Pair Me Up: A Web Framework for Crowd-Sourced Spoken Dialogue Collection

Ramesh Manuvinakurike and David DeVault

Abstract We describe and analyze a new web-based spoken dialogue data collection framework. The framework enables the capture of conversational speech from two remote users who converse with each other and play a dialogue game entirely through their web browsers. We report on the substantial improvements in the speed and cost of data capture we have observed with this crowd-sourced paradigm. We also analyze a range of data quality factors by comparing a crowd-sourced data set involving 196 remote users to a smaller but more quality controlled lab-based data set. We focus our comparison on aspects that are especially important in our spoken dialogue research, including audio quality, the effect of communication latency on the interaction, our ability to synchronize the collected data, our ability to collect examples of excellent game play, and the naturalness of the resulting interactions. This analysis illustrates some of the current trade-offs between lab-based and crowd-sourced spoken dialogue data.

1 Introduction

In recent years, dialogue system researchers have been attracted to crowd-sourcing approaches for a number of data collection tasks that support system training and evaluation. Some of the tasks that have been explored include transcription [9], capture of speech and text for training language models [3], eliciting utterance texts that correspond to specific semantic forms [11], collecting text templates for generation [6], and collecting survey-style judgments about a dialogue system's performance [12]. Crowd-sourcing and online data capture approaches have also been used to collect interactive dialogues in which a single user interacts with a live dialogue system (e.g. [4, 3, 1]).

Ramesh Manuvinakurike and David DeVault
USC Institute for Creative Technologies, Playa Vista, CA 90094, USA
e-mail: {manuvinakurike, devault}@ict.usc.edu

We present in this paper a web framework that supports crowd-sourced collection of spoken dialogue interactions between two remote participants. To the best of our knowledge, this is the first time that crowd-sourcing has been applied to the collection of spoken dialogue interactions between two remote participants in support of dialogue system research. Crowd-sourcing has been used to collect text-based chat dialogues between remote participants; see e.g. [2]. Such human-human dialogue data can be quite valuable, especially in the early stages of designing and building a dialogue system. Human-human data provides examples of domain-specific language and interaction that can inform a range of architecture and design choices in system building, as well as serving as initial training data for system components [2]. The decision to collect spoken dialogues between human interlocutors online, rather than in a controlled lab setting, is a multi-factorial one. Some of the important considerations include the introduction of browser-mediated interaction, limitations in available modalities for interaction, potential changes in demographics, data quality considerations, and the introduction of communication latency.

The research that motivates our crowd-sourced data collection involves fast-paced spoken dialogue games, in which interlocutors describe images to each other. An example interaction, drawn from the lab-based Rapid Dialogue Game (RDG) corpus we previously collected [8], is shown in Figure 1. In this excerpt, one player (the Director) tries to describe the image depicted with a red border at the top left of the screen to the other player (the Matcher), who sees the same array of eight images on his own screen but with their locations shuffled. The players are under substantial time pressure to complete as many images as they can within a fixed time limit. Natural spoken dialogue in this domain includes frequent overlapping speech (shaded in red), low latency turn-taking (as when the matcher asks *how many hands out?* and receives the answer *both hands* 215 milliseconds later), mid-utterance repairs, interruptions, acknowledgments and other low-latency responses. Capturing such rapid spoken exchanges over the internet presents a unique challenge. Particularly important factors for our dialogue system research, which aims to replicate these rapid dialogue skills in dialogue systems, include the quality of captured audio, the effect of communication latency on the interaction, the ability to collect examples of excellent game play, and naturalness of the interaction and turn-taking. In addition to describing our web framework, this paper presents a case study of how these factors differ between the lab-based corpus we previously collected and the crowd-sourced corpus we have collected with the new web framework.

2 The RDG-Image Game and Lab-Based Corpus

In the RDG-Image game [8], one person acts as a director (or "giver") and the other as a matcher (or "receiver"). Players are presented a set of eight images on separate screens. 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 giver's screen with a thick red border as

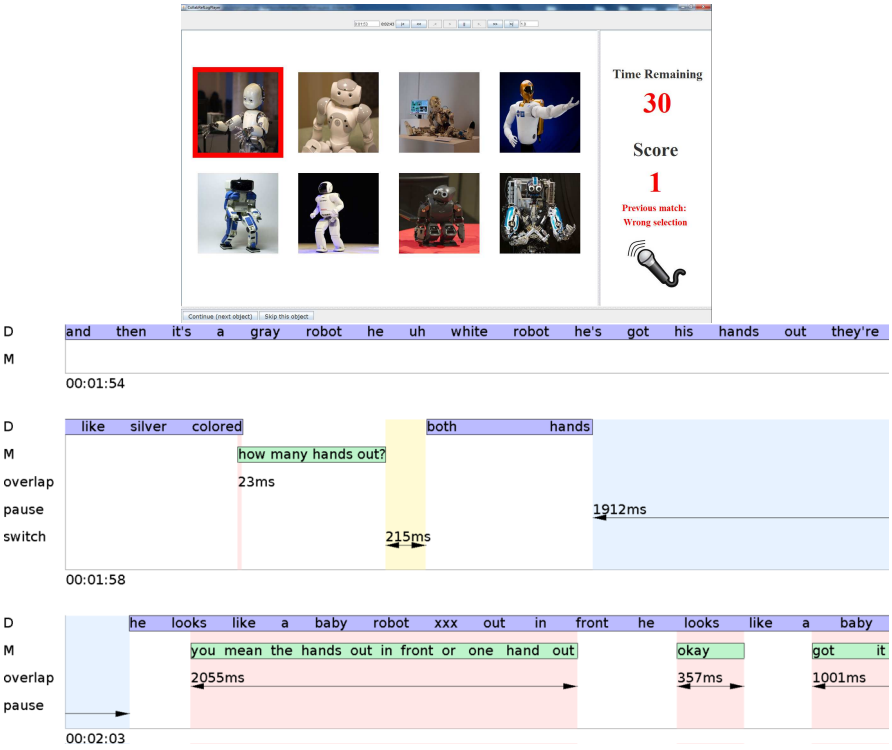


Fig. 1: An excerpt of human-human gameplay from our lab corpus. Segments of participant speech are arranged to show their temporal extents, with time increasing from left-to-right and from top-to-bottom. Speech is segmented at all silent pauses exceeding 300 milliseconds. Periods of overlapping speech are shaded in red. Periods containing silent pauses by a single continuing speaker are shaded in blue. Periods of silence at speaker switches are shaded in yellow.

shown in Figure 1. The goal of the giver is to describe the TI so that the receiver is able to uniquely identify it from the distractors. Different categories are used for the image sets including pets, fruits, people wearing make-up, robots (Figure 1), and castles, among others. When the receiver believes he has correctly identified the TI, he clicks on the image and communicates this to the giver 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 each 140 second round. Each team participates in 4 main game rounds, with roles alternating between rounds.

Our lab-based corpus includes 64 participants (32 pairs) recruited on Craigslist. Our lab-based data collection protocol was carefully designed to capture multimodal data at high fidelity. The gestures and movements of each participant were recorded individually with Microsoft Kinect cameras and multiple Logitech webcams. (Note that the giver was told to provide clues only verbally, and the role of gesture is small in this game.) Audio was also recorded for each subject individually using high-quality Sennheiser microphones and Focusrite USB audio inter-

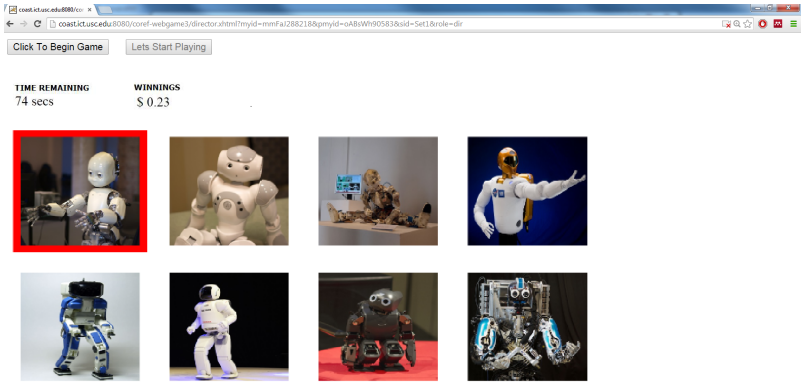


Fig. 2: The browser-based RDG-Image interface. This screenshot show's the director's browser, with the target image highlighted. The images in the matcher's browser appear in random order, and the matcher doesn't have the Next Question button. Otherwise the interface is the same.

faces. Each participant's user interface was run on a separate lab computer. As the two computers were under our direct control, we were able to synchronize their system clocks using the Network Time Protocol [5]. The two computers communicated over a gigabit ethernet connection, and all game events were logged with millisecond precision timestamps. This lab setup allowed us to synchronize all the collected game data with no observable synchronization challenges.

3 The Web-Based RDG-Image Game

To explore crowd-sourced data collection for RDG-Image, we adapted the Java-based RDG-Image user interface to a browser-based interface, shown in Figure 2. The interface is broadly similar to the lab interface (Figure 1). For the web-based game, we elected to pay a bonus to each player based on the number of correct images they achieve together within the time limit. To emphasize the monetary incentive, we display their score in terms of this bonus (marked "WINNINGS" in Figure 2). The score is thus denominated in US Dollars rather than in points.

3.1 The Pair Me Up Web Framework

Pair Me Up is a software framework that supports web-based collection of spoken human-human dialogues between remote participants. It pairs consecutive web users together and connects them into a shared game session where they can con-

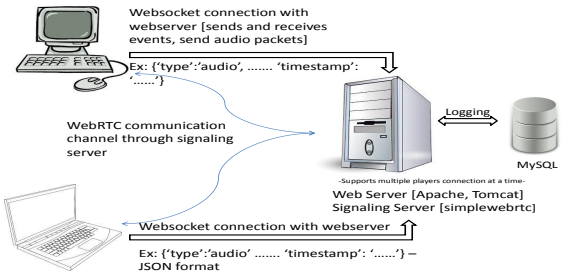


Fig. 3: Architecture of the Pair Me Up system.

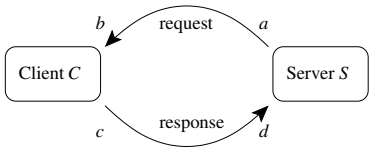
verse freely and interact through their browsers. Pair Me Up leverages recent developments in web technologies that support development of web-based dialogue systems. It shares this approach with recent dialogue system research such as [1], which makes use of emerging web technologies to enable a spoken interaction between an individual remote web user and an automated dialogue system. In Pair Me Up, we use several of these new web technologies 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.

The architecture of Pair Me Up is shown in Figure 3. Two core technologies the system makes use of are websockets and webRTC [10]. 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 audio and game events such as button clicks to the server, without loading a separate URL. The streaming audio communication between the remote clients is currently set up using a separate SimpleWebRTC (http://simplewebrtc.com/) channel. The video channel is disabled for the current study due to bandwidth limitations observed in pilot testing and the fact that RDG-Image players primarily look at the images being described rather than each other.

3.2 Latency Measurement Protocol and Data Synchronization

In a lab-based study, network latency between machines can be minimized through use of high-speed LAN connections, and computer clocks can be synchronized using a method such as the Network Time Protocol [5]. In a crowd-sourced data collection, network latency may be both higher and also harder to control. Additionally, security considerations rule out adjusting a remote user's system clock.

In our web-based game interface, latency can potentially affect the data we collect in several ways. There can be latency between when a remote user initiates an action in their UI and when the server learns that the action occurred, for example. Conversely, if the server initiates an event in the user's UI (e.g. changing the im-



Event	Known time of event on S	Known time of event on C
a : Server S sends request	t_S^a	
b : Client C receives request		t_C^b
c : Client C sends response		t_C^c
d : Server S receives response	t_S^d	

Fig. 4: Latency measurement protocol

age set), this event may not actually occur in the user's UI until some time later. Given the sensitivity of our research to having accurate timing of dialogue events, we implemented a simple latency measurement and synchronization protocol that allows us to (1) estimate the network latency between each client and the server, and (2) translate between timestamps collected from client machine system clocks and timestamps on our server.

Like the Network Time Protocol, our approach relies on the transmission of a series of request/response packets between the server and client machines. The protocol is illustrated in Figure 4. At the beginning of each image set in the game, a request packet is sent from the server S to the remote client C. We denote the server's timestamp when this request is sent by t_S^a, using a subscript for the machine (S or C) whose clock generates the timestamp, and a superscript (a, b, c, or d) for the four sequential events that occur as the request and response are sent and received by each machine. As part of the exchange, Pair Me Up code running in client C's browser computes a client system timestamp t_C^c and immediately sends this value with its response back to the server. The server receives the response at t_S^d. With each request/response cycle, the server therefore has a measure of the round trip latency of server-client communication: roundtrip $= t_S^d - t_S^a$. Over the course of a game this request/response cycle happens in the background many times between the server and each of the remote clients. In order to relate client event timestamps to server event timestamps, we adopt the assumption that the client initiated its response at the midpoint of the server's observed roundtrip time:

$$t_S^c = \frac{1}{2}(t_S^a + t_S^d)$$

This provides us with a series of timestamp pairs, t_C^c and t_S^c, for the same event expressed on the client and system clocks. We then use a linear regression to estimate a translation between any arbitrary client timestamp t_C^e for event e and the corresponding server timestamp t_S^e:

$$t_S^e = w_1 \cdot t_C^e + w_2 \tag{1}$$

Of course, this translation can also be used bidirectionally to translate server timestamps into corresponding client timestamps. To enable approximate synchronization of all the collected data, all events originating on the two remote clients (including user mouse clicks and image selections, button presses, page request times, connection to partner, and audio chunk capture) are logged into the database with the associated client timestamps. Events originating on the server (including image set changes, countdown timer events, and score changes) are logged into the database with the associated server timestamps. All data and events are later approximately synchronized by translating all events onto a common timeline using equation 1. We can reconstruct all the events on the server timeline or user timeline as desired. One limitation of our current approach is that network latency is not completely constant, and thus a dynamic translation might achieve a more accurate synchronization.

4 Crowd-Sourced Data Set

We recruited 196 individuals from Amazon Mechanical Turk (AMT) to participate in the web-based RDG-Image game. The requirements to participate in the HIT were: (i) a high speed internet connection (5 mbps download, 2 mbps upload); ii) the latest Google Chrome web browser; (iii) task acceptance of $\geq 92\%$; (iv) previous participation in at least 50 HITs; (iv) physical location in the United States; (v) must be a native English speaker; (vi) must have a microphone; and (vii) must not be on a mobile device. As part of self-qualifying themselves, Turkers verified their internet connection speed using the speedtest.net web service. Additionally, although this was not a strict requirement, they were strongly encouraged to listen to their computer's sound output using headphones rather than speakers. This instruction was added after pilot testing, to help reduce audio quality issues related to echo.[1] After self-qualifying for the HIT, users proceeded to the instructions, which were provided in both text and video format. The instruction video explained the interface in detail. Users then followed a link and waited until they were paired up with another Turker as a game partner. Access to each Turker's microphone and speakers was then requested from the users. The users then made sure their audio connection worked well. Before playing the game, they were shown a leaderboard where they could see how prior teams performed. After the game, they returned to the AMT site for a post-game questionnaire.

During the data collection, pairing of participants happened on a 'first come, first served' basis. Pair Me Up simply connected each player to the next player who reached the same stage in the HIT, without any scheduling. To attract users, we posted a number of HITs and waited until two consecutive Turkers could be paired up to play the game. Our Pair Me Up server is currently able to support at least 12 simultaneous players (6 simultaneous games). We observed that this approach worked well provided that a sufficient number of HITs was made available on Mechanical

[1] When the users listen through speakers, it often happens that one of their microphones picks up the speech output of their partner, and echo ensues. We currently do not attempt to cancel this echo.

Turk. However, we avoided posting too many HITs at once to prevent exceeding our server's capacity. When too few HITs were available, waiting times for a partner increased.

5 Results

5.1 Data Collection Throughput

In total our web-based data collection took place over 17 days, and included 177 hours of aggregate HIT time by 196 Turkers. We expect each HIT to take a minimum of about 15 minutes to complete, including reading instructions, 9 minutes and 20 seconds of actual gameplay, and the post-game questionnaire. The median time was nearly 38 minutes, which is about the same amount of time it took for the participants in the lab to complete the RDG-Image game and fill out all questionnaires. Most of the time spent by our web study participants was spent waiting for a partner. In future work we would like to reduce this wait time by pairing up partners more efficiently. The main bottleneck to parallel game collection on our server is the actual live gameplay, which requires transmission and logging of speech streams. Because our server can support at least 6 simultaneous live games, and the actual dialogue gameplay requires only 9 minutes and 20 seconds per pair, this translates into a potential data collection throughput for the Pair Me Up framework on a single server of hundreds of spoken dialogue games per day. In comparison, our lab-based data collection, which yielded 32 subject pairs, took about a month to orchestrate and complete, due largely to the overhead of individual subject recruitment and scheduling, as well as the impossibility of parallelism given lab resources.

5.2 Audio Quality

In our lab-based corpus, audio was captured using high-quality microphones and audio hardware, which were calibrated and adjusted by lab staff for each participant. Additionally, our lab is generally a low noise environment that is free of most distractions. By contrast, in our web audio data, we have very little control over the participants' audio hardware and ambient environments. We observed captured audio to include a wide range of background noises, including televisions, cats meowing, dogs barking, and mobile phones ringing, among other distractions. Our primary use for this audio is through transcription and automatic speech recognition (ASR), in support of dialogue system research and development. We therefore assess audio quality by way of its suitability for these purposes. We currently have transcribed a subset of the web-based speech amounting to several hundred utterances. For this subset, despite the variable audio conditions, we have encountered

no difficulties in transcribing the speech. To assess the audio quality in relation to ASR, we selected a random sample of 105 utterances each from the web corpus and the lab corpus. As part of transcription, these utterances were segmented from surrounding speech (using silence regions exceeding 300ms) and manually transcribed. We then evaluated ASR word error rate for both samples using Google's ASR (https://www.google.com/speech-api/v2/recognize), a broad-coverage cloud-based industry speech recognizer which we have observed to have competitive performance in recent ASR evaluations for dialogue systems at our institute [7]. In our corpora, the observed word error rate (WER) of 24.10 in ASR for web-based audio is significantly higher (W=4647.5, p-value=0.04285, Wilcoxon rank sum test) than the WER of 19.83 for lab-based audio. This increase in WER of 4.27 for web-based audio provides perspective on the trade-offs between controlled lab-based audio capture and crowd-sourced online audio capture for dialogue system researchers.

5.3 Effect of Latency on Game Performance and Synchronization

We summarize the network latency for each user using the round trip time observed in the latency measurement protocol described in Section 3.2. Higher values indicate higher network latency that could adversely impact various aspects of gameplay, for example UI responsiveness to user button clicks as well as the speech channel. We observed a mean roundtrip latency of 136.9ms (median 108.0ms, standard deviation 84.9ms, min 29.0ms, max 464.0ms). To understand how latency affects overall game performance, we investigated the relationship between roundtrip latency and score (number of correct images). We observed a slight weak, but significant, negative correlation between latency and score ($r = -0.16, p < 0.05$). Upon closer examination, the negative effect of latency on score seems to be limited to those players with relatively high latency. We find no significant correlation between score and latency for players whose latency is below 250ms ($r = -0.06, p = 0.44$). Comparing the population of low latency players (latency $<=$ 250ms, $N = 177$, mean score 50.7) to high latency players (latency $>$ 250ms, $N = 19$, mean score 40.5), we observe a significant difference in scores ($p < 0.05$, Wilcoxon rank-sum test). We interpret these data as suggesting that if latency is low enough, its effect on game score is negligible. Additionally, we used our latency measurement and synchronization protocol to construct more than 20 synchronized videos that combine the two users' speech streams with a recreation of each user's UI state at each moment (including images observed, button clicks, etc.). If timeline correction using Equation 1 is not performed, such videos exhibit numerous clear synchronization problems. After timeline correction, we have found that the combined videos appear remarkably well synchronized. Upon observation of these videos, we are unable to detect any remaining latency or synchronization issues, and we view the data as sufficiently synchronized for our research purposes. We would like to further investigate the exact accuracy of data synchronization achieved in future work.

	Web	Lab
N	196	64
Average Pay per player	$1.915	$15
Scores(%) [Mean, SD, Min, Max, Median]	49.8, 13.1, 22, 78, 51	45, 13.0, 20, 68, 44
Age(%) [Mean, SD, Min, Max, Median]	31.3, 8.2, 19, 68, 29	36.6, 12.7, 18, 60, 34.5
Gender(%) [Female, Male] (%)	53.3, 46.7	55, 45

Table 1: Cost, scores attained, and demographic data for our web and lab studies.

5.4 Cost, Gameplay Quality, and Naturalness of Interaction

We summarize the study cost, scores attained, and basic demographic data for our two corpora in Table 1. From Table 1 we can see that the web-study data is 7.8x less expensive per participant to collect (once the Pair Me Up infrastructure is in place). In terms of acquiring examples of excellent gameplay, which is one of our research requirements, we found that our web-study players scored significantly higher than the players in lab ($W = 5389$, $p = 0.01875$, Wilcoxon rank sum test). The full explanation for this difference is unclear as there were several differences between the web study and the lab study. One difference is that web-study participants were incentivized with a bonus payment per correct image, while lab study participants were paid a flat rate of $15 for participation. Demographic differences between Turkers and Los Angeles area Craigslist users may also have played a role; for example, our web-study participants were younger on average. In any case, we conclude that it is possible to collect examples of excellent gameplay for RDG-Image with a crowd-sourced data collection. All participants filled out post-game subjective questionnaires, providing answers on a 5-point Likert scale. We were especially interested in the perceived naturalness of the interaction and the usability of the interface, and we present several observations in Figure 5. All significance tests are Wilcoxon rank sum tests.

Web-study participants gave significantly higher ratings of the user interface being intuitive and easy to use (Q1). They also gave higher ratings to the ease of understanding the game rules (Q2) and it being easy to play the game with their partner (Q5). These findings may be partially explained by the more detailed instructions we provided for web users about the browser interface, including the addition of video-based instructions. Demographic differences and possible comfort in using a browser-based interface could potentially play a role as well. In terms of naturalness of the interaction, the results were also favorable for the web-based study. Despite our concern about network latency affecting interaction naturalness, we observed no significant difference in ratings of the speed and flow of communication between the web study and the lab study (Q6). In fact, web-study participants gave significantly higher ratings to it being easy to play the game with their partner (Q5), satisfaction with their score (Q4), and a rating of whether they spoke the way they normally do with the partner they were paired with (Q3). The fact that web-study participants scored higher than lab-study participants may play a role in the perceived ease of playing with their partner and score satisfaction.

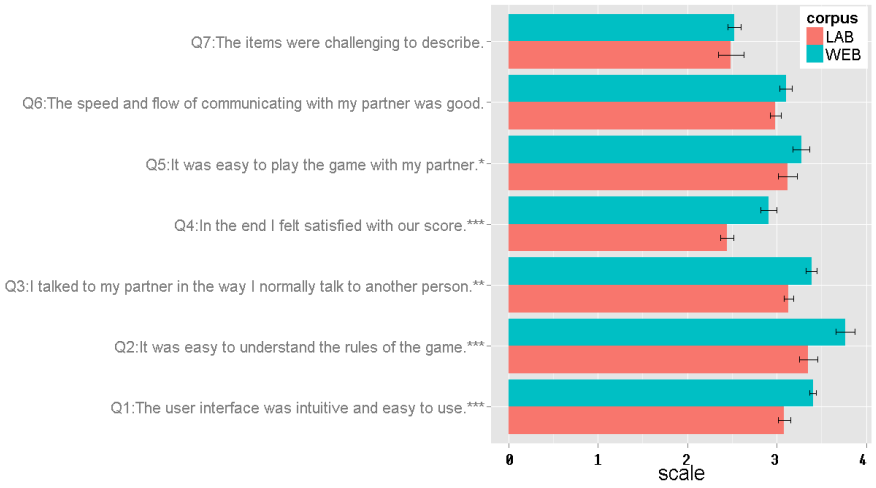


Fig. 5: Subjective questionnaire results for questions related to interaction naturalness and usability of user interface. Means and standard errors are shown for all questions. * indicates $p < 0.05$, ** indicates $p < 0.01$, *** indicates $p < 0.001$)

6 Conclusions

We have presented a web framework called Pair Me Up that enables spoken dialogue interactions between remote users to be collected through crowd-sourcing. We have confirmed, for spoken dialogue interactions in the RDG-Image game, the commonly observed pattern that crowd-sourced data collection over the web can be faster and much less expensive than data collection in the lab. At the same time, we have explored several trade-offs in web-based vs. lab-based data collection for dialogue system research. In terms of audio quality, we have found an increase of about 4% in ASR word error rate for web-based audio data. Such an increase may be acceptable by many researchers in exchange for easier data collection. In terms of network latency, we have found that while it is an important consideration, it does not rule out natural real-time dialogue between the remote participants, and that data can still be synchronized sufficiently for our purposes using a straightforward latency measurement protocol. We have observed that the quality of gameplay, as determined by scores achieved and several subjective assessments by Turkers, was higher for our crowd-sourced study than in the lab.

7 Acknowledgments

We thank Maike Paetzel. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1219253. Any opinions, findings,

and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. For the images in Figures 1 and 2, we thank (numbering 1-8 from left-right, top-bottom): [1,2] Jiuguang Wang (CC BY-SA 2.0), [3] Alex Haeling (CC BY 2.0), [4] NASA, [5] Joe Wu (CC BY-NC-SA 2.0), [6] RoboCup2013 (CC BY-NC-SA 2.0), [7] Waag Society (CC BY 2.0), & [8] Janne Moren (CC BY-NC-SA 2.0).[2]

References

1. Jiang, R., Banchs, R.E., Kim, S., Yeo, K.H., Niswar, A., Li, H.: Web-based multimodal multi-domain spoken dialogue system. In: Proceedings of 5th International Workshop on Spoken Dialog Systems (2014)
2. Lasecki, W., Kamar, E., Bohus, D.: 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 (2013)
3. Liu, S., Seneff, S., Glass, J.: A collective data generation method for speech language models. In: Spoken Language Technologies Workshop (SLT) (2010)
4. Meena, R., Boye, J., Skantze, G., Gustafson, J.: Crowdsourcing street-level geographic information using a spoken dialogue system. In: The 15th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL) (2014)
5. Mills, D., Martin, J., Burbank, J., Kasch, W.: Network time protocol version 4: Protocol and algorithms specification (2010). URL http://www.ietf.org/rfc/rfc5905.txt
6. Mitchell, M., Bohus, D., Kamar, E.: Crowdsourcing language generation templates for dialogue systems. In: the 8th International Natural Language Generation Conference (INLG) (2014)
7. Morbini, F., Audhkhasi, K., Sagae, K., Artstein, R., Can, D., Georgiou, P., Narayanan, S., Leuski, A., Traum, D.: Which ASR should I choose for my dialogue system? In: Proceedings of the SIGDIAL 2013 Conference. Metz, France (2013)
8. Paetzel, M., Racca, D.N., DeVault, D.: A multimodal corpus of rapid dialogue games. In: Language Resources and Evaluation Conference (LREC) (2014)
9. Parent, G., Eskenazi, M.: Toward better crowdsourced transcription: transcription of a year of the let's go bus information system data. In: IEEE Workshop on Spoken Language Technology (2010)
10. Vogt, C., Werner, M.J., Schmidt, T.C.: Leveraging webrtc for p2p content distribution in web browsers. In: ICNP, pp. 1–2 (2013)
11. Wang, W., Bohus, D., Kamar, E., Horvitz, E.: Crowdsourcing the acquisition of natural language corpora: Methods and observations. In: Spoken Language Technology Workshop (SLT), pp. 73–78 (2012)
12. Yang, Z., Li, B., Zhu, Y., King, I., Levow, G., Meng, H.: Collection of user judgments on spoken dialog system with crowdsourcing. In: Spoken Language Technologies Workshop (SLT) (2010)

[2] [1] http://www.flickr.com/photos/jiuguangw/4981810943/ [2] http://www.flickr.com/photos/jiuguangw/4982411246/ [3] http://www.flickr.com/photos/alexhealing/2841176750/ [5] http://www.flickr.com/photos/ozzywu1974/325574892/ [6] http://www.flickr.com/photos/robocup2013/9154156312/ [7] http://www.flickr.com/photos/waagsociety/8463802099/ [8] http://www.flickr.com/photos/jannem/1885853738/