Crowdsourcing Multimodal Dialog Interactions: Lessons Learned from the HALEF Case

Vikram Ramanarayanan[†], David Suendermann-Oeft[†], Hillary Molloy[†], Eugene Tsuprun[‡], Patrick Lange[†] & Keelan Evanini[‡]
Educational Testing Service R&D
[†]90 New Montgomery St, #1500, San Francisco, CA
[‡]600 Rosedale Rd, Princeton, NJ
vramanarayanan@ets.org

Abstract

We present a retrospective on collecting data of human interactions with multimodal dialog systems ("dialog data") using crowdsourcing techniques. This is largely based on our experience using the HALEF multimodal dialog system to deploy education-domain conversational applications on the Amazon Mechanical Turk crowdsourcing platform. We list the various lessons learned from this endeavor over two years and make recommendations on best practices to be followed by practitioners and researchers looking to crowdsource dialog data for a new domain.

Crowdsourcing Dialogic Interactions

Crowdsourcing has emerged as one of the most popular methods of collecting spoken, video and text data over the last few years (Eskenazi et al. 2013). The advent of multiple crowdsourcing vendors and software infrastructure has greatly helped this effort. Several providers also offer integrated filtering tools that allow users to customize different aspects of their data collection, including target population, geographical location, demographics and sometimes even education level and expertise. Managed crowdsourcing providers extend these options by offering further customization and end-to-end management of the entire data collection operation.

In this review paper we will particularly focus on the collection, development and testing of interactions between a human and an automated system using crowdsourcing methods. Several papers in the literature have pushed the use of crowdsourcing for the collection, validation, transcription and annotation of dialog data (see for example (Chernova, DePalma, and Breazeal 2011; DePalma, Chernova, and Breazeal 2011; Bessho, Harada, and Kuniyoshi 2012; Yang, Levow, and Meng 2013; Suendermann and Pieraccini 2013; Lasecki, Kamar, and Bohus 2013; Mitchell, Bohus, and Kamar 2014; Ramanarayanan et al. 2016a)). While different studies have typically used different dialog system infrastructures to crowdsource data, many of the actual techniques and considerations involved in such crowdsourcing data collections are very similar. Along these lines, we frame our discussion of the considerations needed while crowdsourcing dialog data in the context of HALEF, an open-source cloud-based dialog framework we have developed and continue to use for our multimodal dialog research.

Crowdsourcing offers a quick and cheap mechanism for data collection and annotation, particularly if the system is distributed and cloud-based. Crowdsourcing from standalone dialog systems is more challenging owing to potential technical limitations, but nonetheless possible. It also allows developers to design dialog systems in a more iterative manner owing to its rapid turnaround cycle, where one can start out with a system, deploy it, and use the collected data to update the system models and configuration (Ramanarayanan et al. 2016a). However, the drawback of such a data collection mechanism is the relative lack of quality control and verification as compared to in-person, laboratory data collections, which are much more controlled (a meta study on crowdsourcing for speech applications concluded that "although the crowd sometimes approached the level of the experts, it never surpassed it" (Parent and Eskenazi 2011)). This is exacerbated during multimodal dialog data collections, where it becomes harder to quality-control for usable audio-video data, due to a variety of factors including poor visual quality caused by variable lighting, position, or occlusions, participant or administrator error, or technical issues with the system or network (McDuff, Kaliouby, and Picard 2011).

When building and deploying any spoken dialog system (SDS) it is imperative to understand how well the system is performing to ensure an optimal user experience (UX). While such an endeavor is crucial and relevant during the process of bootstrapping a dialog system for a new domain or application, it is equally important to measure UX and system performance metrics for an SDS that is more mature to ensure a high quality of service. Much research has been conducted into the metrics one can use to quantify the performance and UX of an SDS (see for example (Danieli and Gerbino 1995; Walker et al. 1997; Walker, Wright, and Langkilde 2000; Möller 2004; Pietquin and Hastie 2013; Yang, Levow, and Meng 2012; Jiang et al. 2015; Evanini et al. 2008; Forbes-Riley and Litman 2011)). Such ratings can be obtained relatively easily and cheaply using crowdsourced surveys attached to the dialog system interface page.

The rest of this paper is organized as follows: The following section presents the different aspects of our data collec-

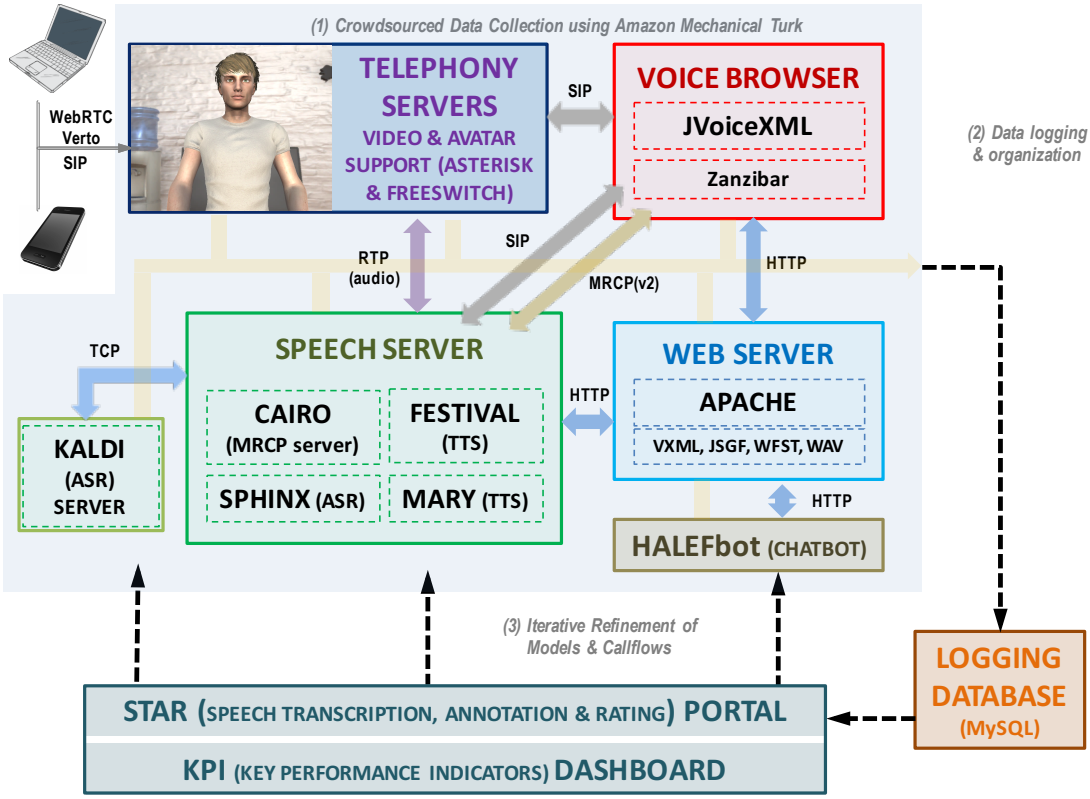

Figure 1: The HALEF multimodal dialog framework that we used for our crowdsourcing experiments.

tion setup, which has formed the basis of our crowdsourcing experiments. We then go through an example sequence of steps that we typically take in order to crowdsource a conversational task. Next, we enumerate a set of 10 lessons learned from our crowdsourcing experiments over two years and present recommendations for practitioners and researchers looking to crowdsource multimodal dialog data. We conclude with a discussion of the current state of the art and an outlook for the future.

Data Collection Setup

This section describes our data collection infrastructure and setup, as well as the conversational applications designed to elicit crowdsourced audiovisual interactions with AMT workers.

The HALEF Dialog System

The multimodal HALEF[1] (Help Assistant–Language-Enabled and Free) framework depicted in Figure 1 leverages different open-source components to form an SDS framework that is cloud-based, modular and standards-compliant. For more details on the architectural components, please refer to prior publications (Ramanarayanan et al. 2016b; Yu et al. 2016).

[1] http://halef.org

Conversational Tasks

The analysis in this paper is based on a set of goal-oriented conversational tasks developed for English language learners. The tasks were designed to provide speaking practice for non-native speakers of English across a wide range of common linguistic functions in a workplace environment, including scheduling a meeting, interviewing for a job, making requests, responding to offers, placing and taking orders, requesting a refund, etc. In addition, the tasks are designed to be able to provide feedback to the language learners about whether they have used the required linguistic skills to complete the task successfully.

For instance, one spoken dialog task (Food Offer) consists of a short conversation in which the system interlocutor (a co-worker) offers some food to the user and the user is expected to accept or decline the offer in a pragmatically appropriate manner. Figure 2 presents a flowchart schematic of this task and indicates the different branches in the conversation based on whether the users accepted the offer or not and whether the user's response was pragmatically appropriate or not. The range of expected user responses is quite limited in this task and the number of crowdsourcing responses that need to be collected for system training is relatively small. On the other hand, some of the conversational tasks are much more open-ended and elicit a wider range of speech from the user; for example, in a Billing Dispute task,

Table 1: *A sampling of some of the conversational tasks deployed. Along with the number of dialog states for each task, #(DS), we also list the number of dialog states which required a speech recognition and subsequent language understanding hypothesis to go to the next dialog state, #(DDS) (as opposed to an inconsequential state which just moves to the next state after end of speech has been detected).*

Item	Brief Task Description	# DS	# DDS	# of Calls	Handling Time (sec) Mean	Std. Dev.
Food Offer	Accept or decline an offer of food in a pragmatically appropriate manner	1	1	808	59.5	45.5
Meeting Scheduling	Invite a co-worker to a meeting based on a given schedule	4	3	323	110.5	124.1
Job Hiring Interview	Answer questions posed by an interviewer based on a given resume	8	3	660	294.6	104.6
Pizza Service	Pose as a customer services representative at a pizza restaurant and take a customer order	7	7	789	144.9	80.3
Meeting Request (Boss)	Request a meeting with your boss	5	4	909	80.2	35.9
Meeting Request (Friend)	Request a meeting with a co-worker peer	5	0	743	77.8	44.3
Order Refund	Request a refund on a defective item	5	5	952	82.3	77.0
Job Placement Interview	Interact with an interviewer at a job placement agency	30	4	1282	345.2	114.1
Coffee Shop Order	Order food and drink from a coffee shop	9	1	1210	135.3	66.8
Billing Dispute	Dispute charges on a customer phone bill	5	3	986	154.0	79.4

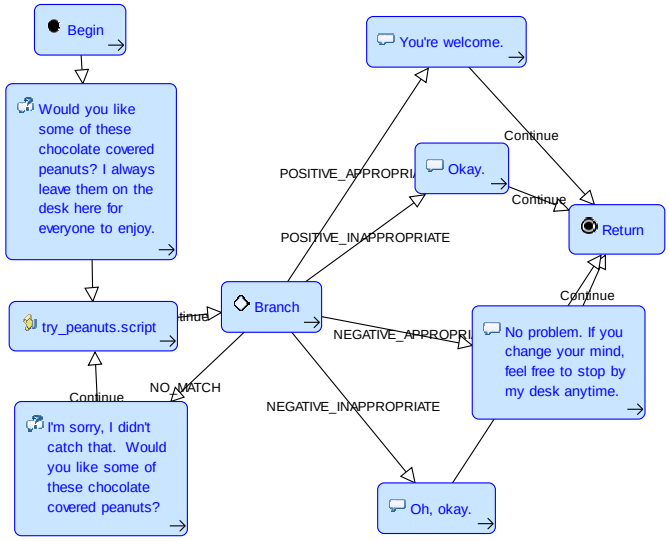

Figure 2: Example design of the Food Offer conversational task targeted at non-native speakers of English in which the language learner has to accept or decline an offer of food in a pragmatically appropriate manner.

the user has to participate in a conversation with an automated customer service representative and explain why the charges on the user's monthly phone bill (which is provided in an image along with the task) are incorrect.

Table 1 lists the conversational tasks that were deployed to the AMT platform as part of this study. Most of the tasks (including the two described above) contain system-initiated dialog scenarios, some also contain user-initiated dialogs. For example, in the Pizza Service task, users are required to play the role of a customer service representative at a pizza restaurant and take an order from a customer (the system interlocutor) who wants to order a pizza. In such a scenario, the automated customer waits for the user to ask a question ("What is your name?", "What toppings would you like on your pizza?", etc.) before replying with the appropriate response. Therefore, this task might be potentially harder than the other three, imposing more cognitive load on the user.

Table 2: *Crowdsourcing statistics.*

50+	unique native/first languages (L1s)
20+	conversational applications
21,333	interactions (68% include both audio and video)
785	hours of audio data
512	hours of video data
35,722	transcribed utterances
$\sim$ 325,368	transcribed words
$\sim$ 5,551	annotations
$\sim$ 17,093	ratings

Crowdsourcing Setup

We used Amazon Mechanical Turk[2] (and to a much smaller extent, Microworkers[3]) for our crowdsourcing data collection experiments. As mentioned earlier, crowdsourcing, and particularly AMT, has been successfully used in the past for the assessment of SDSs as well as for collection of interactions with SDSs (McGraw et al. 2010; Rayner et al. 2011; Jurcıcek et al. 2011). In our case, each spoken dialog task was its own individual HIT (Human Intelligence Task). In addition to reading instructions and calling into the system to complete the conversational tasks, users were requested to fill out a 2-3 minute survey regarding different aspects of the interaction, such as their overall call experience, how engaged they felt while interacting with the system, how well the system understood them, to what extent system latency affected the conversation, etc. Since the targeted domain of the tasks in this study is non-native English conversational practice, we restricted the crowdsourcing user pool for some of the HITs to non-native speakers of English; however, we also collected data from native speakers of English in order to test the robustness of the system and to obtain expected target responses from proficient speakers of English. Over the past couple of years, we have collected over 20,000 calls into the HALEF system amounting to nearly 800 hours of dialog speech data (more than two thirds of which have also video) from people all over the world (see Figure 5

[2] https://requester.mturk.com/
[3] https://microworkers.com/

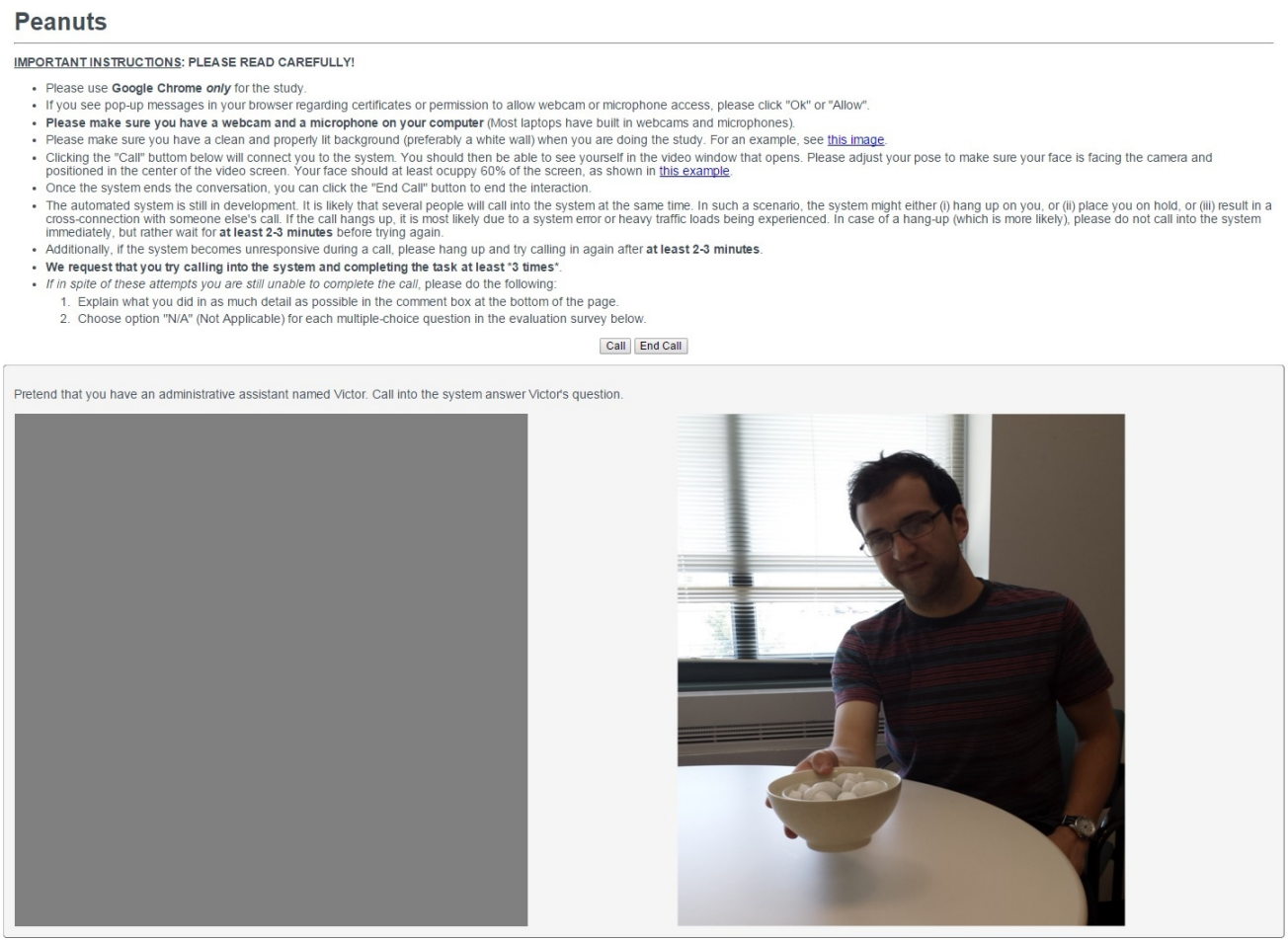

Figure 3: Example webpage that allows users to video-call into the HALEF dialog system by leveraging the WebRTC protocol. The page provides instructions to the caller and then directs the user to dial into the system by pressing the "Call" button. The specific application shown is a conversational task targeted at non-native speakers of English in which the language learner has to accept or decline an offer of food in a pragmatically appropriate manner. The callflow for this application is also depicted in Figure 2.

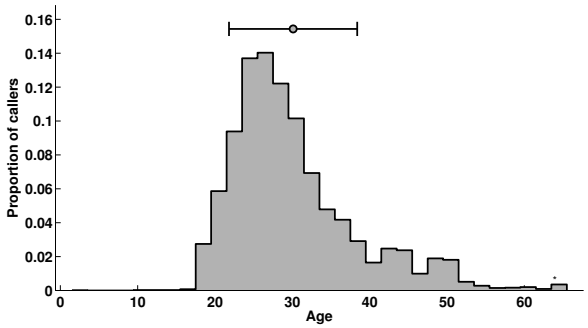

Figure 4: Distribution of ages of callers into the HALEF system. The bar on top represents the mean$\pm$1 standard deviation.

for a cartographic illustration of user locations) who interacted with multiple conversational applications (see Table 1). These users spanned a wide age range, with people as young as 20 and as old as 50 years of age (see Figure 4). See Table 2 for a compilation of our crowdsourcing statistics.

Typical Data Collection Process

The design, deployment and crowdsourced data collection of conversational applications include roughly the following steps:

1. Assessment developers first design the conversational application and develop a callflow for it using a flowchart-based software tool such as OpenVXML[4]. See Figure 2 for an example callflow. Such a callflow application encapsulates various resources required for the dialog system, such as grammars or language models and audio recordings for playback, and also other contains specifications, such as voice activity timeout threshold, barge-in settings, grammar formats, etc. During this process:
   - One or more statistical language models or grammars are trained using collected data (if previous data from this application/domain was collected earlier) or a set of sample utterances (if this is a new application).

[4] https://github.com/OpenMethods/OpenVXML

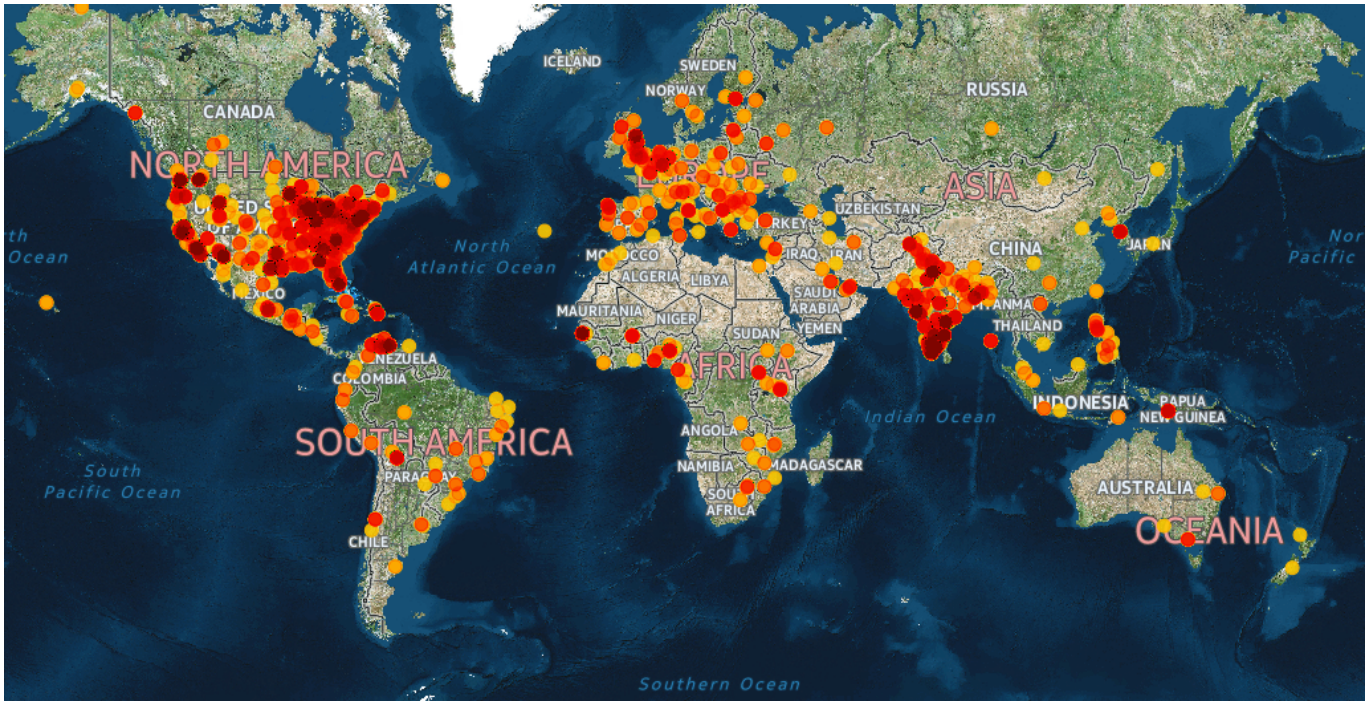

Figure 5: Geographical distribution of callers into the HALEF dialog system from all over the world. Hotter colors (closer to deep red) indicate a higher density of callers.

- If the conversational application is to use prerecorded voices, then one or more voice talents record the desired audio files.

2. We then write a web-based HTML interface page for Turkers to call into the dialog system that includes detailed instructions on how to complete the task, including, but not limited to, stimulus material, troubleshooting tips and survey questions. We also include instructions requesting callers to test that their audiovisual recording equipment such as microphones and webcams are in good working order. See Figure 3 for an example screenshot of one such webpage for a conversational application.
3. We create a task on the crowdsourcing platform of interest, choosing crowd filters as per the requirements of the task. For example, for certain applications, we might want to collect data from primarily non-native speakers of English, in which case we set a geographical filter. We also set the payment for the task at this stage.
4. Since we host our cloud-based dialog system on our own servers and not on the crowdsourcing website, we need to redirect crowdsourced workers to our servers, but also allow them to provide a token of task completion on the original crowdsourcing website in order to receive compensation. In order to achieve this, after configuring the task appropriately on the crowdsourcing website, we redirect them to our servers, and provide them with a code or token once they complete the specified task, which they can then enter on the crowdsourcing website as a token of successful task completion, and by extension, payment.
5. We then check and approve each workers' submission by verifying submitted tokens against database records.
6. Next, we quality check the data to see what updates/improvements to the conversational applications and system models need to be made before redeploying the task among the crowd again. This includes, but is not limited to, transcribing, annotating and rating the input data.

Lessons Learned and Recommendations

The following section enumerates some of the lessons we learned during the process of collecting crowdsourced conversational interactions.

1. Payment: This is arguably one of the most important factors to consider while setting up a crowdsourced task, as this is (not surprisingly) the primary reason why workers are willing to complete the tasks in the first place. Especially when collecting data from systems under development, we found it useful to institute a two-tier payment system: a low to moderate base pay followed by a good bonus upon *successful* completion of the application. This ensures that workers are still paid for unsuccessful attempts owing to no fault of their own (for example, if there were technical difficulties with the system) and ensures that workers are enthusiastic to continue working on future tasks. Note that when new conversational tasks are posted for the first time, the completion rate might be lower due to technical and/or task design issues. However, we recommend paying Turkers generously during this period nonetheless and maintaining email communication

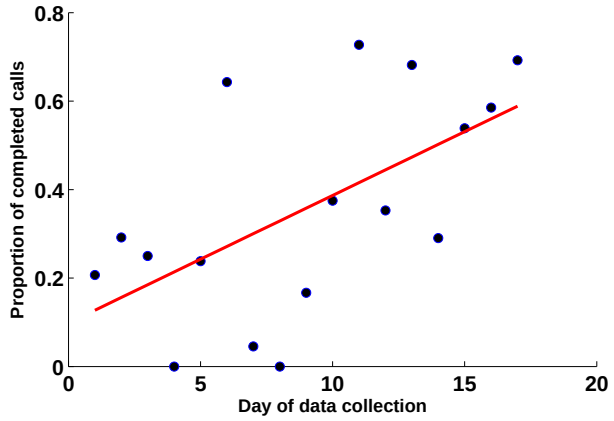

Figure 6: Completion rates over time for two applications of higher complexity (Job Hiring Interview and Pizza Service), depicted by filled circles. The 20 days depicted are chosen during the initial deployment of the applications in question, in chronological order but not necessarily consecutive (because we didn't necessarily redeploy applications every day, owing to the time required to adequately fix issues and bugs). Note the increasing trend, depicted in red. The linear regression slope was significant at the 95% level ($p \approx 0.0$), and a left-sided Wilcoxon rank sum test showed that the completion rates after the ninth day were significantly higher than those on or before ($p \approx 0.001$). This graph shows the usefulness and effectiveness of the iterative development framework, which allowed us to find and correct issues with the system (whether they were in the callflows, system code, or models) and redeploy.

regarding the cause of the failures and ongoing proposed improvements. This will boost worker morale and ensure the continued popularity of the tasks.

2. Requester Ratings: For crowdsourcing tasks on AMT, the Turkopticon platform[5] provides independent, 3^{rd}-party ratings for all requesters and tasks on AMT. These requester ratings, provided by workers, are independently moderated and allow requesters to keep track of the popularity of their tasks, and help workers more effectively choose which tasks to work on. We recommend keeping track of task feedback on Turkopticon in order to receive ongoing feedback about potential issues with the tasks along a variety of dimensions, including fairness of the task, speed and adequacy of payment, and excellence of communication with workers.
3. Qualifications: These are useful in order to control the worker population who perform tasks. For instance, to elicit a higher percentage of data from non-native speakers of English, we use geographical qualification filters to ensure that workers are not calling in from the continental United States. Such filters are not readily available on all crowdsourcing websites, but are on the bigger ones such as AMT and Microworkers.
4. Batch Sizes: We have found that it is useful to start with small batches initially to iron out kinks in the stimulus material or troubleshoot technical issues with the system, before deploying much larger batches to collect a larger sample size. Also note that larger the batch size, the more traffic is likely to hit the system within a given time window.
5. Crowdsourcing Multimodal Data: While one can collect multimodal dialog data relatively quickly and easily using crowdsourcing, video data collections introduce more challenges (and cost) relative to text or audio-only collections. For instance, in our case, with a cloud-based dialog system that can be accessed using a web browser, there were multiple technical issues that arose, including browser compatibility with the HALEF system, careful presentation of task-specific and system-specific instructions, and dealing with failures within the HALEF system. Especially when starting out with a new conversational application, the success/completion rate is likely to start out low, before the issues are identified and addressed. However, using an iterative improvement-based crowdsourcing cycle, one can rapidly improve the completion rate. Figure 6 shows an graphic example of this. In the interest of making sure we had a robust system for at least one system/browser configuration, we restricted Turkers to use the Google Chrome web browser to call into the system and follow a very specific (and detailed) set of instructions regarding how to call into the system and retry in case of unsuccessful attempts.
6. Privacy Issues: While crowdsourcing allows collection of multimodal dialog data from a large number of people at their convenience, one must be aware of the precautions that need to be taken while recording personally identifiable information from participants, the degree of which increases as we move from text to audio to video data of participants. We would also recommend clearly instructing participants that their audio and video will be recorded, along with having them sign a consent form (approved by the appropriate Institutional Review Board) that explicitly states that their data will be recorded and how it will be used, depending on the purpose of the study.
7. Participant Geography: As one can see in Figure 5, a lot of crowdsourcing providers are based out of the United States, certain countries in Europe and India. Fewer participants, nonetheless a reasonable number, are available from still other locations in Europe and South America. Certain locations like China in particular, however, are much harder to crowdsource data from owing to website restrictions and prohibitive internet firewall policies. If collecting data from such locations is essential, it might be easier to use the services of a managed crowdsourcing provider that is local to the country of interest. While this reduces flexibility, it might be more advantageous in terms of time, effort and cost.
8. Technology Infrastructure Considerations: There are several technology infrastructure factors one must consider while crowdsourcing dialog data:
   (a) First, ensure that the system is able to handle a certain number of concurrent calls (or deal with them in

[5] https://turkopticon.ucsd.edu/

a graceful manner, such as a busy message or placing them in a wait queue), and/or choose the number of participants in a batch accordingly to ensure a healthy completion rate and user experience. This might involve the use of more machines, sophisticated dialplan routing and queueing, and/or an automated on-demand spin-up of machines that dynamically adjusts based on the quantum of incoming traffic.

(b) Two, check that each different subsystem or machine in the dialog framework has adequate memory and processor configurations. For instance, the speech recognition server should have a fast processor and adequate RAM for real-time operations, and the database server should have adequate memory. This is particularly important for audiovisual data collections, where video data tend to occupy a lot of memory.

(c) Three, it is better to have one's dialog system servers co-located (or as close as possible) to the intended geographical locations of workers one would like to crowdsource from. While this is difficult for standalone dialog systems, cloud-based distributed dialog systems can avail the services offered by cloud providers such as Amazon Web Services in order to spin up servers in the target region of their choice.

9. Performance Monitoring and Alerts: Since dialog systems are typically composed of multiple subsystems such as speech recognizers, voice browsers, etc. inter-operating with each other, this also introduces multiple potential sources of system failure, especially in distributed, cloud-based systems and systems that are still in development. We found that implementing an alert system that sent out emails whenever a subsystem was down helped in increasing the completion rate, and user experience ratings. Furthermore, we also implemented a call-viewing portal and a dashboard that allows us to monitor various key performance indicators such as call completion rate, speech recognition performance, system latency, busy rate and many more. Regularly monitoring such metrics during data collections were also instrumental in improving performance.

10. Iterative Conversational Task Design: As mentioned earlier, crowdsourcing lends itself ideally to rapid and iterative development of conversational tasks, since the models and branching structure of the conversation flow from one administration, including the specific prompts and handling of different user strategies, can be updated and improved for subsequent administrations (also, see Figure 6). One strategy we have found particularly useful along these lines is to first deploy one or more administrations of a text-only (chatbot) version of the conversational task, where there is no speech elicited from the user. The user chats collected from these are then used to update language models or grammars, and used to enhance the callflow and/or design optimal questions/prompts, which can then be used to design an audio or audiovisual version of the task with either recorded prompts (by a voice talent) or synthesized voices. Deploying a text-only version of the task to refine the application before moving on to a full-blown multimodal dialog data collection is advantageous since there are fewer moving parts in the former, and therefore fewer chances of system-related bugs/issues. Moreover, this way one can also quickly get a sampling of typical user responses in order to refine the application quickly and cheaply, before engaging voice talents to record desired prompts.

Discussion and Outlook

We have presented a retrospective on crowdsourcing multimodal dialog data based on our experiences in using the HALEF multimodal dialog system to develop conversational applications primarily in the education domain. While there are many positives to take away from our experience, including the rapid and cheap method of data collection, iterative improvement cycle of models and applications, and access to participants of different ages across different geographical locations, there are other limiting factors one needs to take into account, some of which we have hinted at in earlier sections. For instance, one thing would be to ensure the dialog system can efficiently handle variable amounts of caller traffic that may arise at different times of the day by employing an automated spin-up and spin-down of cloud machines on demand. This is nontrivial given the number of subsystems that are part of the ensemble framework. The large number of subsystems and moving parts involved also makes it more challenging for newcomers to build and deploy dialog-based conversational applications, and increases the number of points of failure. Also, while unmanaged crowdsourcing allows one freedom to choose the price point and to set restrictions on who is eligible to complete the task, there are certain limitations here. The more specific and constrained the set of requirements on the target population, the more difficult it is to find the exact crowd of workers one is looking for. The reasons are many – (i) there may not be as many free-market workers signed up from a given country one wants to crowdsource from (for example, certain African countries), (ii) there might be internet firewall or other network/security issues that prevent people from either signing up to the crowdsourcing website and/or accessing the tasks in question[6] (for example, in China), (iii) the lack of interest or availability of the existing pool of workers from a particular geographical location, or in other cases, (iv) specific skill requirements on workers which thin out the existing crowd (for example, education level requirements or language ability). Such cases might warrant engaging the services of a managed crowdsourcing provider (albeit at a higher cost) in order to collect data from the specific target demographic of choice. Furthermore, during multimodal dialog data collections which involve the recording of multiple data modalities, ensuring robustness and fidelity of data collected is a challenge, as mentioned earlier. One way to address these concerns is to explicitly include tests of caller audiovisual equipment and other network or system related issues prior to the task administration. Note, however, that this will also

[6]Further, recall that in the case of multimodal dialog collections one needs to be able to transmit and receive video and audio traffic, which could cause complications.

increase the task overhead on the caller (and therefore, the time and the cost investment). Another potential solution is to standardize as many portions of the task as possible to ensure robustness.

While there is no universal solution to crowdsourcing multimodal dialog data, the field looks very promising and is already making a noticeable impact to different aspects of dialog system development. With the potential to exploit many of the rapid developments in computing hardware, software and design technology, crowdsourcing could rapidly become *the* go-to data collection solution for the collection, transcription, rating and annotation of dialog data.

Acknowledgments

We thank Nehal Sadek, Elizabeth Bredlau, Juliet Marlier, Lydia Rieck, Katie Vlasov, Ian Blood, Phallis Vaughter and other members of the ETS Assessment Development team for contributions toward the application designs as well as suggestions for system development. We also thank Yao Qian, Zydrune Mladineo, Juan Manuel Bravo, Ben Leong, Saad Khan and other current and former members of the ETS Research Team for valuable suggestions and discussion, and Robert Mundkowsky and Dmytro Galochkin for system development support.

References

Bessho, F.; Harada, T.; and Kuniyoshi, Y. 2012. Dialog system using real-time crowdsourcing and twitter large-scale corpus. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 227–231. Association for Computational Linguistics.

Chernova, S.; DePalma, N.; and Breazeal, C. 2011. Crowdsourcing real world human-robot dialog and teamwork through online multiplayer games. *AI Magazine* 32(4):100–111.

Danieli, M., and Gerbino, E. 1995. Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI spring symposium on Empirical Methods in Discourse Interpretation and Generation*, volume 16, 34–39.

DePalma, N.; Chernova, S.; and Breazeal, C. 2011. Leveraging online virtual agents to crowdsource human-robot interaction. In *Proceedings of CHI Workshop on Crowdsourcing and Human Computation*.

Eskenazi, M.; Levow, G.-A.; Meng, H.; Parent, G.; and Suendermann, D. 2013. *Crowdsourcing for speech processing: Applications to data collection, transcription and assessment*. John Wiley & Sons.

Evanini, K.; Hunter, P.; Liscombe, J.; Suendermann, D.; Dayanidhi, K.; and Pieraccini, R. 2008. Caller experience: a method for evaluating dialog systems and its automatic prediction. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, 129–132. IEEE.

Forbes-Riley, K., and Litman, D. 2011. Benefits and challenges of real-time uncertainty detection and adaptation in a spoken dialogue computer tutor. *Speech Communication* 53(9):1115–1136.

Jiang, J.; Hassan Awadallah, A.; Jones, R.; Ozertem, U.; Zitouni, I.; Gurunath Kulkarni, R.; and Khan, O. Z. 2015. Automatic online evaluation of intelligent assistants. In *Proceedings of the 24th International Conference on World Wide Web*, 506–516. International World Wide Web Conferences Steering Committee.

Jurcıcek, F.; Keizer, S.; Gašic, M.; Mairesse, F.; Thomson, B.; Yu, K.; and Young, S. 2011. Real user evaluation of spoken dialogue systems using amazon mechanical turk. In *Proceedings of INTERSPEECH*, volume 11.

Lasecki, W. S.; Kamar, E.; and Bohus, D. 2013. Conversations in the crowd: Collecting data for task-oriented dialog learning. In *First AAAI Conference on Human Computation and Crowdsourcing*.

McDuff, D.; Kaliouby, R. E.; and Picard, R. 2011. Crowdsourced Data Collection of Facial Responses. In *Proc. of the ICMI*.

McGraw, I.; Lee, C.-y.; Hetherington, I. L.; Seneff, S.; and Glass, J. 2010. Collecting voices from the cloud. In *LREC*.

Mitchell, M.; Bohus, D.; and Kamar, E. 2014. Crowdsourcing language generation templates for dialogue systems. In *Proc. of INLG*, volume 14.

Möller, S. 2004. *Quality of telephone-based spoken dialogue systems*. Springer Science & Business Media.

Parent, G., and Eskenazi, M. 2011. Speaking to the Crowd: Looking at Past Achievements in Using Crowdsourcing for Speech and Predicting Future Challenges. In *Proc. of the Interspeech*.

Pietquin, O., and Hastie, H. 2013. A survey on metrics for the evaluation of user simulations. *The knowledge engineering review* 28(01):59–73.

Ramanarayanan, V.; Suendermann-Oeft, D.; Lange, P.; Ivanov, A. V.; Evanini, K.; Yu, Z.; Tsuprun, E.; and Qian, Y. 2016a. Bootstrapping development of a cloud-based spoken dialog system in the educational domain from scratch using crowdsourced data. *ETS Research Report Series*.

Ramanarayanan, V.; Suendermann-Oeft, D.; Lange, P.; Mundkowsky, R.; Ivanou, A.; Yu, Z.; Qian, Y.; and Evanini, K. 2016b. Assembling the jigsaw: How multiple w3c standards are synergistically combined in the halef multimodal dialog system. In *Multimodal Interaction with W3C Standards: Towards Natural User Interfaces to Everything*. Springer. to appear.

Rayner, E.; Frank, I.; Chua, C.; Tsourakis, N.; and Bouillon, P. 2011. For a fistful of dollars: Using crowd-sourcing to evaluate a spoken language call application.

Suendermann, D., and Pieraccini, R. 2013. Crowdsourcing for industrial spoken dialog systems. *Crowdsourcing for Speech Processing: Applications to Data Collection, Transcription and Assessment* 280–302.

Walker, M. A.; Litman, D. J.; Kamm, C. A.; and Abella, A. 1997. Paradise: A framework for evaluating spoken dialogue agents. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, 271–280. Association for Computational Linguistics.

Walker, M.; Wright, J.; and Langkilde, I. 2000. Using natural language processing and discourse features to identify understanding errors in a spoken dialogue system. In *Proceedings of the 17th international conference on machine learning*, 1111–1118.

Yang, Z.; Levow, G.-A.; and Meng, H.-Y. 2012. Predicting user satisfaction in spoken dialog system evaluation with collaborative filtering. *Selected Topics in Signal Processing, IEEE Journal of* 6(8):971–981.

Yang, Z.; Levow, G.-A.; and Meng, H. M. 2013. Crowdsourcing for spoken dialog systems evaluation. *Crowdsourcing for Speech Processing: Applications to Data Collection, Transcription and Assessment* 217–240.

Yu, Z.; Ramanarayanan, V.; Mundkowsky, R.; Lange, P.; Ivanov, A.; Black, A. W.; and Suendermann-Oeft, D. 2016. Multimodal halef: An open-source modular web-based multimodal dialog framework.