Reducing the Cost of Dialogue System Training and Evaluation with Online, Crowd-Sourced Dialogue Data Collection

Ramesh Manuvinakurike[1], Maike Paetzel[1,2] and David DeVault[1]
[1]USC Institute for Creative Technologies, Playa Vista, CA, USA
[2]University of Hamburg, Hamburg, Germany
{manuvinakurike,devault}@ict.usc.edu, 8paetzel@informatik.uni-hamburg.de

Abstract

This paper presents and analyzes an approach to crowd-sourced spoken dialogue data collection. Our approach enables low cost collection of browser-based spoken dialogue interactions between two remote human participants (human-human condition) as well as one remote human participant and an automated dialogue system (human-agent condition). We present a case study in which 200 remote participants were recruited to participate in a fast-paced image matching game, and which included both human-human and human-agent conditions. We discuss several technical challenges encountered in achieving this crowd-sourced data collection, and analyze the costs in time and money of carrying out the study. Our results suggest the potential of crowd-sourced spoken dialogue data to lower costs and facilitate a range of research in dialogue modeling, dialogue system design, and system evaluation.

1 Introduction and Motivation

The work reported in this paper helps address a critical bottleneck in the design and evaluation of spoken dialogue systems: the availability and cost of collecting human dialogue data for a new domain. When designing, training, or testing a new dialogue system, the collection of in-domain dialogue data, either between two human roleplayers (human-human) or between a human user and a system prototype (human-agent), is both important and expensive. In-domain dialogue data is important because it provides examples of domain-specific language and interaction that serve to highlight important semantic and pragmatic phenomena in the domain, inform system design choices, and also serve as initial training data for system components such as speech recognition, language understanding, and language generation (Lasecki et al., 2013).

At the same time, the collection of this data can be expensive in terms of both time and money. Potential costs include the time needed to locate and recruit participants, the staffing overhead to schedule and coordinate visits by participants to a lab or system installation, and the payment of participation fees. As an example, for the dialogue game discussed in this paper, participants in a recent lab study were paid $15 each, and required the close supervision of a lab staff member for approximately 35 minutes per participant. These costs are substantial, especially when large amounts of data are desired for training system models based on machine learning.

Deploying dialogue systems on the web, and using crowd-sourcing to recruit remote participants, offers the possibility of increasing the availability of participants while simultaneously driving down the costs of data acquisition.

In this paper, we report on a case study in which web-based crowd-sourcing was used to carry out a substantial data collection and evaluation involving 200 remote human participants who played a fast-paced, browser-based image matching game called RDG-Image (Paetzel et al., 2014). The study included 150 participants in human-agent conditions and 50 participants in human-human conditions. By providing a substantial number of human participants at relatively low cost, the study enabled six different system versions to be compared with each other as well as to human-human teams as a baseline.

The contributions of the paper are as follows. First, we present and describe how a web-based framework for spoken dialogue data collection, called Pair Me Up (Manuvinakurike and DeVault, 2015), allows for the collection of human-agent

Proceedings of the 19th Workshop on the Semantics and Pragmatics of Dialogue, September 24-26, 2015, Gothenburg, Sweden.

spoken dialogues with remote participants. This framework had previously only been applied to human-human data collection. To our knowledge, this is the only current software framework in use by dialogue researchers that can crowdsource both human-human and human-agent dialogue data from remote web users. Second, we report and analyze our case study data collection involving 200 crowd-sourced participants. We discuss the technical challenges we encountered in achieving this data collection, and highlight issues and lessons likely to be valuable to other dialogue researchers who aim to carry out similar crowdsourced data collections. Finally, we analyze the costs in time and money of carrying out this study, and compare them to the corresponding costs associated with another similar in-lab human-human data collection.

The focus of this paper is on the research methodology of crowd-sourcing as a method of acquiring spoken dialogue data for system development and evaluation. The detailed technical design of our agent and an evaluation of its performance are presented in Paetzel et al. (2015).

We begin in Section 2 with a discussion of the RDG-Image game, which serves as the domain for this study. Section 3 discusses related work on crowd-sourced dialogue data collection. Section 4 briefly summarizes the automated agent used in this study. Section 5 presents our data collection process. Section 6 discusses technical challenges we encountered, and Section 7 presents our analysis of the costs of carrying out this study.

2 The RDG-Image Game

The RDG-Image game is a two player, dialogue-based image matching game (Paetzel et al., 2014; Manuvinakurike and DeVault, 2015). In the game, pictured in Figures 1 and 2, one person plays the role of *director* and the other is the *matcher*. Players are presented a set of eight images. The set of images is exactly the same for both players, but they are arranged in a different order on the screen. One of the images is randomly selected as a target image (TI) and it is highlighted on the director's screen with a thick red border as shown in Figure 1. The goal of the director is to give verbal clues for the TI so that the matcher is able to uniquely identify it from the distractors. Different categories are used for the image sets including pets (Figure 1), fruits, sign language (Figure 2), robots, and necklaces, among others. When the matcher believes he has correctly identified the TI, he clicks on the image and communicates this to the director who has to press a button to continue with the next TI. The team scores a point for each correct guess, with a goal to complete as many images as possible within the stipulated time for each round. Participants are incentivized to score quickly with a bonus of $0.02 per point scored. The player roles remain the same throughout the game. An example of human-human dialogue for a TI is given in Figure 2.

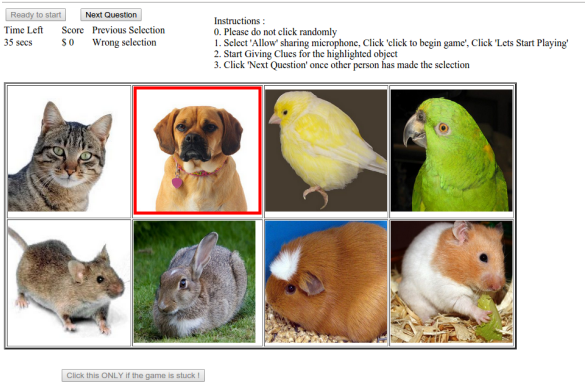

Figure 1: Web browser interface for the RDG-Image game (director's view).

3 Background and Related Work

3.1 Prior Work on Pair Me Up

This study was carried out using a software framework for web-based spoken dialogue collection called Pair Me Up (PMU) (Manuvinakurike and DeVault, 2015). The PMU framework has previously been applied to human-human data collection for the RDG-Image game, and the resulting crowd-sourced data has been analyzed in terms of audio quality, the effect of communication latency, the ability to synchronize collected audio and game events, and the perceived naturalness of remote human-human interactions (Manuvinakurike and DeVault, 2015).

The PMU architecture for human-human data collection is shown in the Figure 3. The system pairs two web users together and connects them into a shared game session where they can converse freely and interact through their browsers. PMU leverages recent developments in web technologies that support development of web-based dialogue systems. It shares this approach with recent dialogue system research such as Jiang et

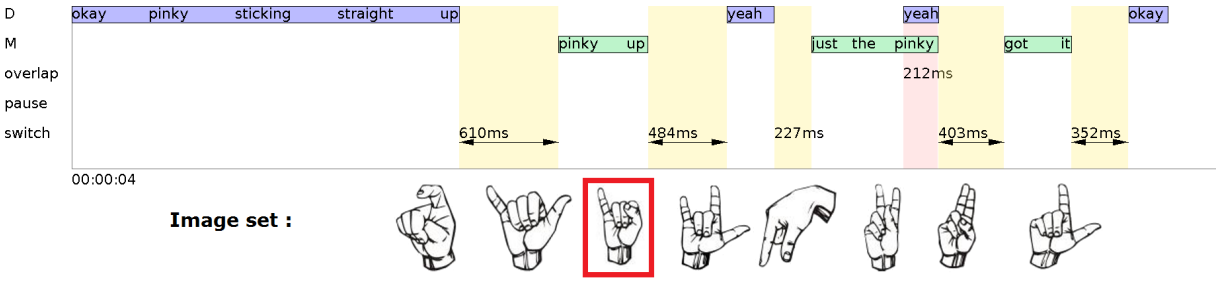

Figure 2: An example from RDG-Image: director D describes the highlighted image to matcher M.

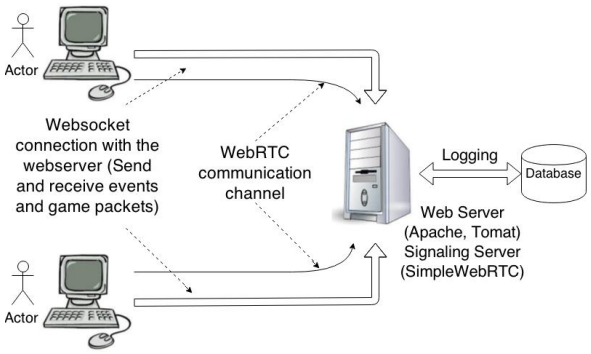

Figure 3: Pair Me Up architecture in human-human mode

al. (2014), which makes use of emerging web technologies to enable a spoken interaction between an individual remote web user and an automated dialogue system. In PMU, several of these new web technologies are used to build an interactive game where the servers can initiate events on remote client browsers, audio is streamed between two remote client browsers, and audio is captured to a server database. Two core technologies the system makes use of are websockets and webRTC. Websockets enable two way communication between the client and server, and they specifically enable the server to push events such as image set changes to the clients, and the clients to send events such as button clicks to the server, without loading a separate URL. The streaming audio communication between the remote clients uses a separate SimpleWebRTC (http://simplewebrtc.com/) channel.

3.2 Prior Work on Crowd-Sourced Dialogue Data Collection

Several large technology companies have recently deployed spoken dialogue systems reaching millions of users on mobile devices (Apple Siri, Google Now, Microsoft Cortana). Such wide deployment suggests the potential in principle for dialogue system builders to acquire large data sets to support designing, training, and evaluating their systems. In the dialogue research community, several researchers have recently taken steps toward collecting dialogue data from systems deployed on the web. Jiang et al. (2014) describe an architecture for capturing typed dialogue interactions in a human-agent configuration, with user speech optionally recognized by Google's cloud-based ASR service. Meena et al. (2014) have also been attracted to crowd-sourcing as a potential source of data, and reported a small-scale experiment in this direction. Some research applications such as Let's Go (Raux et al., 2005) as well as commercial applications (Suendermann et al., 2011; Pieraccini et al., 2009) have collected telephone-based dialogue data from large user populations. One way our work is different from this related work is that our architecture is able to collect both human-human and human-agent spoken dialogues from remote web users.

4 Summary of the agent's design

In this section, we describe the use of the PMU framework for human-agent data collection, briefly summarize the internal design of the agent, and discuss six agent versions used in the study.

4.1 Pair Me Up for human-agent data

The human-agent mode for PMU is configured in a similar way to the human-human mode, as shown in Figure 4. The user connects to the PMU server by following a URL in their browser. A websocket connection is used to transmit game events and system audio between the remote user and the PMU server. The PMU server runs both a webserver process and the automated agent, and these two communicate with each other through TCP sockets. Some modifications were required in PMU to accommodate the

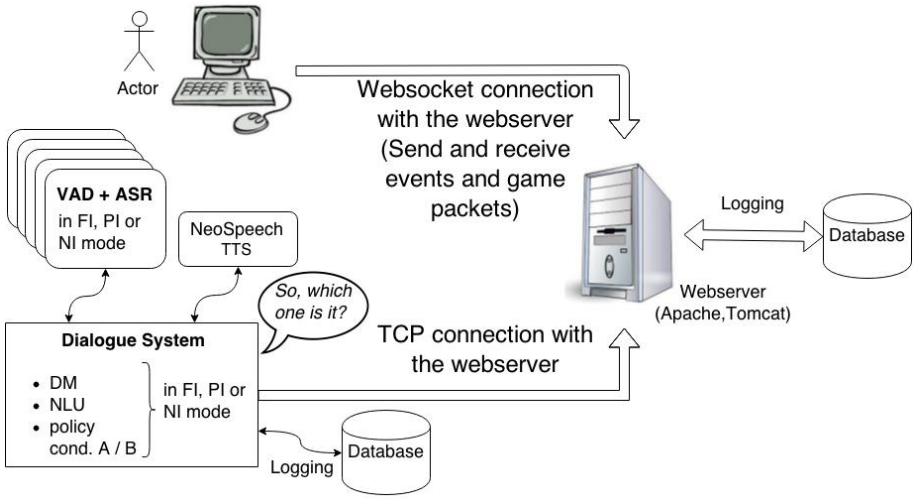

Figure 4: Pair Me Up architecture in human-agent mode

human-agent mode. In human-human mode, bidirectional audio streaming was done through SimpleWebRTC. In human-agent mode, client audio is streamed to the server using HTTP POST requests, and system audio is sent to the client using the websocket.

The agent includes internal modules for Natural Language Understanding (NLU), Dialogue Management (DM), and Dialogue Policy. The agent communicates using TCP socket connections to external processes for Voice Activity Detection (VAD), Automatic Speech Recognition (ASR), Text-To-Speech (TTS), and a database for logging.

4.2 Agent internal architecture

One main design goal for the agent architecture was to build a system which enables us to collect a large data set for multiple agent versions in a short time. On Amazon Mechanical Turk (AMT), there are certain times of the day when many people are available to participate in a study, while during work or sleep hours, among others, data collection is much slower. The peak times can be used best by enabling multiple user interactions at the same time. Thus, we designed the agent such that it can play with multiple users simultaneously, while still keeping track of the dialogue and game states for each user individually. Most agent modules operate in separate threads. This design ensures that the agent is always listening to the user speech, transforming the audio to text, taking decisions and communicating with the dialogue partner at the same time; the use of multiple threads was important to enable the agent to potentially

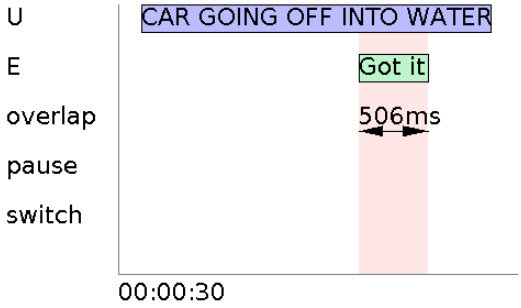

Figure 5: An example from this study: user U describes an image to agent Eve (E).

handle multiple users simultaneously with minimal latency. We now briefly summarize the various modules in the agent; see (Paetzel et al., 2015) for additional details.

VAD. Streaming audio from the user's browser is first processed by a Voice Activity Detector (VAD). Detected speech is sent to the ASR either every 100ms or at the end of each VAD segment, depending on the incrementality type (see Section 4.3).

ASR. We use a version of the Kaldi ASR system which is based on (Plátek and Jurčíček, 2014) and was specifically adapted for this study. The ASR provides support for both incremental and non-incremental speech recognition (see Section 4.3).

As audio is streamed into the VAD and ASR, the VAD and ASR both maintain an internal state for decoding the current speech segment. This means one instance of the VAD and ASR cannot serve multiple users at the same time. Thus, multiple instances of the VAD and ASR were running at

the same time, with each of them listening to a separate port, as illustrated in Figure 4. The agent takes care of the mapping between a specific user and the respective VAD+ASR instance.

NLU. For language understanding, the agent uses a data-driven statistical classifier to map either partial or final ASR results to one of the eight candidate images on the screen.

DM and Policy. In this study, the agent is always in the matcher role, and its dialogue policy uses statistically optimized rules to decide when the agent should commit to its best guess about the image being described by the user (by saying *Got it!*).

An example of the agent's gameplay is shown in Figure 5. In this example, a picture of a roadsign that warns of a hazardous driving condition is being described.

4.3 Six agent versions

The research motivation for this study is an investigation into the value of alternative types of incremental processing and incremental policy optimization in a system. To support this research, we wanted to run a data collection and evaluation involving six different versions of the agent. While other researchers might not share our specific interest in these six versions, the desire to compare several alternative system designs in an empirical way, ideally using interactive human-agent data, is common to many research efforts.

In our case, our study was designed to evaluate three versions of incrementality and two different policy optimization metrics against each other. The three incremental versions consist of the fully incremental (FI), partially incremental (PI) and non-incremental (NI) versions. Figure 6 illustrates the different versions and their modes of operation. In the FI architecture, the ASR, NLU, DM, and Policy are all operating incrementally after every additional 100 ms of user speech. This setup enables the agent to give fast-paced feedback while the dialogue partner is still talking. For the PI version, only the ASR is operating incrementally; the NLU, DM, and Policy wait for a VAD segment (inter-pausal unit) to finish before they start processing. Here, the agent cannot interrupt the user, but is still able to give a quick response once a pause is detected. In the NI architecture, the ASR, NLU, DM, and Policy are all operating on complete VAD segments as input, which increases the delay between the end of the user's speech and the beginning of the agent's response.

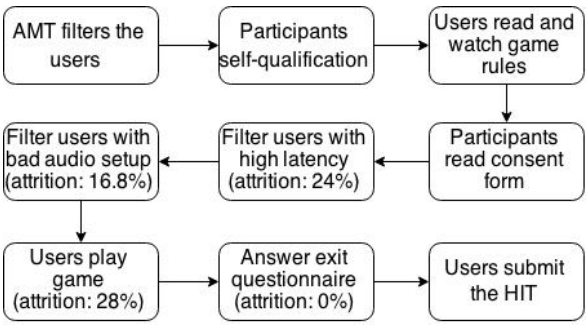

Figure 7: The HIT process during the study

Additionally, we optimized policies using two different optimization metrics, which we denote simply A and B in Figure 4. The details of the two optimization metrics are omitted; their technical rationale and motivation is beyond the scope of this paper. Together, the incrementality type and policy type variations creates a 3x2 study design, for a total of six agent versions to evaluate.

An ability to evaluate so many different agent prototypes empirically is valuable for many research questions, but it also confronts researchers with the difficulty of evaluating them with a significant number of participants in a tight timeframe and with limited financial resources.

The agent's internal modules are designed so that the agent can easily switch between different policies and incrementality types at run-time. Different versions can even be used simultaneously.

5 Crowd-Sourced Data Collection

200 native English speakers aged over 18 were recruited on Amazon Mechanical Turk (AMT) to participate in the study. 25 of them were paired with another human (25×2), and 25 played with each of the six versions of the agent (25×6). The study was conducted over a period of 10 days. Table 1 summarizes the participant demographics in the study. The study was conducted entirely over the Internet. The protocol involved in recruiting and filtering the participants to guarantee congenial data for the human-agent condition is shown in Figure 7 and discussed in the rest of this section.

AMT filters the users: AMT is able to apply certain filtering criteria for the participants. We had AMT apply the following criteria: (i) Participants have an acceptance rate equal to or greater 92% in their previous Human Intelligence Task

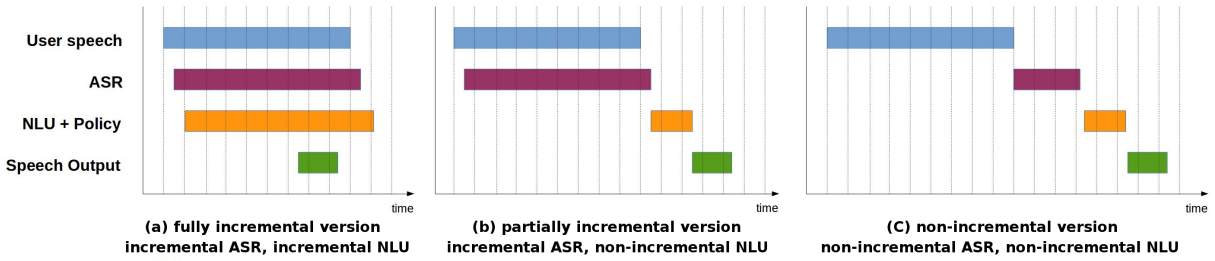

Figure 6: Three different incrementality types in our agent

(HIT) participations; (ii) previous participation in at least 50 HITs; (iii) physical location in the United States or Canada.

Participant's self qualification: The users who AMT qualified for the HIT were provided instructions to participate only if they met the following criteria: (i) must have the latest Google Chrome web browser; (ii) must be a native English speaker; (iii) must have a microphone; (iv) must not be on a mobile device; (v) must have a high speed Internet connection (5 mbps download, 2 mbps upload). Additionally, users were asked to use earbuds or headphones rather than external speakers, which helps prevent their microphone picking up sounds from their own speakers. Next, **users read and watch game rules** in text and video format. The users are then led to a consent form web page where **participants read consent form** and decide if they want to participate or not. The users enter their ID and submit their consent. To prevent certain users with problematic network latency from participating, we measure the network latency between the user and our server (see Section 6.1). 24% of the users who consented to the experiment were filtered out due to high latency or highly variable latency.

Filter users with bad audio setup: The users in the next step were made to listen to an audio file and transcribe it. If the transcriptions were wrong, the users were disqualified. This is to make sure that the users had a functioning speaker/headphone set up. The users then had to speak three pre-selected sentences in their microphone. An ASR transcribed the spoken audio and if the user had at least one word right from the sentences, the users were qualified, else disqualified. 16.8% of the users got disqualified at this step due to a "bad audio set up".

The qualified **users play the game** with the agent. 28% of the users who qualified from the previous stage did not finish playing the game with the agent. It happened that sometimes turkers closed the browser or otherwise stopped participating for reasons we could not discern. After the game, the users were made to **answer an exit questionnaire**. After answering the questionnaire the users were instructed to return to AMT and asked to **submit the HIT**.

	Agent	Human
N	150	25
Female(%)	54.7	44
Age(yrs)		
Mean	31.12	31.12
Median	28	28
SD	10.2	10.4

Table 1: Demographic data for the 175 human directors, based on whether the matcher was an agent or another Human.

6 Technical Challenges Encountered

We faced several technical challenges in achieving this data collection. The challenges can be categorized into three main headings.

6.1 Filtering out Users based on Latency

In the RDG-Image game, latency can potentially affect the collected data in several ways. For example, there can be latency between when a remote user initiates an action in their UI and when the server learns that the action occurred. Pair Me Up includes a latency-measurement protocol that allows for network latency to be monitored and adjusted for (Manuvinakurike and DeVault, 2015). It uses a variant of Network Time Protocol (Mills et al., 2010) to measure the latency. Essentially, ping-pong packets are sent continuously, with timestamps attached, to measure the round trip latency between the client and the server. In (Manuvinakurike and DeVault, 2015), a negative correlation between high mean roundtrip latency

and game score was observed. To prevent high latency from affecting this study, we generated 100 such test packets in the **filter users with high latency** step in Figure 7. We then calculated the mean and standard deviation in round trip latency. Users with mean roundtrip latency greater than 250 ms, or with a standard deviation of greater than 45 ms, were filtered out. This helps ensure that latency does not negatively affect the audio channel or gameplay with the agent.

6.2 Dealing with Effects of Variable Latency

Even with the thresholds mentioned in the previous section, transient fluctuations in network latency can sometimes occur, and we found we needed a special mechanism to ensure the integrity of the audio channel. Audio packets are recorded and sent to the PMU server from the client's browser in chunks of approximately 100 ms. Each chunk is sent separately, and is subject to variable transit time due to varying network latency from moment to moment. The order of these packets is thus not guaranteed and they can arrive out of order. For instance, if the audio packets A, B, C are recorded at times t, $t + 100$ms, $t + 200$ms respectively, it is possible for the server to receive them in order A, C, B. If not corrected, this order violation would corrupt the captured audio waveform and potentially degrade ASR and system performance. To overcome this issue, we used an auto-incrementing sequence ID that was appended to each audio packet before it left the user's browser. On the server, we monitor these sequence IDs to make sure that the audio packets either arrive in order or are reordered appropriately by the server.

6.3 Managing Server Load

Even though the agent was designed to handle multiple users at a time, we found in pilot testing that processor and memory usage by the system (agent, webserver, database, ASR) was sometimes too high to support low-latency gameplay by multiple simultaneous users on the available hardware. We therefore decided to limit the agent to one user per server to avoid this issue affecting gameplay, and deployed the system on a commercial cloud-hosting provider using six different servers. Our study could thus support up to 6 simultaneous users. Due to the high attrition rates of participants at various steps in the HIT (Figure 7), sometimes a server was left idle for the maximum HIT completion time of 40 minutes. We did not attempt to build a resource management system to enable more efficient use of our computing resources.

	Web	Lab
Participant Fees	\$1.24	\$15
Staff time per participant	2.5 min	∼35 min
Cost of Server Time	\$0.72/hour/machine	–
Participant Time	1193.1 sec	∼1800 sec

Table 2: Comparison between studies in the lab and web. Estimated numbers are indicated by ∼.

7 Analysis of Crowd-Sourced Study Cost

Table 2 shows several types of measured costs that were incurred in this web-based study (Web column). It also includes, for comparison, an estimate of what the corresponding costs would be for a lab-based human-agent study. The costs in Table 2 for running the study in the lab environment are estimated based on the human-human lab study detailed in (Paetzel et al., 2014).

Participant Fees The web users were compensated an average of \$1.24 (Max=1.56, Min=1.04, SD=0.12) (N=150) per player when interacting with the agent. In the lab, a payment of \$15 was granted for 30 minutes of participation in the human-human study. **Staff time per participant** To manage the HITs on the web required about 2.5 minutes of staff time per participant. In the lab, a staff member needs 30 minutes plus about five more minutes per participant for preparing the lab and the recording equipment.

Cost of Server Time For the 150 successful human-agent participants, the servers in this study were actually used for a total 49.71 hours. The 50 human-human participants required approximately an additional 20 hours of server time. However, due to inefficiencies in our process, during the study, the six servers were kept active for 10 days (1440 server hours). Each server hour costs \$0.72. In the lab, the hardware expenditures for a similar study would be highly dependent on the researcher's environment, but they include the cost of a computer and high-quality audio equipment (about \$800 in our lab).

Participant time The mean total gametime on the web was 275 seconds, but mean participant time was 1193.1 seconds. The additional time was spent by the users on validation steps and answering the questionnaire. In the lab, we esti-

mate that participants would need about 30 minutes for completing the study, including reading and signing the consent form, reading the game rules, playing the game, and answering the questionnaire in the end. In practice, the process takes a little more time in the lab as there is additional time needed for the staff member to greet the participant, manually start the software, adjust the microphone placement, answer any questions, etc.

Over all, it can be seen that this crowd-sourced, web-based approach to human-agent dialogue collection offers potential reductions in several types of costs, including substantial reductions in participant fees and staff time per participant.

8 Limitations

There are several limitations in the way this study was conducted. In the human-human condition, one of the major hurdles is the waiting times involved in creating pairs, which can sometimes be measured in hours (Manuvinakurike and DeVault, 2015). To try to streamline the pairing process, in pilot testing we attempted several methods. We put up a calendar scheduling system where the users could mark their availability, with time slots provided every 30 minutes. Users could avoid waiting to make a pair by selecting a time when another user had stated they were available. However, we found many turkers would select a time slot but then not show up at the specified time. Another technique we tried was a variant of a calendar where the users were paid $0.05 to mark their availability and to then show up at that time. However, again many turkers would not show up at the appointed time. We finally adopted a first-come, first-served method that paired consecutive participants, as was done in (Manuvinakurike and DeVault, 2015). Although this method was relatively slow, as individuals had to wait until a pair could be formed, and had high attrition rates, it was found to work sufficiently well to obtain 25 human-human pairs.

In the human-agent condition, the primary limitation was that there was a large amount of idle system time across our six servers (totaling to about 1370 server hours). This suggests that we had unmet capacity which could have been used to support additional dialogues, or alternatively, we could have used fewer servers to support the same number of users (thus reducing hosting costs). This idle time is related to the high attrition rates (Figure 7) and non-uniform participant presence on AMT during the times when our HITs were active. We aim to tackle these issues by optimizing our HIT and qualification processes in future work.

9 Conclusions and Future Work

In this paper we have reported on a web-based framework that helps address a critical data-collection bottleneck in the design and evaluation of spoken dialogue systems. We demonstrated the viability of our framework through a data collection study in which 200 remote participants engaged in human-human and human-agent dialogue interactions in an image matching game. We discussed several of the technical challenges we encountered and some of the limitations in our current process for collecting dialogue data over the web. In future work, we aim to address the challenge of managing available computing resources better in order to further reduce costs and accelerate data collection.

Acknowledgments

This work was supported by the National Science Foundation under Grant No. IIS-1219253 and by the U.S. Army. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views, position, or policy of the National Science Foundation or the United States Government, and no official endorsement should be inferred. The 8 images of pets used in Figure 1 are excerpted from pictures protected by copyright and released under different licenses by their original authors.[1]

[1]Thanks to Joaquim Alves Gaspar for image 1 http://commons.wikimedia.org/wiki/File:Cat_March_2010-1a.jpg and Magnus Colossus for image 3 http://commons.wikimedia.org/wiki/File:Canario_canary_p%C3%A1jaro_bird.jpg, both published under CC BY-SA 3.0. Thanks to Randy Pertiet for image 2 http://www.flickr.com/photos/34652102N04/5428922582/, Brent Moore for image 7 http://commons.wikimedia.org/wiki/File:2006_TN_State_Fair_Guinea_Pig.jpg and Domenique Godbout for image 8 https://www.flickr.com/photos/dominiquegodbout/5140544743/, all licensed under CC-BY 2.0 and to Opacha for image 4 http://commons.wikimedia.org/wiki/File:Baby–_Yellow_Naped_Amazon_Parrot_Closeup.jpg and TomiTapio for image 6 http://tomitapio.deviantart.com/art/The-bunny-says-nothing-129138755, both licenced under CC-BY 3.0. Thanks to Ilmari Karonen for image 5 https://commons.wikimedia.org/wiki/File:Mouse_white_background.jpg (Public Domain)

References

Ridong Jiang, Rafael E. Banchs, Seokhwan Kim, Kheng Hui Yeo, Arthur Niswar, and Haizhou Li. 2014. Web-based multimodal multi-domain spoken dialogue system. In *Proceedings of 5th International Workshop on Spoken Dialog Systems*.

Walter Lasecki, Ece Kamar, and Dan Bohus. 2013. Conversations in the crowd: Collecting data for task-oriented dialog learning. In *Human Computation Workshop on Scaling Speech and Language Understanding and Dialog through Crowdsourcing*.

Ramesh Manuvinakurike and David DeVault. 2015. Pair Me Up: A Web Framework for Crowd-Sourced Spoken Dialogue Collection. In *International Workshop on Spoken Dialogue Systems (IWSDS)*.

Raveesh Meena, Johan Boye, Gabriel Skantze, and Joakim Gustafson. 2014. Crowdsourcing street-level geographic information using a spoken dialogue system. In *The 15th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*.

D. Mills, J. Martin, J. Burbank, and W. Kasch. 2010. Network time protocol version 4: Protocol and algorithms specification.

Maike Paetzel, David Nicolas Racca, and David DeVault. 2014. A multimodal corpus of rapid dialogue games. In *Language Resources and Evaluation Conference (LREC)*, May.

Maike Paetzel, Ramesh Manuvinakurike, and David DeVault. 2015. "So, which one is it?" The effect of alternative incremental architectures in a high-performance game-playing agent. In *The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SigDial)*.

Roberto Pieraccini, David Suendermann, Krishna Dayanidhi, and Jackson Liscombe. 2009. Are we there yet? research in commercial spoken dialog systems. In *Text, Speech and Dialogue*, pages 3–13. Springer.

Ondřej Plátek and Filip Jurčíček. 2014. Free on-line speech recogniser based on Kaldi ASR toolkit producing word posterior lattices. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 108–112, Philadelphia, PA, U.S.A., June. Association for Computational Linguistics.

Antoine Raux, Brian Langner, Dan Bohus, Alan Black, and Maxine Eskenazi. 2005. Let's go public! taking a spoken dialog system to the real world. In *Interspeech*.

D. Suendermann, J. Liscombe, J. Bloom, G. Li, and R. Pieraccini. 2011. Large-scale experiments on data-driven design of commercial spoken dialog systems. In *Interspeech*.