their ability to provide accurate results. *Accuracy* refers to the correct completion of tasks posted by a requester, according to explicit or implicit criteria defined by the requester. Achieving high accuracy is a central challenge for all human computation systems. We classify several reasons why systems may provide inaccurate results:

1) Online workers may not want to do what they are being asked to do in exchange for the rewards offered, due to lack of motivation or malice. This is the *incentive* problem.

2) Even when workers are properly motivated to do a task correctly, the task may be ambiguous or unclear. This is the *task specification* problem.

3) Workers may make errors despite wanting to carry out a properly specified task. This is the *human error* problem.

The first two challenges are intertwined: when workers do not understand a task due to ambiguous design, they are likely to lose motivation.

Other problems are due to the market structure of human computation platforms. For instance, many tasks posted to online marketplaces languish and are never completed because workers view their rewards as inadequate or cannot find the tasks in a marketplace, a problem termed *starvation*. In aggregate, these problems complicate efforts to incorporate human computation services into software, since it becomes difficult to provide bounded task response times and accuracy guarantees.

The most well-studied commercial crowdsourcing platform today, Amazon's Mechanical Turk, operates as a web-based marketplace where employers can post groups of tasks to be solved and workers can browse these tasks and choose to answer them. The problems of designing effective tasks, filtering unqualified workers, and eliminating incorrect answers are largely left to employers; only rudimentary support such as filtering workers based on past accuracy is available. Most employers using Amazon Mechanical Turk must build extensive quality control infrastructure and employ domain-specific techniques to deal with possible errors.

MobileWorks is an alternative crowdsourcing platform designed to prevent the accuracy and speed deficiencies faced by employers in online labor marketplaces. A central difference is that MobileWorks is *not* a marketplace: it operates as an algorithmically managed service, routing work to qualified workers and recruiting additional participants as needed. Tasks presented to the system are automatically matched and presented to qualified workers identified through a combination of human and programmatic testing. Discrepancies are resolved by the best workers (managers) who play an active role in maintaining the quality of the system and managing other workers. Worker-to-worker interaction, led by managers, permits additional worker training and discussion of individual tasks. These mechanisms enable MobileWorks to address the same class of tasks solved on conventional labor marketplaces, while providing substantially higher accuracy and shielding employers from the burdens of quality control.

MobileWorks operates with a social mission to provide employment to marginalized populations in the developing world; as such, the majority of its workers are drawn from low-income populations in South and Southeast Asia and paid a fair baseline wage. We discuss novel aspects of MobileWorks' architecture in several areas, including task routing, pricing, peer management, and worker-to-worker interaction. Following this discussion, we present a brief analysis of the 500,000 results produced in the lifetime of MobileWorks examining and examine how effective these techniques were at maintaining quality in a real-world commercial tasks.

RELATED WORK

A substantial body of literature focuses on how to reduce worker error during human computation. In early work in human computation, the ESP Game [1] used redundancy to control error, finding that sending the same microtask to multiple workers could be effective in mitigating the effects of error and malice. Redundancy works well in mitigating human error with random structure, but is vulnerable to structural confusion in tasks and to worker collusion. Alternatives for maintaining quality include voting systems and iterative improvement [2]. Not all human computation platforms have taken the form of labor marketplaces. Alternatives include online games such as GWAP for labeling [3] and fold.it for protein-folding [4], as well as content-distribution networks that embed crowdsourcing tasks into other web activities such as CAPTCHAs [5]. While incentives are different, these systems face many of the same challenges in maintaining quality as marketplaces.

Other approaches, such as in those used in the commercial Crowdflower platform, combine redundancy with tracking of a worker's historical performance and ongoing worker assessment on so-called "gold standard" tasks with known answers [6]. Historical performance is readily gamed in open marketplaces and may not adequately predict worker performance on unfamiliar tasks [7]; gold standard tasks are unavailable for many kinds of work like content creation.

More sophisticated approaches use *workflows* that divide complex tasks such as essay-writing and editing into various short-duration tasks that are distributed among different workers. Soylent introduced the *find-fix-verify* workflow for complex text editing, supported by a peer review step that let workers verify that another worker's

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

edits were carried out correctly [8]. Turkomatic introduced the price-divide-solve meta-workflow that uses workers to assist in designing workflows for complex tasks [9]. Soylent identified important gradients in quality and behavior among worker populations, and Turkomatic found that improper allocation of workers to tasks within a workflow could reduce quality of results. These workflows are generally agnostic to the type of worker, leaving open the possibility that matching tasks to specific types of worker can yield enhanced accuracy.

Shepherd found that worker-to-worker feedback and requester-to-worker feedback displayed immediately after a task was useful in improving the quality of work on crowd platforms [10]; however, it did not investigate the impact of real-time worker interaction.

Several recent papers have explored mechanisms to reduce the latency of crowd algorithms, including combating task starvation and obtaining real-time results. VizWiz showed that streaming tasks to workers could maintain interest until relevant tasks become available [11], and found that using SEO-style techniques could enhance responsiveness of workers. Adrenaline showed that it is possible to pull results from crowd workers in seconds by keeping them "on call" until needed [12].

The use of crowdsourcing with populations in the developing world has also been explored before. TxtEagle, deployed in Kenya, used SMS text messages to provide tasks like audio transcription, local language translation and market research [13], but was limited by the capabilities of SMS to deliver more sophisticated tasks. SamaSource also seeks to employ marginalized populations with microwork, but manually optimizes outsourcing processes and partners with on-the-ground outsourcing agencies [14]. A 2010 study found that Mechanical Turk was largely unsuitable for workers at the bottom of the pyramid to find employment without additional modification [15] and caused additional issues of fairness [16, 17], even as other forms of online work such as e-lancing presented a rising opportunity for the developing world [18 19, 20].

THE MOBILEWORKS ARCHITECTURE

MobileWorks is based on a *task routing system* that assigns tasks from a priority queue to individual workers. Multiple assignments per task are issued for quality assurance. As part of this system, we have implemented techniques for identifying appropriate expertise within the crowd, for automatically pricing tasks, and for escalating errors or difficult instances to a special population of managers. Managers have high accuracy and proficiency, and we use them for worker recruitment, to evaluate potential problems with requester-defined tasks, and to resolve discrepancies in tasks.

Task routing and the dynamic work queue

Tasks are posted to MobileWorks via a REST API or through a web dashboard. At posting, requesters specify a set of instructions, a set of answer fields, preferences for what kind of workflow they require, and optionally, a set of skills. The set of skills is chosen from a list of keywords; alternately, requesters can define custom skills by providing gold standard tasks. Work pushed into MobileWorks is inserted into a priority queue of tasks that are processed by workers in turn.

When a worker arrives at MobileWorks via the web or a web-enabled mobile phone, he is assigned the next novel task in the queue for which he meets the required skills. Workers whose overall accuracy is below a certain level are reassigned to training tasks until their accuracy improves.

Workers are assigned tasks, rather than selecting them from a list. We hypothesize that this interface enables workers to perform work more efficiently than a marketplace listing possible tasks, since they do not need to interrupt their work to search for subsequent tasks. Assignment also provides a useful guarantee that every task will be answered. If tasks remain near the end of the priority queue for an excessive amount of time, we interleave them with tasks near the front by adjusting their priority to prevent large jobs from deadlocking the system. This means starvation is impossible so long as workers continue using the system.

Using a queue rather than a marketplace enables fine-grained control over the speed with which work is completed; by inserting work in the front of the queue, we can ensure that it is completed more quickly. If work is not completed within a reasonable timeframe, emails are sent out to the worker population informing them that there are tasks available. While not as robust at scale as other approaches to control latency in real-time systems, providing a work queue affords a simple mechanism for preventing starvation.

Quality Control Workflows and Exception Handling

Since workers are assigned tasks, it is crucial to build effective quality control mechanisms into the platform architecture. Each submitted task is inserted into the queue multiple times and presented to multiple workers via either a parallel or iterative model chosen by the requester when work is posted. In the iterative model, tasks are given to a sequence of workers in turn until a prespecified quantity of workers have sequentially edited an answer to the task. In the parallel model, if n distinct workers submit the same response or set of responses to a given question, the response is presumed to be correct and returned to the end user via a callback. If the workers disagree, the task is served to additional workers until a quorum is reached or an upper limit on the number of workers is hit, at which point the task is considered *ambiguous* and marked for review by managers.

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

Workers can elect not to do a particular task, but must provide a reason. Workers who are unfamiliar with a task or confused by its instructions can choose to report confusion via a "report problem" button on the interface, which requires workers to select a reason – whether the instructions were unclear, the instructions did not cover what to do in a particular instance, whether information was not found, or whether individual links or resources were not available. This guarantees that all tasks in MobileWorks will terminate with some form of feedback to requesters rather than starving silently. Reporting provides a kind of exception handling unique to MobileWorks: much as low-level errors in conventional programs can be caught by exception handlers, tasks that are not answered due to faulty design can be escalated to managers for potential resolution (Figure 2), and eventually back to the requester to help debug tasks.

Expert Sourcing and Qualification

A large number of tasks submitted to crowdsourcing systems require the attention of workers with specialized skills. Historically, expert workers have been identified either by automated tests or by constructing dedicated communities like 99Designs or StackOverflow that can attract these groups. MobileWorks allows requesters to specify qualification tests required for tasks. However, tests may not be able to identify the right set of workers for complex tasks such as writing and content editing because such open-ended tasks are difficult to score automatically.

MobileWorks uses crowd-powered mechanisms to recruit additional expertise as needed. When a task requiring skills is posted in the system, the system checks whether enough expertise is available in the worker population to meet the demand. If not, tasks are posted asking workers to refer individuals in their network or from their own contacts who might be able to meet the requirements. A manager reviews sample tasks or resumes submitted by the referred worker to ensure they have the skills claimed.

Once at least one expert with a given skill is identified by the system, he is tasked with the problem of checking the work of other potential experts in a peer-review system. Would-be experts are given subjective tasks that can be assessed by other experts to determine eligibility. This strategy assumes that managers are capable of assessing quality for tasks they lack fluency in; a more comprehensive option that we may implement is to use requesters for this role.

Fixed Payment and Requester Pricing

We believe that one cause of worker malice and task starvation in crowd marketplaces is improper incentivization: workers cannot earn appropriate wages. A brief survey given by the authors to 100 workers on Mechanical Turk indicated that the most pervasive complaint was low pay or failure to pay by requesters. Part of the reason is that employers are often unaware of the appropriate market price of their task, and so price tasks at excessively low prices on existing marketplaces. In contrast to markets, MobileWorks sets the price per task automatically based on required worker effort, enabling workers to earn fair or above-market hourly wages (on average) in their local zones.

The payout for a certain type of task is determined by dividing a target hourly wage by the average amount of observed time required to complete the task, excluding outliers. This pay structure incentivizes talented workers to work efficiently while ensuring that average workers earn a fair wage. Worker payouts are set so that individual workers in India will earn an average above-market wage of INR 65 (roughly $1.50) / hour working on computers and INR 25 ($0.60) / hour working on mobile devices. Given that typical workers in MobileWorks earned less than $3/day prior to joining, this wage provides a compelling source of income. Gaming this system to reach inaccurate prices would require collusion between large numbers of workers. Requesters are charged a fixed price for each task based on a multiple of the worker wage required. If a requester's tasks start to vary significantly in time, the requester is notified and asked to approve the new price before work continues.

We expect to admit US-based workers at US wages in the near future. For tasks such as English video transcription where market prices can be as high as $1-2 per minute of audio, as well as tasks requiring high domain expertise or US citizenship, these wages should be well-matched by market rates for their services.

Workers are paid for each answer they provide that matches the correct final answer determined through redundancy and manager review; their payout is tiered, with workers whose accuracy is below 80% only earning 75% of their overall possible earnings. This was a choice to encourage long-term attention to accuracy rather than considering individual tasks – when a worker recognizes that incorrect answers will affect payout for correct tasks downstream, their cost for providing incorrect answers increases. Workers can view their list of incorrect responses via a profile page; if they object, they can report tasks for review by a manager, which further improves system accuracy.

SOCIAL MANAGEMENT TECHNIQUES

Social management techniques provide a new strategy for producing effective work in crowdsourcing platforms. These contributions are unique to MobileWorks' architecture. We facilitate worker-to-worker and manager-to-worker communication using tools within and outside the platform.

Manager Recruitment and Supervision

MobileWorks grants managerial privileges to certain high-performing members of the crowd, termed *managers*. These workers play a central role in the efficient functioning of the architecture, as they supervise the work of other members and share in their earnings. Managers in

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

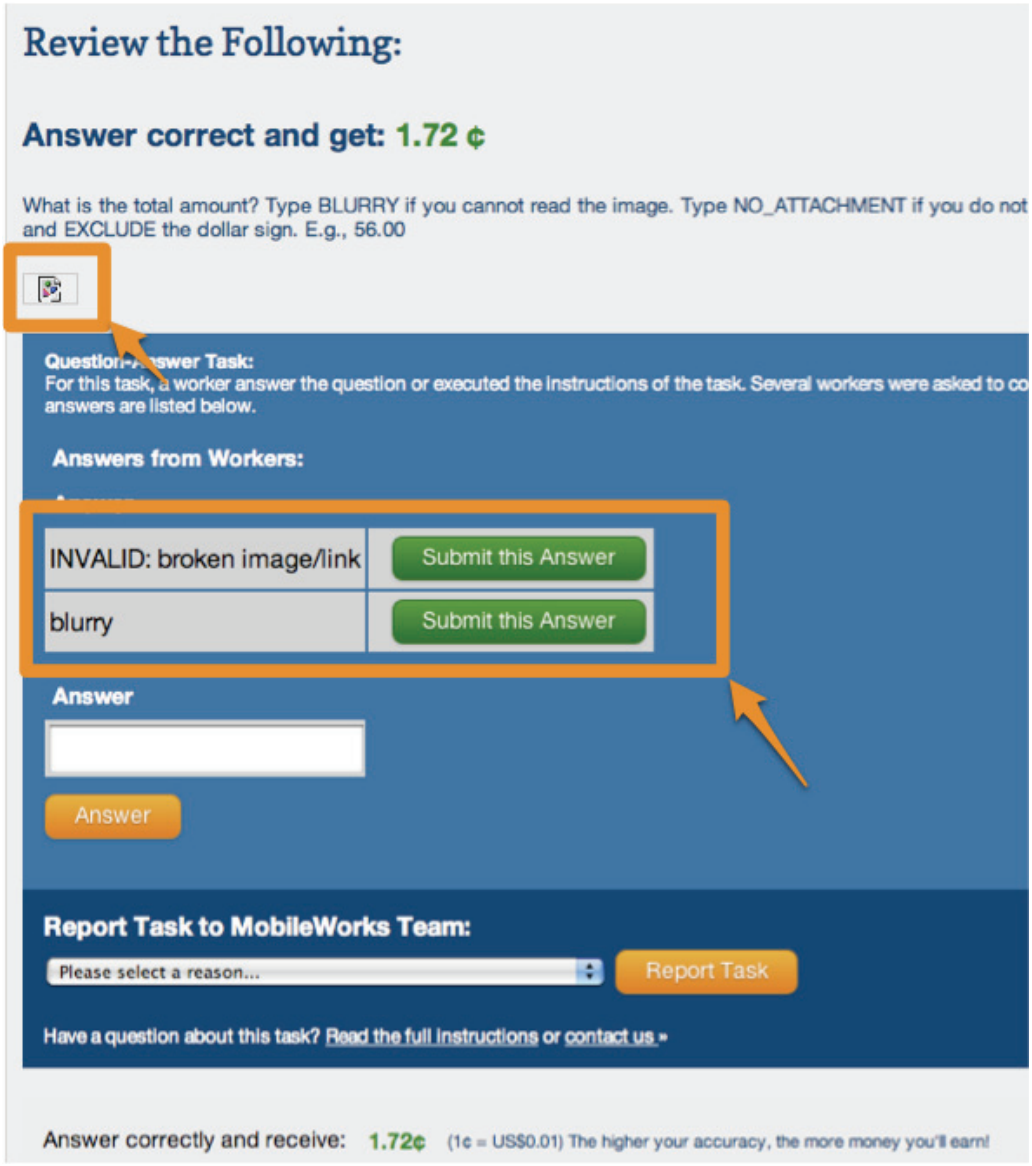

Figure 2. The interface presented to managers lets them determine the nature of exceptions to be reported to users (such as broken images or unclear instructions), disambiguate differing worker responses, and review performance of a team of workers they've recruited.

MobileWorks earn an additional 10-15% of what each worker they supervise earns, and are paid an additional per-task fee for each ambiguous task they resolve on behalf of the system (Figure 2).

This model initially emerged as a byproduct of our efforts to recruit overseas workers, but proved effective enough to use as a mechanism for maintaining quality. We identified the highest-performing members of our crowd by average accuracy and task volume and designated them as managers, asking them to recruit new workers. Peer recruitment allowed us to filter incoming workers based on demographic profiles and likely motivations; this process was largely controlled by managers, who were given a financial incentive to assemble effective teams. While the current crowd size of 500 workers is smaller than Turk's estimated 2000-10000 active workers, we expect that this kind of efficient filtration process can yield a comparably effective workforce over time.

Managers proved effective in unexpected ways. At their own initiative, we observed that managers began using outside methods not facilitated by the platform, such as online screencasts and emails, to demonstrate to workers how to carry out work (Figure 3). These had the added benefit of institutionalizing knowledge about how to carry out work, which could be passed on to other workers. Because these methods seem to be effective, and because

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

some workers have indicated they would use them more if embedded directly in the interface, we expect to add support for these interfaces in the future.

Worker-to-Worker communication

Worker error is often caused by an inability to understand the stated task or when instructions fail to cover corner cases that arise in a specific example of a task. To remedy this situation, we have implemented the ability for workers to communicate in real-time with each other and managers via a chat box embedded in the interface. This proved to be a useful tool to collaborate on work. Figure 3 shows instances where worker chat allowed workers to work around inadequate explanations in the task, to suggest additional examples that could be given to requesters, to teach other workers how to use the interface, and to confirm their theories about what a task meant.

SYSTEM PERFORMANCE

Aggregate Data

We have processed over 500,000 individual results since MobileWorks began field trials in March 2011, with the majority of tasks submitted by paid commercial customers and the remainder generated during testing and training periods. The posted tasks have varied from image tagging, processing, and speech recognition tasks, to web research and database population tasks, OCR and data extraction tasks, content creation tasks and simple planning tasks.

Few tasks (fewer than 5%) processed within MobileWorks were further processed by customers as part of their own workflows. While this is representative of the needs of customers using MobileWorks, it also indicates that the model succeeds in reducing the need for developers to engineer for quality on their end in many scenarios.

At its peak load, MobileWorks has produced roughly 20,000 answers per day. The average accuracy of individual workers in our system across all tasks is 85%, with the top quartile having an accuracy of 90-98%, the second quartile at 85-90%, and the third at 80-85%. To date, starvation has not occurred in MobileWorks, indicating that our architectural measures against it are effective – though we expect that our ability to qualify expert workers may bottleneck performance in the future.

Overall, the response of workers to MobileWorks has been overwhelmingly positive. We asked 30 web-based workers to rate the usability of the system on a five point Likert Scale. 30 out of 30 users rated the usability at a four or higher. Moreover, all users indicated that they would recommend the system to their friends and family, giving a 4 out of 5 or higher on a Likert scale. Workers said that the biggest advantage of the system was that the work could be done anywhere, at any time of the day. Typical comments include: “I could do the work and earn money while traveling to my regular job or even while watching television”.

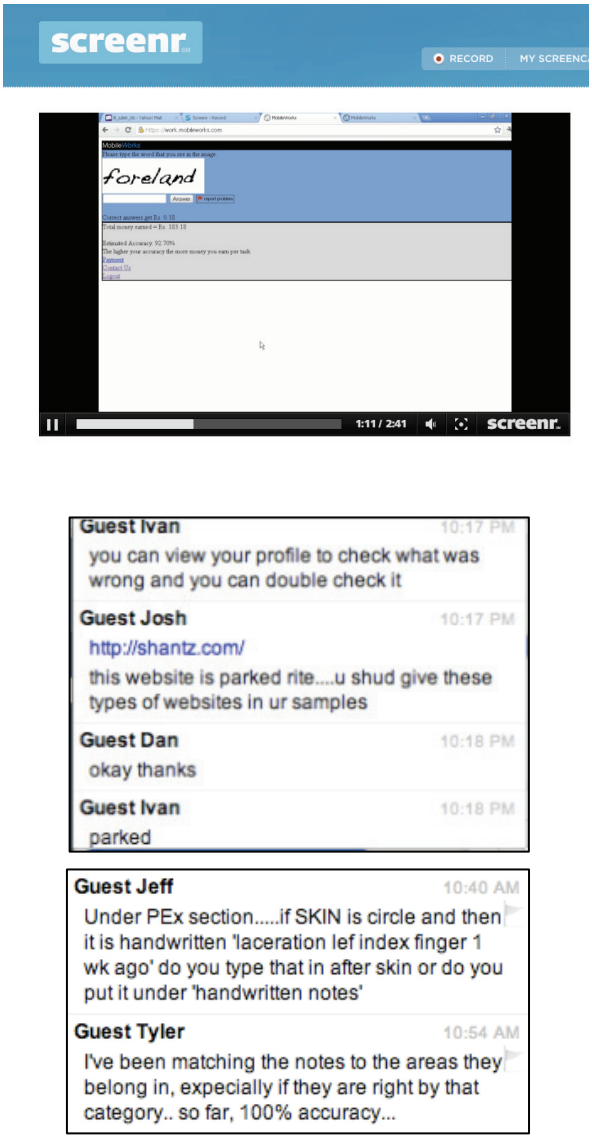

Figure 3. Manager-to-worker and worker-to-worker communication.

(above) A training video produced independently by a manager in MobileWorks, discussing how to use the mobile-phone interface and how to deal with task discrepancies. This video emerged organically as a consequence of our incentive structures.

(below) Two examples of worker-to-worker communication being used to clarify the user interface, the answer to a task, and the strategy used for solving it. Collaboration permits workers to debug tasks in situ and deal with unfamiliar tasks with managerial help without returning to requesters.

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE

How Much Does Managerial Review Help?

In addition to training workers, managers also play an essential procedural role in the quality assurance process by determining the final answer for tasks that are reported by workers as difficult. How much does this process improve quality? As a preliminary answer, we examined a set of 15,000 identical web research tasks for which the worker had to visit a website and search for a contact email address. If there was no email address available, the worker was to report the task as 'information not available'. When two workers reported a task, it would escalate to a manager.

Our results show that out of 800 escalated tasks marked reported by two workers, the managers found email addresses for around 550 that would otherwise be sent back to the user, which is a 70% improvement. This indicates that higher-quality managers can prove effective in disambiguation.

CONCLUSION AND FUTURE WORK

MobileWorks combines several known techniques for quality control with novel contributions in human computation in routing, pricing, management, and worker-to-worker interaction. Whereas the core ideas of routing, redundancy and review are not new to the crowdsourcing literature, their combination can provide an effective architectural alternative to the marketplace model.

Our experiences with MobileWorks suggest that as the crowd control systems place an increasing emphasis on accuracy, their architecture will begin to mirror the structure of traditional real-world firms, with collaboration and supervision playing a more important role in achieving accuracy than purely automated techniques. This is to be expected; crowd computing systems represent a mix of technological and organizational components. There are many additional improvements to the system under development that mirror the processes used within organizations, such as introducing workflows and improving task standardization.

We have reported only a subset of results from MobileWorks to give a sense of the overall impact of the techniques we have used. However, we have a sizable body of data from over a half-million tasks executed in the system. We look forward to making these available to the research community. MobileWorks is available for use online at www.mobileworks.com.

REFERENCES

1. Ahn, L. von, Dabbish, L. Labeling images with a computer game. CHI 2004.
2. Little, G., Chilton, L., Goldman, M., Miller, R.C., Exploring iterative and parallel human computation. HCOMP 2010.
3. Ahn, L. von. Games with a Purpose. IEEE Computer 39, 2006.
4. Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popovic, Z., and Foldit Players. Predicting protein structures with an online game. Nature, 2010.
5. Ahn, L. von, Maurer, B., McMillen, C., Abraham, D., Blum, M. reCAPTCHA: Human-based Character Recognition via Web Security Measures. Science, September 2008.
6. Oleson, D., Sorokin, A., Laughlin, G., Hester, V., Le, J., Biewald, L. Programmatic Gold: Targeted and Scalable Quality Assurance in Crowdsourcing. HCOMP 2011.
7. Ipeirotis, P. Mechanical Turk, Low Wages, and the Market for Lemons. 27 July 2010. http://www.behind-the-enemey-lines.com/2010/07/mechanical-turk-low-wages-and-market.html
8. Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., et al. Soylent : A Word Processor with a Crowd Inside. UIST 2010.
9. Kulkarni, A., Can, M., Hartmann, B. Collaboratively crowdsourcing workflows with Turkomatic. CSCW 2012.
10. Dow, S., Kulkarni, A. Hartmann, B., Klemmer, S. Shepherding the Crowd Yields Better Work. CSCW 2012.
11. Bigham, J.P., Jayantc, C., Ji, H., et al. VizWiz: nearly real-time answers to visual questions. UIST 2010.
12. Bernstein, M., Brandt, J., Miller, R., Karger, D. Crowds in two seconds: enabling realtime crowd-powered interfaces. UIST 2011.
13. Eagle, N. txteagle: Mobile Crowdsourcing. Internationalization, Design and Global Development, 2009.
14. Samasource. http://samasource.org
15. Khanna, S., Ratan A, Davis, J., Thies, W. Evaluating and Improving the Usability of Mechanical Turk for Low-Income workers in India, ACM SigDev 2010.
16. Benjamin B. Bederson and Alexander J. Quinn. Web workers unite! Addressing challenges of online laborers. CHI EA 2011.
17. Silberman, M.S., Irani, L. and Ross, R.. Ethics and tactics of professional crowdwork. XRDS 17 (2), 2010.
18. Lehdonvirta, V., Enrkvist M. Knowledge map of the virtual economy. InfoDev, 2011.
19. Ipeirotis, P. Analyzing the Mechanical Turk Marketplace. XRDS, 17 (2), 2010.
20. Ross, J., Irani, L., Silberman, M. S., Zaldivar, A., and Tomlinson, B. 2010. Who are the crowdworkers?: shifting demographics in Mechanical Turk. CHI 2010.

Digital Object Indentifier 10.1109/MIC.2012.72 1089-7801/$26.00 2011 IEEE