Interactive Crowds: Real-Time Crowdsourcing and Crowd Agents

Walter S. Lasecki and Jeffrey P. Bigham

Introduction

People can now connect instantly via nearly ubiquitous broadband connections, enabling interactive computational systems that are no longer constrained to machine-based automation alone. Instead, they can work in concert with the on-demand labor of people available on the web (the crowd), recruited on-demand and working synchronously together to complete tasks in real-time. The resulting model resembles a Distributed AI (discussed in Chapter <Dist AI>), but with a mix of human and machine agents composing the network.

Crowdsourcing workflows typically involve dividing tasks into smaller, separable tasks (Little et al. 2010; Dai et al. 2010). Using independent tasks provides an effective means of leveraging human intelligence to solve discretized problems, such as image labeling, offline transcription, handwriting recognition, and more. However, this model cannot handle acquiring consistent input for an ongoing task from workers. In order to expand the power of crowd algorithms, new models have been introduced that present approaches for continuous real-time crowdsourcing. This allows the crowd to be used to generate responses within the few-second time window needed for interactive tasks (Nielsen 1993). In these workflows, workers are engaged for longer periods of time, allowing them to receive feedback from the system as the task evolves due to their own input, as well as the input of others.

W.S. Lasecki (✉)
Department of Computer Science, University of Rochester, Rochester, USA
e-mail: wlasecki@cs.rochester.edu

J.P. Bigham
Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, USA

Department of Computer Science, University of Rochester, Rochester, USA
e-mail: jbigham@cmu.edu

P. Michelucci (ed.), *Handbook of Human Computation*,
DOI 10.1007/978-1-4614-8806-4_38, © Springer Science+Business Media New York 2013

We describe how models of continuous crowdsourcing can be used to enable task completion using both synchronous and asynchronous groups of workers. We then explore a new model of continuous real-time crowdsourcing called a "crowd agent" that allows groups of workers to interact with both users of crowd-powered systems and their environment, as if they were a single individual. This model provides a means of abstracting away the collective in crowdsourcing by making the crowd appear as a single intelligent entity.

Next, we discuss a set of crowd agents that have been developed based on this new model that are capable of a variety of different functions and actions that were not previously possible to complete using the crowd. Finally, we conclude with a discussion of the potential future advances that this approach enables.

Background

Human computation was introduced to integrate people into computational processes to solve problems too difficult for computers to solve alone, but has not been applied to real-time control problems. Human computation has been shown useful in writing and editing (Bernstein et al. 2010), image description and interpretation (Bigham et al. 2010; von Ahn and Dabbish 2004), and protein folding (Cooper et al. 2010), among many other areas.

Most abstractions for human computation focus on increasing quality, and generally introduce redundancy into tasks so that multiple workers contribute and verify the results at each stage. For instance, guaranteeing reliability through answer agreement (von Ahn and Dabbish 2004) or the find-fix-verify pattern of Soylent (Bernstein et al. 2010). Unfortunately, this takes time, making these approaches poorly suited for real-time domains. For a more in-depth discussion of several of the most widely used crowdsourcing workflows, see Chapter <Workflows>.

Several systems have previously explored how to make human computation interactive. For example, VizWiz (Bigham et al. 2010) answers visual questions for blind people quickly. It uses quikTurkit to pre-queue workers on Amazon's Mechanical Turk so that they will be available when needed. Crowd agents need multiple users to be available at the same time in order for its input mediators to work correctly. Prior systems have also needed multiple workers to be available. For instance, the ESP Game encouraged accurate image labels by pairing players together and requiring them both to enter the same label, although ESP Game players could also be paired with simulated players (von Ahn and Dabbish 2004). Seaweed reliably got Mechanical Turk workers to be available at the same time to play economic games by requiring the first worker to arrive to wait (generally for a few seconds) (Chilton 2009). Crowd agents similarly utilize the input of multiple workers and ask workers to wait until other workers have arrived, but engages them for longer control tasks. Specialized remote control systems even allow aircraft to be piloted remotely. The main difference between these prior systems and general real-time crowd control systems such as Legion, which allows multiple workers to

control a single interface, is the idea that multiple workers can collectively solve a problem as a single, more reliable worker.

Assistive Crowds

Prior work has shown how crowds can be used to assist users in their daily lives. Systems such as VizWiz (Bigham et al. 2010), which provides blind users with answers to visual questions in nearly real-time, shows that crowds can provide vital aid to users. Soylent (Bernstein et al. 2010) introduced the idea that crowd work could be made accessible from inside our existing applications—in that case, inside an existing word processor. Similarly, EmailValet (Kokkalis et al. 2013) uses the crowd to generate to-do lists from a partial view of a user's inbox. Mobi (Zhang et al. 2012) helps a user by generating a travel itinerary offline. But to truly work *with* the crowd, these systems need to be able to be recruited quickly and work synchronously with the end user. Soylent's reported delays of tens of minutes make the difference between collaborative cooperative work and iteration.

Overview

In this chapter, we present a discussion of the following:

- Real-time crowdsourcing, which provides responses to users within seconds
- Continuous crowdsourcing, which allows workers to engage in longer individual sessions to complete tasks which require workers to maintain context
- Crowd Agents, which combine the input of multiple workers contributing to continuous real-time tasks into a single output that retains the properties of a single reliable individual

Real-time crowdsourcing grew from the need for assistive systems, but allows a greater range of capabilities and interaction than was previously possible. These crowd-powered systems provide a useful service to end users, as well as insight into how users would interact with intelligent systems if they worked robustly.

Real-Time Crowdsourcing

Crowdsourcing has been shown to be an effective means of leveraging human computation to compute solutions to difficult problems; however, existing models only support usage in offline cases. Enabling systems that are able to use human computation to quickly and intelligently respond to user input can support a key aspect of nearly all systems: interaction, but requires latencies of only a few seconds. Current platforms such as Mechanical Turk typically require workers to browse large lists of

tasks, making it difficult to recruit workers within such a small time window. To recruit crowd workers to answer immediately, work on real-time and nearly real-time crowdsourcing has looked at pre-recruiting workers for a given task, then having them remain partially engaged with a task until they are prompted to switch to a new one (Bigham et al. 2010; Bernstein et al. 2011). Using this approach, it is possible to get workers to join a task in less than 1 s (Bernstein et al. 2012).

Applications

With such low response times possible, we can begin to think of interactive systems powered by the crowd. Bernstein et al. used these quick-acting crowds to enable a camera application that filters a short video down to a single best frame within seconds. Unlike Soylent, real-time crowds allow Adrenaline (Bernstein et al. 2011) to work behind the scenes the same way an automated would, without an explicit request step by the user, and VizWiz allows users to ask time-relevant questions and get responses within a minute. This responsiveness enables a new style of interaction with the crowd: seamless integration of new functions within the paradigm of traditional interfaces.

Continuous Crowdsourcing

Even with synchronous real-time systems, there are tasks that cannot be completed using one-off responses. For instance: instead of selecting a video frame, what if we wanted workers to help caption the video in real-time? Traditional approaches would divide the task into multiple pieces, and ask workers to accept a short task transcribing one of them. However, this means these approaches do not allow workers to maintain context of the topic or the terms being used, often divide words over two different pieces, and require workers to immediately recognize their place in the task and begin working from there in order to work properly. All of these factors reduce workers' ability to complete the task quickly and correctly.

Furthermore, issuing discrete tasks presents a model in which workers are frequently interrupted by either being asked to change topics or delayed before continuing to a subsequent task. Industrial and organizational psychology and cognitive science have looked at modeling the effects of these interruptions. For instance, prospective memory measures the ability to remember an ongoing task when interrupted by another. This ability to remember context is most strongly affected by the length and magnitude of the topic difference in the interrupting task. Unfortunately, on many crowd platforms, it would not be surprising to have a multi-part college course transcription task interleaved with a flower-labeling task.

In order to maintain context and allow workers to interact more robustly with the task at hand (for instance, learning new words that are used later in a conversation, or reacting to an object falling in the path of a robot being driven by the crowd),

Lasecki et al. introduced the idea of continuous crowdsourcing (Lasecki et al. 2011). In continuous crowdsourcing tasks, workers are engaged for longer periods of time in order to allow them to maintain this context. In the most general case, they are asked to connect to a task for as long as they choose, and will be able to seamlessly continue working on the same job until they choose to leave. In this model, workers must be compensated for their input on the fly in order to make the payments scale with the size of the task. In the next section, we will see how the idea of continuous crowdsourcing can be combined with real-time and synchronous tasks to enable the crowd to provide accurate, highly generalizable feedback.

Crowd Agent Model

Real-time, synchronous, and continuous tasks each individually present means of providing functionality in a fundamentally different way. However, in order to leverage human computation in this way, we are forced to develop one-off systems that use this collective input to accomplish some task. For instance, continuous crowdsourcing offers many advantages, but presents issues with how to provide users or systems with reliable responses (those verified by agreement between workers) in real-time. Allowing for repeated or multi-stage interaction, using the crowd requires a framework for merging collective input in real-time.

To address these issues, Lasecki et al. (2011) introduced the Crowd Agent model of crowdsourcing. This model recruits workers to complete synchronous continuous tasks in real-time, then uses a task-specific input mediator to merge their results into a single output. This allows the system to harness the power of collective intelligence, while abstracting away the multitude of responses and allowing the user or system to interact with a single, highly skilled, intelligent agent. Furthermore, using this model presents the ability to easily partially automate tasks by using existing automated systems as individual contributors, allowing systems to benefit from a synthesis of human and machine intelligence, and to scale gracefully towards becoming fully automated in the future.

Advantages

This method not only allows for new types of repeated-interaction tasks to be completed, but also strives to enable the crowd to retain many of the beneficial aspects of a single human user. Some of these properties that endow the crowd with a sense of agency include:

- **Unified Output:** By merging the input of workers in real-time, these systems create a single output stream, similar to that of a single skilled user. This is a critical aspect for combining crowd-powered systems with existing single-user systems (i.e. GUIs).

- **Collective Memory:** Organizational memory refers to a process in which groups of individuals collectively remember and pass down information from one generation to the next. In the context of crowdsourcing, part actions and decisions can be passed down to the current set of workers both explicitly via messages or labels (Lasecki et al. 2013b) and implicitly via behaviors (Lasecki et al. 2012b), even when no workers from the current session were present when the memory was created.
- **Consistent Output:** We can leverage the idea of collective memory to support consistent actions by the crowd. That is, actions or behaviors that are in line with the crowd's previous actions. This is important in systems that users engage in repeated interactions with, such as intelligent assistants (Lasecki et al. 2012c), where the prior interactions must be reliably recalled to facilitate the interaction.

Crowd Agents

In this section, we briefly describe a few recent systems that have been developed using the crowd agent model, and explore how they each demonstrate new potential uses for and capabilities of the crowd (Fig. 1).

Legion

Legion (Lasecki et al. 2011) is a system that enables the crowd to control existing single-user interfaces. It was the first work to introduce the idea of crowd agents. It leveraged the ability of this model to create a single combined output using an input mediator to control existing interfaces without the need to modify them. Legion's input mediator selected a single 'leader' at any given time step (usually

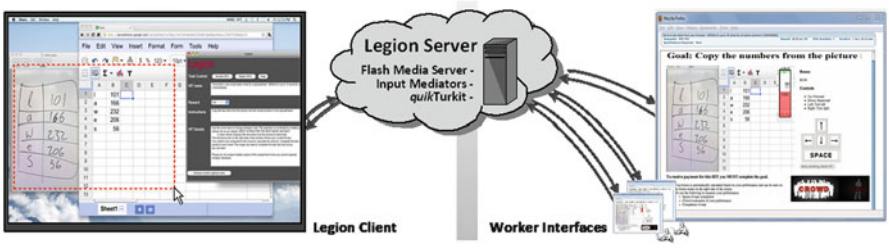

Fig. 1 Legion system architecture. In this example, a user has outsourced a spreadsheet text-entry task. The Legion client allows end users to choose a portion of their screen to send to crowd workers (outlined in *red* on the *left*), sends a video stream of the interface to the server, and simulates key presses and mouse clicks when instructed by the server. The server recruits workers, aggregates the input of multiple workers using flexible input mediators, and forwards the video stream from the client to the crowd workers. The web interface presents the streaming video, collects worker input (key presses and mouse clicks), and gives workers feedback

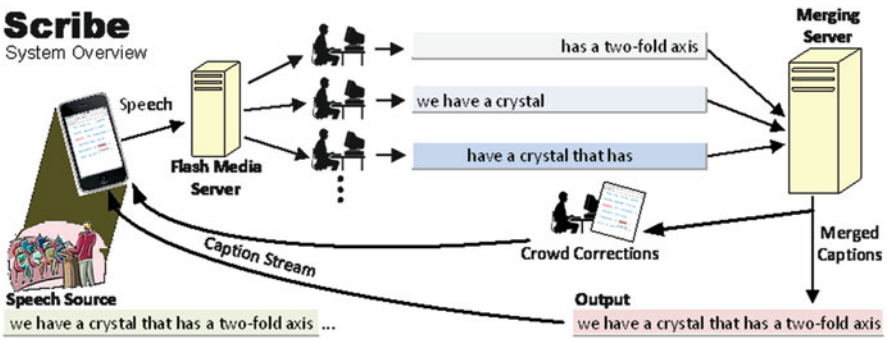

Fig. 2 Legion:Scribe system. Audio is sent to multiple non-expert captionists who use Scribe's web-based interface to caption as much of the audio as they can in real-time. These partial captions are sent to a server to be merged into a final output stream, which is then forwarded back to the user's mobile device

~1 s in length) to control the system, instead of averaging inputs, taking a vote, or letting a single individual or series of individuals control an interface. This leader is selected from the set of workers based on their past agreement with the collective as a whole, and at each time step workers are re-ranked, and the best leader at that point is selected.

Legion was demonstrated effective on a variety of tasks that ranged from robot navigation to controlling word processing applications, performing OCR, and enabling assistive keyboards. Further work demonstrated the crowd's ability to remember information over time, even in the presence of complete worker turnover, on a virtual navigation task (Lasecki et al. 2012b).

Legion:Scribe

Legion:Scribe (aka 'Scribe') (Lasecki et al. 2012a) is a system that enables groups of non-expert typists, such as those available from the crowd, to caption audio in real-time. To accomplish this, Scribe extended the underlying input mediator used in Legion to synthesis inputs from the whole set of workers instead of deciding on a single worker to listen to at any given time. This merger is performed using Multiple Sequence Alignment (MSA), a process most commonly associated with genome sequencing in computational biology (Naim et al. 2013). Using this approach, Scribe is able to use the partial captions of multiple workers to generate a single complete final transcript (Fig. 2).

Real-time captioning converts speech into text with a per-word latency of less than 5 s. Real-time captions are a vital accommodation for deaf and hard of hearing people that provides access to spoken language. Prior to Scribe, the only viable option for providing real-time captions were expensive and hard-to-schedule professionals who required 2–3 years of training and cost $100–$300 per hour or more (depending on skill set). Since Scribe is able to use anyone who can hear and type,

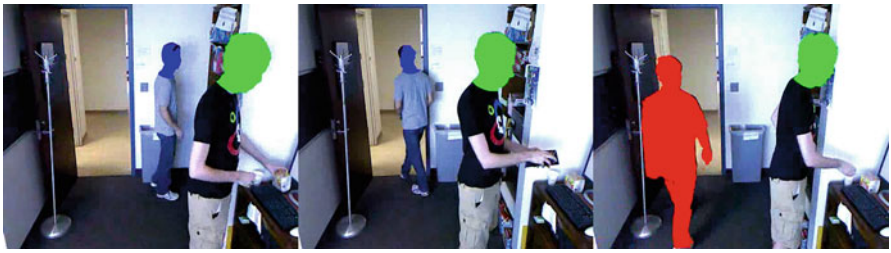

Fig. 3 Example of the video steam (with automatically generated privacy veils) that Legion:AR presents to workers during the activity labeling process. Each actor in the scene is color-coded to allow workers to identify which person's actions are being labeled

without the need for prior training, workers can easily be recruited for $8–10 per hour. Scribe can use as few as 3–6 workers to reach professional-level quality, meaning the same accommodation can be provided for a fraction of the cost. Furthermore, by recruiting crowd workers on-demand, the captioning services can be made far more available than is possible using professionals.

Legion:AR

Legion:AR (Lasecki et al. 2013b) is a system that uses the crowd to generate activity recognition labels in real-time. Activity recognition is important because it allows systems to provide context-relevant responses to users. Legion:AR focuses primarily on two domains: (i) an assistive living domain in which prompting and monitoring systems help cognitively impaired and older users live more independently for longer, and (ii) a public monitoring domain in which systems provide timely assistance by observing actions in a public space, such as calling an ambulance when a car accident is observed, or calling the police when an armed robbery or other crime is observed (Fig. 3).

Automated systems have struggled with these types of tasks because people perform actions in very different ways and in very different settings. In most cases these variations require explicit training in advance, meaning that systems are brittle and unable to handle change or new actions well. Legion:AR allows an automated system to call on the crowd in real-time to provide labels and training data. Unlike even experienced labelers, Legion:AR is able to produce these labels with extremely low latency, and keep up with live video. To do this, it uses an input merging approach similar to Legion:Scribe, in which labels are aligned and merged into a single stream. In order to provide consistent labels, it also presents workers with a display of what the other workers are currently suggesting, and labels those that have previously been used for similar activities (Fig. 4).

Once these labels are generated, they are used to train an automatic system. This means that over time, the system is able to gracefully transition from being fully crowd-powered to being fully automated.

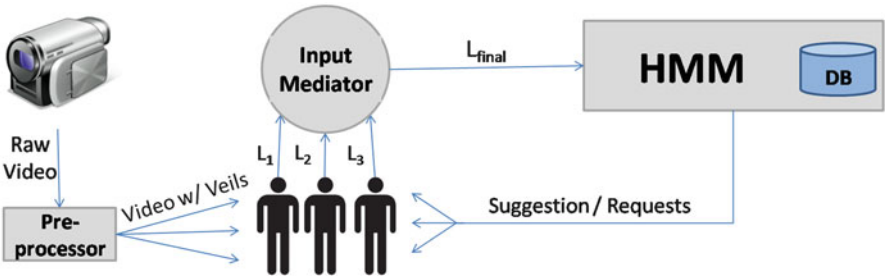

Fig. 4 Legion:AR system. Workers submit labels to Legion:AR, which forwards a final label and segment to train the HMM. The HMM can then add this to a database and use the information to identify the activity later and forward that prediction back to the crowd

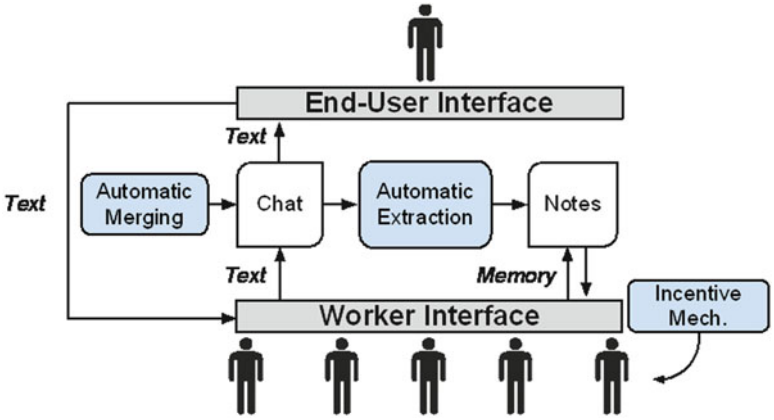

Fig. 5 Chorus system. Chorus is a framework for conversational assistance that combines human and machine intelligence. To end users, Chorus operates like a typical instant messaging client—workers type or speak messages, and receive responses in natural language. Crowd workers propose and vote on candidate responses, and are motivated to contribute via the incentive mechanism

Chorus

Chorus (Lasecki et al. 2013c) is a system that provides a conversational interface to a crowd-powered personal assistant. Conversational interaction has been a goal of natural language processing (NLP) researchers for several decades because it enables fluid interactions that more closely resemble those between people. Chorus leverages people's understanding of the state of the conversation, the context it occurs in, and common knowledge facts, to complete a requested information-finding task (Fig. 5).

Fig. 6 An example of a multi-part question that required accurate framing in the image. This task took over 10 min to complete using VizWiz, while similar tasks take less than a minute using Chorus:View because workers can help the user iteratively refine the framing and use prior knowledge to answer questions

In order to respond with the same consistency as a single assistant would, the Chorus uses an explicit propose-and-vote scheme combined with an incentive mechanism that encourages workers to contribute useful information, and then agree with others if their response would be more appropriate. It also provides a 'notes' window to explicitly allow a second synchronous crowd to curate memories from throughout the conversation. This allows workers to not only be consistent within a given discourse, but also between multiple sessions in which different workers may be present. Experiments showed that the crowd was able to answer over 84 % of user questions (including follow-up and clarification questions) accurately, and successfully remember information from prior interactions.

Chorus:View

Chorus:View (Zhong et al. 2012) combines the ability to hold conversations with the crowd with a visual question answering service similar to VizWiz. By using streaming video instead of single images, and allowing users to engage in an ongoing conversation with the crowd about what is shown, View enables a much more fluid support of visual tasks, similar to how a single person assist if they were collocated with the user. This is particularly beneficial for tasks that involve consecutive questions, such as finding a given type of food, locating the instructions, then finding the cook time, or even just framing an item in the camera's view. All of which turn out to be very common tasks that blind users need answers to (Brady et al. 2013) (Fig. 6).

Future Directions

Using the models of crowd work discussed here allows for a class of systems which can interact with users in a natural way, understand the context of the surroundings, and respond in a manner consistent with prior interactions. In the future, we expect

to be able to expand the scope of the human intellect that such systems are able to harness in order to understand longer term collective beliefs, desires, and intents. We can also use these crowds as a means of training existing artificial intelligence systems on-demand.

Novel Workflows

Using multiple synchronous workers instead of just one allows for novel workflows that would not otherwise be feasible using a single individual. For example, Lasecki et al. (2013a) showed that using multiple automatically coordinated workers, it is possible to slow down the playback rate of audio to make the captioning task in Scribe easier, while reducing the per-word latency. Slowing down the audio to be captioned presents workers with an easier motor task, resulting in increased performance. This also allows workers to keep up with each word as they hear it instead of first listening to a segment of audio, memorizing it, and then typing it, meaning latency is also reduced. However, while effective, this approach is not possible to use in real-time with a single worker, who would necessarily fall behind. In contrast, by using the crowd, it is possible to automatically interleave workers so that someone is always captioning live content. In the future we expect to see more such workflows that improve on what is possible for a single user to accomplish alone.

Improved Synthesis with Automatic Approaches

For many of the problem domains presented above, automated systems exist that try to generate solutions with varying degrees of success. One of the greatest benefits to the crowd agent model is that it treats each contributor as a noisy input, while being agnostic to exactly how it has generated its answer. This allows automated systems to be used as individual contributors that can learn and grow over time, just as human workers do. Using multiple inputs simultaneously also presents new opportunities to train systems which are still being explored. As seen above in Legion:AR, it also provides a means of smoothly transitioning between a system that is fully crowd powered to one that is fully automated, all without ever needing to expose end-users to an unreliable system during training.

Conclusion

In this chapter, we have presented an overview of the current work in interactive crowd systems, as well as some possible directions of future work. Work on continuous real-time crowdsourcing systems promises to enable interactive intelligent systems that can both operate using human intelligence and train automated systems

in novel ways. They also allow human workers to work jointly on tasks in novel and efficient ways. The potential of these models is just beginning to be explored, but these systems lay the groundwork for interactive intelligent systems, powered by the crowd.

References

Bernstein MS, Little G, Miller RC, Hartmann B, Ackerman MS, Karger DR, Crowell D, Panovich K (2010) Soylent: a word processor with a crowd inside. In: Proceedings of the 23nd annual ACM symposium on user interface software and technology, UIST'10, ACM, New York, pp 313–322

Bernstein MS, Brandt JR, Miller RC, Karger DR (2011) Crowds in two seconds: enabling real-time crowd-powered interfaces. In: Proceedings of the 24th annual ACM symposium on user interface software and technology, UIST'11, ACM, New York, pp 33–42

Bernstein MS, Karger DR, Miller RC, Brandt J (2012) Analytic methods for optimizing real-time crowdsourcing. In: Proceedings of the collective intelligence, Boston, MA

Bigham JP, Jayant C, Ji H, Little G, Miller A, Miller RC, Miller R, Tatarowicz A, White B, White S, Yeh T (2010) Vizwiz: nearly real-time answers to visual questions. In: Proceedings of the 23nd annual ACM symposium on user interface software and technology, UIST'10, ACM, New York, pp 333–342

Brady E, Morris MR, Zhong Y, Bigham JP (2013) Visual challenges in the everyday lives of blind people. In: Proceedings of the ACM SIGCHI conference on human factors in computing systems (CHI 2013), Paris

Chilton L (2009) Seaweed: a web application for designing economic games. Master's thesis, MIT

Cooper S, Khatib F, Treuille A, Barbero J, Lee J, Beenen M, Leaver-Fay A, Baker D, Popovic Z, Players F (2010) Predicting protein structures with a multiplayer online game. Nature 466(7307):756–760

Dai P, Mausam, Weld DS (2010) Decision-theoretic control of crowd-sourced workflows. In: Twenty-fourth AAAI conference on artificial intelligence (AAAI 2010), Atlanta

Kokkalis N, Köhn T, Pfeiffer C, Chornyi D, Bernstein MS, Klemmer SR (2013) EmailValet: managing email overload through private, accountable crowdsourcing. In: Proceedings of the 2013 conference on computer supported cooperative work (CSCW '13), ACM, New York

Lasecki WS, Murray K, White S, Miller RC, Bigham JP (2011) Real-time crowd control of existing interfaces. In: Proceedings of the ACM symposium on User Interface Software and Technology, UIST'11, ACM, New York, pp 23–32

Lasecki WS, Miller CD, Sadilek A, Abumoussa A, Borrello D, Kushalnagar R, Bigham JP (2012a) Real-time captioning by groups of non-experts. In: Proceedings of the ACM symposium on User Interface Software and Technology (UIST 2012), Boston, MA, pp 23–34

Lasecki WS, White S, Murray K, Bigham JP (2012b) Crowd memory: learning in the collective. In: Proceedings of collective intelligence (CI'12), Boston

Lasecki WS, Miller CD, Bigham JP (2013a) Warping time for more effective real-time crowdsourcing. In: Proceedings of the international ACM conference on human factors in computing systems, CHI'13, page to appear, Paris, France

Lasecki WS, Song Y, Kautz H, Bigham J (2013b) Real-time crowd labeling for deployable activity recognition. In: Proceedings of the international ACM conference on computer supported cooperative work and social computing (CSCW 2013), San Antonio, TX, pp 1203–1212

Lasecki WS, Wesley R, Nichols J, Kulkarni A, Allen JF, Bigham J (2013c) Chorus: a crowd-powered conversational assistant. In: Proceedings of the 23nd annual ACM symposium on user interface software and technology (UIST'13), St. Andrews, UK. UIST 2013. PP 151-162

Little G, Chilton LB, Goldman M, Miller RC (2010) Turkit: human computation algorithms on mechanical Turk. In: Proceedings of the 23nd annual ACM symposium on user interface software and technology, UIST'10, ACM, New York, pp 57–66

Naim I, Lasecki WS, Bigham JP, Gildea D (2013) Text alignment for real-time crowd captioning. In: Proceedings of the North American Chapter of the association for computational linguistics (NAACL 2013), To appear, Atlanta, GA

Nielsen J (1993) Usability engineering. Morgan Kaufmann, San Francisco

von Ahn L, Dabbish L (2004) Labeling images with a computer game. In: Proceedings of the SIGCHI conference on human factors in computing systems, CHI'04, ACM, New York, pp 319–326

Zhang H, Law E, Miller RC, Gajos K, Parkes DC, Horvitz E (2012) Human computation tasks with global constraints. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI'12), pp 217–226, Austin, TX

Zhong Y, Thiha P, He G, Lasecki WS, Bigham JP (2012) In: Proceedings of the ACM conference on human factors in computing systems work-in-progress (CHI 2012), Austin