# Powering Spoken Language Interactions With the Crowd

**Walter S. Lasecki**
ROC HCI, Computer Science
University of Rochester
Rochester, NY 14620 USA
wlasecki@cs.rochester.edu

**Alan Ritter**
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213 USA
aritter@cs.cmu.edu

**Jeffrey P. Bigham**
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
jbigham@cmu.edu


*CHI 2014*, April 26–May 1, 2014, Toronto, Ontario, Canada.

ACM 978-1-4503-2474-8/14/04.

## Abstract
Spoken language interfaces (SLIs) have the potential to facilitate more natural interactions between people and computers, but realizing this potential requires spoken language interfaces to not only recognize the words people say but also the meaning and intent behind them. These problems are very difficult to solve individually and each must be solved for the other to perform optimally. As a result, it is very difficult to build prototype systems that work well enough to allow spoken language interaction to be studied in the real world. In this paper, we present a crowd-powered pipeline that allows robust, interactive SLIs to be prototyped and deployed today. The pipeline is at first entirely powered by human intelligence, but smoothly scales towards being fully automated by using the data it collects to improve the system.

## Author Keywords
Crowdsourcing; conversational interaction; dialog systems

## ACM Classification Keywords
H.5.m. [Information Interfaces and Presentation]: Misc.

## Related Work
Spoken dialogue systems have made tremendous progress over the past several years. We have a fairly good understanding of how to build dialogue systems given current technological limitations, but these systems are highly engineered for specific scenarios, and quickly fail once the conversation falls outside of their scope. To

move beyond these limitations, previous work has leveraged Wizard-of-Oz studies to design user interactions in the absence of a working system. In the meantime, *crowdsourcing*, has allowed human computation can be accessed easily and on-demand [2]. We discuss two main examples of continuous real-time systems that make the crowd appear to end-users as a single reliable entity to create interactive systems powered by the crowd [5].

*Legion: Scribe*
Legion:Scribe [4] is a system that allows groups of non-expert workers to convert speech to text in real-time (within 5 seconds) by dividing the audio input task between workers. Scribe was initially envisioned as a means of providing more affordable real-time captions for deaf and hard of hearing students in classrooms, where it lets non-experts caption for a fraction of the cost of a professional. More generally, Scribe provides an alternative to ASR that is more reliable, lower latency, and more adaptable to new domains. When used as part of a spoken dialog interface, Scribe can ensure reliable conversion of speech to text for use by the dialog system.

*Chorus*
Chorus [5] is a crowd-powered conversational assistant that allows multiple crowd workers to act as a single, reliable conversational partner capable of answering general knowledge questions. Chorus asks workers to propose and vote on one another's responses to the user. An incentive mechanism elicits accurate responses from workers by paying more for more impactful contributions. Workers are also able to make notes to other current and future workers so that they can see the current context of the conversation, even if they just joined. This allows the crowd to collectively stay on the same page, and appear to the end-user as a single conversational partner.

Chorus can generate more robust training data sets than possible with prior systems. Typically, systems are able to see a question or comment provided by a user, and 'accept', 'reject', or 'retry' information from the user's response. In Chorus, multiple answers are created and voted on by the crowd fro each response, meaning alternate answers and their relative agreement level can be used in the training process for a dialog system.

## Crowdsourcing Spoken Dialog Interaction

Our goal is to create a framework that uses the crowd to enhance the ability of automated systems, instead of replacing it entirely. Using the crowd to support SDSs allows systems to make definable parts of their system robust while others remain experimentally automated, or choose to create systems that are completely reliable from the end-users' point of view, but of which only specific portions of the interaction is supported automatically.

*Architecture*
Figure 1 shows the architecture of a hybrid spoken dialog system that uses both human and machine input to hold reliable conversations with users. This architecture is, by design, similar to that of a traditional dialog system: the Scribe plays the role of the speech recognizer, Chorus is the input parser, dialog manager, and natural language generator, and the output module provides feedback. The difference is that by using people as part of the underlying computational resources available, this architecture can more reliably maintain context, prevent conversations from getting derailed, and better understand users by leveraging common sense.

*Building and Sharing Context*
Using this hybrid architecture, we can build a shared context between different parts of the system to improve

reliability. Introducing the crowd can also help further, for example, Scribe can create word sets that can be used to recover topic information for Chorus workers, who can also be asked to mark key words, helping to increase the accuracy of topic modeling. Chorus workers can use the topic models created from the proposed responses to create better language models that Scribe can in turn use to better predict which words that workers have typed are correct and fix mistakes where they are found. Furthermore, both Scribe and Chorus can elicit the tone and mood of the conversation from worker to ensure the response provided by the output module have the correct inflection and emphasis.

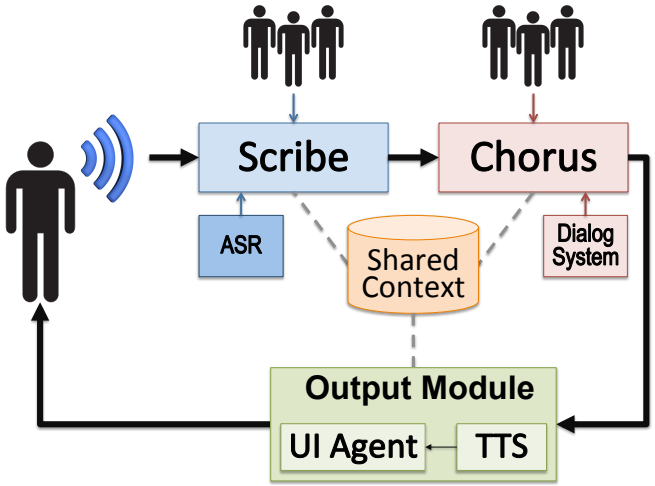


**Figure 1:** A crowd-powered spoken dialog system architecture. When users speak, it is converted to text in under 5 seconds by Scribe using non-expert captionists. The text is then sent to Chorus, a crowd-powered conversational assistant that generates an answer that is sent to the user in the form of text or speech (using text-to-speech), depending on the output module used. Both Scribe and Chorus can use input from existing automated systems by treating them as if they were workers providing error-prone input. Context such as topic information can be shared between speech processing, dialog, and output components to further improve performance.

*Training AI Systems*

Using the architecture described above, we propose to gather high-quality training data within the context of a working system to train models models capable of automating components of the conversational agent in ways not previously possible. For example, Chorus naturally generates multiple competing responses suggested by workers, each of which is rated based on other workers' votes, providing a natural source of training data for an automated response ranking algorithm. We also envision crowd enabled open-domain spoken dialog systems as a natural way to enable the collection of conversationally relevant commonsense knowledge, complementary to existing large-scale knowledgebases such as NELL [3]. Additionally, both Scribe and Chorus allow automated systems to propose their own answers (which are treated as potentially unreliable just like any other worker) with the knowledge that the crowd can correct for their mistakes. This lets systems get feedback on even lower-confidence guesses without disrupting the end-user's experience from the system, letting the automated system "fail fast" to learn more quickly.

*Combining Human and Machine Intelligence*

Humans and machines are better at different things, and as such, combining their strengths can lead to better results than either can achieve independently. For example, people are still better at holding conversations that feel natural, but cannot search for information as fast as a computer. Augmenting peoples' ability to respond appropriately by automatically fetching information that might be useful can help crowd-powered systems provide quicker responses without asking workers to generate answers based on insufficient information Humans and machines also make different types of errors, which can be modeled to help improve error detection and the ability to automatically rate confidence in a response. For example, when comparing ASR to non-expert human captioning in Scribe, we see that ASR is more likely to substitute an incorrect word that *sounds* the same (e.g., "Lexus" for "axis"), whereas people are more likely to substitute words that *mean* the same thing (e.g., "someone" for

"somebody"). On the other hand, if both agree on a word, experience has shown that it is much more likely to the a correct word.

*Prototyping and Exploratory Interactions*
Creating and prototyping SDSs for new situations and roles is often made difficult by low user expectations and systems unprepared to handle situations that are not known a priori. As a result, users may tend to avoid the system because they don't know where and when it will break. In these types of prototyping cases, the crowd can be used to fill in for the automated system the first few times a specific domain is encountered. This allows end users to learn the capabilities of the intended system, while also providing a means of collecting use-case and training data that is helpful during the continuing development of the automated system.

*Deployable Systems*
Crowdsourcing can also help systems already-deployed in real-world situations. Unknown domains in deployed systems are often a problem, similar to in the prototyping case. For instance, even a change as simple as a new domain where the context or lexicon used differs from prior examples can result in complete failure of an automated system. In these cases, our crowd-powered pipeline can provide a means of supporting deployed systems until enough training data can be provided. Additionally, not all interactions that might have been prototyped using the crowd in the manner described above can be automated currently, or they fall in the "long tail" and are too costly to develop relative to each one's expected usage. This means a deployed system might have known deficiencies. In these cases, the crowd can be used to "fill in the gaps" until automated approaches can be developed.

## Conclusion
We have presented the architecture of a crowd-powered spoken dialog pipeline that is capable of robust interaction with users in open domains. This system can be used to quickly and flexibly take the place of a dialog system, or help train an existing dialog systems in order to scale towards being fully automated. For the first time, this makes it possible to rapidly iterate on spoken dialog interactions with real users, in real-world settings, to gain a better understanding of how spoken interaction can support their needs. [1]

## Acknowledgements
This work was supported by the NSF under awards #IIS-1149709 and #IIS-1116051, and by a Microsoft Research Ph.D. Fellowship.

## References
[1] Allen, J. F., et al. Plow: a collaborative task learning agent. In *AAAI* (2007), 1514–1519.
[2] Bigham, J. P., et al. Vizwiz: nearly real-time answers to visual questions. In *UIST* (2010), 333–342.
[3] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E. J., and Mitchell, T. Toward an architecture for never-ending language learning. In *AAAI* (2010), 1306–1313.
[4] Lasecki, W. S., Miller, C., Sadilek, A., Abumoussa, A., Borrello, D., Kushalnagar, R., and Bigham, J. P. Real-time captioning by groups of non-experts. In *UIST* (2012), 23–34.
[5] Lasecki, W. S., Wesley, R., Nichols, J., Kulkarni, A., and Bigham, J. P. Chorus: A crowd-powered conversational assistant. In *UIST* (2013), 151–162.

[1] For an extended version of this paper, see: http://hci.cs.rochester.edu/pubs/pdfs/crowd_slt-full.pdf