CROWDSOURCING THE ACQUISITION OF NATURAL LANGUAGE CORPORA: METHODS AND OBSERVATIONS

William Yang Wang, Dan Bohus, Ece Kamar and Eric Horvitz

Microsoft Research, Redmond, WA, 98052, U.S.A.
ww@cmu.edu, {dbohus, eckamar, horvitz}@microsoft.com

ABSTRACT

We study the opportunity for using crowdsourcing methods to acquire language corpora for use in natural language processing systems. Specifically, we empirically investigate three methods for eliciting natural language sentences that correspond to a given semantic form. The methods convey frame semantics to crowd workers by means of sentences, scenarios, and list-based descriptions. We discuss various performance measures of the crowdsourcing process, and analyze the semantic correctness, naturalness, and biases of the collected language. We highlight research challenges and directions in applying these methods to acquire corpora for natural language processing applications.

Index Terms— crowdsourcing, natural language elicitation methods, language understanding, spoken dialog.

1. INTRODUCTION

Interactive technologies with natural language input and output must capture the variation in language usage associated with specific users' intentions. For example, data-driven spoken language understanding systems rely on corpora of natural language utterances and their mapping to the corresponding semantic forms. Similarly, data-driven approaches to natural language generation use corpora that map semantic templates to multiple lexical realizations of that template. Multilingual processing tasks such as machine translation rely on the availability of target language sentences in a parallel corpus that capture multiple valid translations of a given sentence in the source language.

Developing or extending the recognition prowess of an existing interactive natural language system can be a challenging task. In the initial stages, a deployed system is required to collect natural language data from users. same However, data is required to build the initial system. System developers must often create initial grammars and prompts, either manually or based on small-scale wizard-of-Oz studies. Once an initial version of a system is deployed, data is collected, transcribed and labeled, and the language models and grammars are updated. This approach suffers from several drawbacks. The initial grammars might not generalize well to real users, and poor system performance in the initial stages can subsequently bias the users' input and the collected data. The development lifecycle can have high costs, and refining the system's performance can take a long time. Also, the systems face adoption difficulties in the early stages, because of limited functionality and lack of robustness. Moreover, every time new functionality is added to an existing system, developers are faced with the challenge of building or acquiring new language resources and expanding grammars.

We investigate the use of crowdsourcing methods for collecting natural language corpora. Previous work with crowdsourcing in language technologies has focused largely on transcription, search relevance, speech rating, and collecting read speech. In this work, we focus on the problem of collecting natural language expressions that correspond to a given semantic form. We investigate an approach for this *structured natural language elicitation* problem using crowdsourcing. The approach aims to harvest at low cost a language corpus that reflects the natural variation of human-generated language.

We investigate three alternative structured language elicitation methods: we present sentences, scenarios, or list-based descriptions to a crowd and ask workers to rephrase the language in their own words. We discuss several performance measures of this crowdsourcing process, we analyze of the semantic correctness and naturalness of the collected language, and we discuss some of the biases these methods might create. We highlight several lessons learned and outline directions for future work.

2. RELATED WORK

Crowdsourcing methods have attracted considerable attention because of assumed efficiencies of collecting data and solving tasks via programmatic access to human talent. In the realm of language technologies, crowdsourcing has been used for speech transcription [1], system evaluation [2], read speech acquisition [3], search relevance [4], translation [5], and most recently, paraphrase generation [6, 7]. We shall introduce and address the problem of crowdsourcing language that corresponds to a given semantic form. Some of the methods we use bear similarities to previous work in paraphrase generation. While paraphrase generation seeks mappings between surface-level realizations of language without knowledge of the underlying semantics, we focus on capturing the mapping from semantic to lexical forms. To the best of our knowledge, this is the first report on the use of crowdsourcing to address this problem.

Our language acquisition task aligns with challenges in the domain of spoken dialog systems. While authoring spoken dialog systems is a labor-intensive process that requires specific domain knowledge [8], many spoken dialog systems (e.g., [9, 10, 11]), still rely on developers to author predefined grammars and plausible language rules. An alternative is to use transcriptions of human to human conversations [12] or data collected in wizard-of-Oz studies [13], but it is generally difficult and expensive to collect data in these ways. Crowdsourcing of grammars holds promise for alleviating the "cold-start" problem seen at the outset of the fielding of a system by enabling access to natural language data, and helping developers to extend functionality incrementally. Furthermore, the methods we describe may also be useful for other tasks, such as generating templates for the natural language generation module of spoken dialog systems.

3. ELICITATION METHODS

We now present several methods for eliciting natural language data for given semantic forms via crowdsourcing. Consider a request for finding a Chinese restaurant for dinner in Seattle. Semantically, this request might be captured by a frame with slots and values, such as FindRestaurant(City=Seattle; Cuisine=Chinese). Lexically it may be expressed in a variety of ways, for instance: *Could you please find a Chinese restaurant in Seattle?* or *I'm looking for a Chinese restaurant in Seattle*., etc. Our goal is to convey the information captured by the semantic frame to the crowd worker in a manner that prompts them to produce corresponding natural language. By repeating this task with multiple workers, we seek to harvest the natural language usage that corresponds to the given semantic form.

We propose and investigate three different methods for conveying the frame semantics to the crowd worker:

- In the *sentence-based method* we present a corresponding natural language sentence, e.g. "*Find a Seattle restaurant that serves Chinese food*."
- In the *scenario-based method* we adopt a story-telling scheme that presents multiple sentences that form a scenario with a specific goal, e.g. "*The goal is to find a restaurant*. *The city is Seattle*. *You want to have Chinese food*."
- For the *list-based method,* we present a specific goal, and a set of items corresponding to the slots and values in the form of a list. For instance:

  Goal: Find restaurant
  City: Seattle
  Cuisine type: Chinese

In each case, we ask the worker to construct a single sentence, in their own words, that captures all the given information.

In the next section, we study different characteristics of these elicitation methods with experiments performed on a crowdsourcing platform. First, we study empirically whether crowd workers can perform this type of task efficiently and correctly; the elicitation methods are successful only in as much as the language collected corresponds to the given semantic form, *i.e.,* no information is omitted or added. Second, the elicitation methods rely on specific templates authored by system developers (sentence, scenario, or list) for every semantic frame. Since the aim is to elicit the natural language variation that corresponds to a given semantic form, we seek to understand how much the templates affect the language data produced by the crowd. Do the methods create systematic biases, and are there significant differences among methods regarding the biases they create?

4. EXPERIMENTS

4.1. Setup

The experiments discussed below were designed to investigate whether we can elicit natural language for a set of semantic frames, based on the methods described above. Specifically, we create an ontology consisting of nine semantic frames that cover different information seeking domains (shown in Figure 1). The choice of frames is informed by previous work in the dialog community and by existing corpora, so that lessons learned from these experiments generalize to typical dialog domains.

The proposed elicitation methods take as input instantiated semantic frames. We illustrate the process we use for instantiating semantic frames with an example in Figure 1. For each of the nine frame types in the ontology, we create instances of that type that have none, one, or more slots instantiated. For example, the frame-type ReserveRoom can take up to three slots in our ontology (NumPeople, Duration, Day – see Figure 1). We create instances for this frame with all possible subsets of these slots instantiated. In addition, since the order of slots could bias the crowd workers, for each instantiated frame we consider all possible slot orderings. The resulting frames are displayed in the middle column of Figure 1.

The proposed elicitation methods present an instantiated frame to workers by means of a corresponding sentence, scenario, or list template. For experiments reported in this paper, these templates were authored by three co-authors of the paper. For the sentence case, a separate template was generated for each frame and slot order. For the list and scenario cases, we only generate the sub-templates for each slot, and we concatenate these sub-templates according to the slot order. The templates are then instantiated by randomly sampling slot values from a predefined list of possible values for each slot. The authoring effort involved

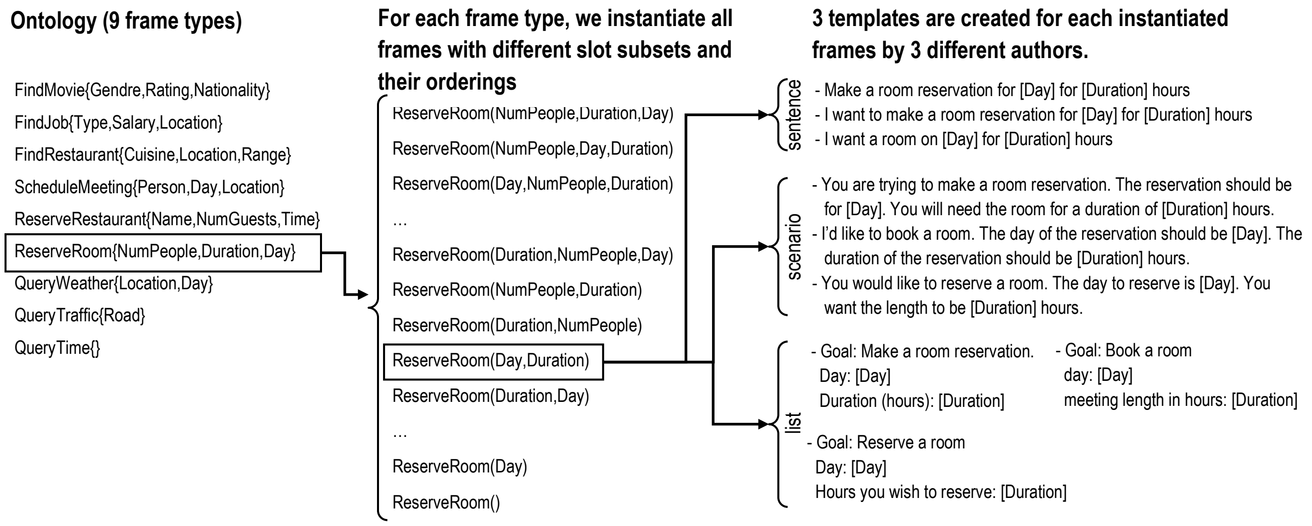
Ontology (9 frame types)
FindMovie{Gendre,Rating,Nationality}
FindJob{Type,Salary,Location}
FindRestaurant{Cuisine,Location,Range}
ScheduleMeeting{Person,Day,Location}
ReserveRestaurant{Name,NumGuests,Time}
ReserveRoom{NumPeople,Duration,Day}
QueryWeather{Location,Day}
QueryTraffic{Road}
QueryTime{}
For each frame type, we instantiate all frames with different slot subsets and their orderings
ReserveRoom(NumPeople,Duration,Day)
ReserveRoom(NumPeople,Day,Duration)
ReserveRoom(Day,NumPeople,Duration)
…
ReserveRoom(Duration,NumPeople,Day)
ReserveRoom(NumPeople,Duration)
ReserveRoom(Duration,NumPeople)
ReserveRoom(Day,Duration)
ReserveRoom(Duration,Day)
…
ReserveRoom(Day)
ReserveRoom()
3 templates are created for each instantiated frames by 3 different authors.
sentence
- Make a room reservation for [Day] for [Duration] hours
- I want to make a room reservation for [Day] for [Duration] hours
- I want a room on [Day] for [Duration] hours
scenario
- You are trying to make a room reservation. The reservation should be for [Day]. You will need the room for a duration of [Duration] hours.
- I'd like to book a room. The day of the reservation should be [Day]. The duration of the reservation should be [Duration] hours.
- You would like to reserve a room. The day to reserve is [Day]. You want the length to be [Duration] hours.
list
- Goal: Make a room reservation. Day: [Day] Duration (hours): [Duration]
- Goal: Book a room day: [Day] meeting length in hours: [Duration]
- Goal: Reserve a room Day: [Day] Hours you wish to reserve: [Duration]

Figure 1. Methodology for generating instantiated frames.

is thus larger for the sentence-based method than for the scenario and list methods.

As we seek to acquire natural language for use in an interactive system, we explicitly asked crowd workers to "imagine talking to an automated assistant in a natural manner." For example, the guideline for the list method states: *We'd like to find out how you would naturally request assistance from the assistant. We'll provide you with a list and specify a goal that the assistant can help you with and we'd like you to type the words that you'd say to make the request*. The instructions and user interface design were minimally modified to accommodate the differences between methods.

We used Microsoft's Universal Human Relevance System (UHRS) crowdsourcing platform. Similar to other crowdsourcing platforms, such as Amazon Mechanical Turk, UHRS is a marketplace that connects a large worker pool from different countries with human intelligence tasks. UHRS workers are hired, qualified, and managed by third-party vendors. For the American English market, UHRS provides access to thousands of unique workers. Previous experiments on this system have shown that it can gather high-quality responses from workers with low task latency.

4.2. Crowdsourcing Experiments

In an initial experiment, we created 10 tasks for each instantiated frame, template and method by sampling values for slots. We required that each task be performed by a single judge. Due to the constraints of the UHRS system, we were unable to set the maximum number of tasks to be performed by each judge. As a result, in many cases, many of the 10 tasks for each instantiated frame were performed by the same judge, which led to many repeated sentences for the same semantic form. Furthermore, while we posted the three methods nearly simultaneously, they were addressed largely in the order of appearance on the market. Thus, the tasks for each method ended up being performed at different times of the day and night. As a consequence, the workload distribution was different across the methods: there were few judges in the sentence method and many more in the scenario and list methods, which made the comparison of methods challenging. To address these issues, we modified the experimental design as follows: we created a single task for each instantiated frame, and asked that 10 unique judges perform each of these tasks. Furthermore, we divided the workload into 6 batches: each batch was posted in the morning on a weekday and had a different ordering of the methods. Each batch corresponded to one of the 6 possible orderings of methods. The analysis presented in the following section is based on this latter experimental design.

5. ANALYSIS

We begin by reporting crowdsourcing process statistics. In Subsection 5.2, we discuss whether the language collected from the crowd matches the given frame semantics. Then, in Subsection 5.3 we investigate whether and to what extent these methods elicit the natural structure of language.

5.1. Crowdsourcing process statistics

9360 tasks were completed by 68 judges: 53 on the sentence method, 51 on the scenario method, and 51 on the list method. As typical for crowdsourcing tasks, the amount of work performed by different judges was unequal. For instance, the top 20% most active judges performed 55% of the tasks in the sentence method, 58% in the tasks in the scenario method, and 55% in the list method. Overall, the distribution of the number of tasks performed by the judges was similar across the three methods.

Each of the six batches posted were completed in about five hours. The average duration per task was 26.9 seconds for the sentence method, 28.5 seconds for scenario method and 25.7 seconds for the list method. The duration for the list method is statistically significantly different than that of sentence and scenario, with $p < 10^{-4}$ in a Mann-Whitney test. Across all methods, the average duration per task increases from 19.5 seconds for frames with 0 slots, to 21.1 seconds for one slot, 26.9 seconds for two slots, and 32.6 seconds for three slots. These differences are statistically significant with $p < 10^{-4}$ in a Mann-Whitney test.

5.2. Semantic error rate

We now examine the *semantic correctness* of the collected sentences. We used a semi-automatic labeling process to assess the semantics for the collected data and compared them to the given semantics.

We first performed spell checking by using Microsoft Word. Next, we manually inspected the resulting vocabulary to identify additional errors that were not found in the first pass. Overall about 8% of the collected utterances contained at least one spelling error. Finally, a text normalization step was performed, *e.g*. lowercasing, eliminating punctuation, converting numbers, time to a common format, etc.

We used an automated process to construct semantic labels for the normalized response utterances: for each utterance, we assumed that the frame was correct and we scanned for the slot values presented in the task. If all values were found, the result utterance was labeled as *correct* (94.5% of utterances), otherwise it was labeled as *error* (5.5% of utterances). We note that this labeling process is imperfect. For instance, it may mistakenly label some collected utterances as *error* due to the use of synonyms for slot values. To more accurately estimate the semantic error rate, we manually inspected all utterances that were labeled as *error*, as well as a random sample of equal size (5.5% of total) of the utterances labeled as *correct*.

This analysis revealed that 154 of the 513 utterances which had been automatically labeled *error* were in fact semantically correct; the labeling process failed on these utterances due to use of synonyms for slot values (*e.g*. expensive → high-end) and remaining spelling and normalization issues. The remaining 359 utterances (3.8% of the total) contained errors. 243 of these (2.6% of total) can clearly be assigned to worker mistakes, such as the use of incorrect slot values (149), missing or added slots (77), or garbage utterances (18). An analysis of these worker errors across methods shows that 107, 77, and 59 of these errors occurred in the sentence, scenario, and list methods respectively. 116 other errors (1.2% of total) could be traced to ambiguities introduced by the experimental design: workers doing separate tasks were not aware of the set of possible values in the ontology for the Price slot, and sometimes replaced values like "*very expensive"* with "*expensive"* (considered distinct values in our ontology).

The analysis of the 516 randomly sampled utterances that were tagged *correct* by the semantic labeler reveal that only a very small proportion (2.3%) contained semantic errors. We estimate therefore the semantic error rate in the entire corpus is 6%.

5.3. Slot order analysis

Next, we investigate the *structure* of the language collected from the crowd, focusing our attention on the order of slots in the collected utterances. We explore whether the methods elicit language that follows a natural distribution over possible slot orderings. Consider for instance the FindRestaurant(Location, Price, Cuisine) semantic frame. In our experiment, we created templates for each of the 6 possible slot orderings over these three slots (numbered 1 through 6, shown in Figure 2.D) and presented an equal number of templates based on each of these orderings to the crowd. If the elicitation method created a strong bias and the workers followed the same slot order as presented in the template, we would expect the distribution over the slot orders in the collected language to be uniform. However, in natural language, the distribution over the order of slots is non-uniform, as certain orderings are more likely than others.

Figure 2.D (top row) shows the distribution over slot orderings in the language collected for the FindRestaurant frame. The figure shows that the collected language strongly converges on a preferred ordering: FindRestaurant(Price, Cuisine, Location). The bottom row in Figure 2.D decomposes this distribution further by conditioning on the *slot ordering of the template*. As these figures show, regardless of the slot ordering in the template, the crowd is converging to the single preferred order. This is not surprising: *Find me an expensive Italian restaurant in Bellevue* is a more likely and natural request than *Find me a restaurant in Bellevue that is expensive and serves Italian food*.

Figure 2 also shows these distributions for the five other frames that take three slots in our ontology. We notice that for three of the frames, *i.e*. FindJob, FindMovie and Find-Restaurant, there is high convergence to preferred slot orderings. For the other three frames, ScheduleMeeting, ReserveRoom and ReserveRestaurant, while there is still convergence (the resulting distributions exhibit similarities for different source orderings), there is no single preferred ordering. It is important to note that this result is in line with the linguistic structure of the frames. For the Find frames the slot values operate as adjectives, and the result is consistent with the existence of a canonical order of adjectives in English. For the other three frames, the slot values often appear in the sentence as post-modifying phrases (such as prepositional phrases), and the order of attachment is not constrained. We observed similar results for frames that contain two slots: strong convergence to preferred orderings for frames like FindRestaurant(Price, Cuisine), FindMovie(Natio-nality, Gendre), FindJob(Location, Type), but no strong convergence for ReserveRestaurant(Hour, Name), Reserve-Room(Day, Duration), etc. These results suggest that the proposed methods do not significantly bias the crowd in terms of slot ordering. Where a natural ordering exists, it is being captured by the crowd responses.

While the analysis above aggregates the data across all elicitation methods, we also investigated whether any of the three methods is more or less sensitive than the others to the

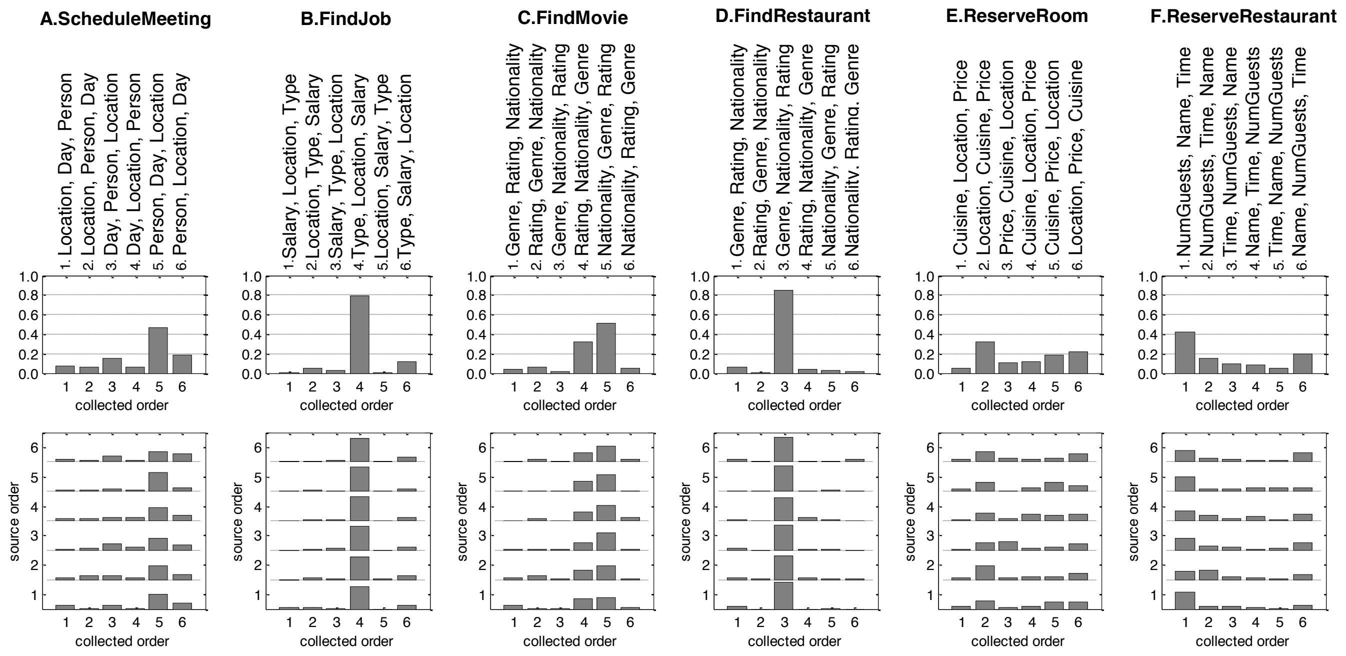

Figure 2. Slot ordering analysis. **Top row:** Each plot shows the probability of different slot orderings in the collected data for a given frame. **Bottom row:** Plots show the probability of different slots orderings in the collected data for a given frame, conditioned on the slot order in the seed template.

ordering of slots in the prompt. To quantify how sensitive the collected language is to the prompt ordering, we computed an *ordering sensitivity score* for a frame (in a given method) as follows: for each pair of templates with different slot orderings we computed the distance between the resulting slot ordering distributions in the corresponding collected data using Hellinger distance. The ordering sensitivity score is the average of these distances across all pairs. We compared the ordering sensitivity scores across methods in a paired (by frame) sign test. For the 19 instantiated frames with two slots, a statistically significant difference was detected between the list and scenario methods, with the scenario method being less sensitive ($p<0.01$). For the six instantiated frames with three slots, no significant differences were detected. Further experiments are needed to better understand the sensitivity of the methods and how it is affected by the number of slots.

6. DISCUSSION

The analysis in Section 5 suggests that the crowd can perform the task efficiently and with high accuracy, and that the methods are capable of eliciting some of the natural patterns in language.

An important question with text-based elicitation is whether the lexical content of the prompts presented to the crowd significantly influences the language collected. In the absence of a language gold standard, it is difficult to compare the biases various elicitation methods might create. In an effort to understand this phenomenon, we investigate the sensitivity of the elicited language to the prompt. For each pair of prompts for a given frame, we calculate the distance between the prompts (prompt-distance) and the distance between the corresponding elicited language (language-distance). We then explore the correlation between prompt distance and language distance for different elicitation methods as a measure of lexical sensitivity. Intuitively, if a method has low sensitivity to the prompts, the language collected would be similar regardless of how different the prompts are (the prompts follow the same semantics). The Spearman rank correlation coefficient was lowest for the list method $\rho=0.08$, followed by the scenario method $\rho=0.23$, followed by the sentence method $\rho=0.43$. These results seem to indicate that the list method is the least sensitive. We believe that more study is needed to confirm results about the relative lexical sensitivity of the methods. One issue is that the prompts for the list and scenario methods are not independently generated (we generate a prompt for each slot and compose the list and scenario), whereas individual prompts are generated for each slot order in the sentence method. Also, this analysis aggregates across frames with different number of slots, and hence does not consider the possible effects of number of slots on lexical sensitivity.

Other approaches for structured language elicitation can also be envisioned. One example is priming with images. While images would eliminate biases due to lexical variation, an image-based approach could pose design and authoring challenges, and ambiguities might arise with attempts to convey semantic content with imagery.

To date, crowdsourcing methods have focused largely on building consensus and accuracy (*e.g.* transcription). In language elicitation, the objective function is more complex; beyond accuracy (in this case semantic accuracy), we seek to elicit the natural distribution of language usage across a population of users. As such, task design and crowdsourcing controls on the worker population are important. We learned an important lesson in our first experiment: allowing the same worker to address multiple instances of the same task can lead to a lack of diversity. In the second experiment, using crowdsourcing with more controls enabled us to engage a wider population. In principle, engaging a wide population should lead to the construction of a corpus that better captures the natural distribution of language patterns, than when the corpus is authored by a single person, whether that is a crowd worker or a system designer. We seek in future study, a deeper understanding of the tradeoffs between providing enough tasks to attract workers and maintaining engagement, balancing workloads, and developing and refining these controls in a crowdsourcing platform.

Another lesson is the challenge of performing method comparisons based on data collected via crowdsourcing. As important variables (*e.g.* maximum number of tasks per worker, at which time of the day different tasks will be performed, etc.) cannot be controlled, it is generally challenging to ensure a balanced design for controlled experiments. We believe that repeated experiments and, ultimately, end-to-end evaluations are required to draw robust conclusions. Future work is needed to investigate the performance of deployed spoken dialog systems with language corpora elicited with different methods.

The methods we have discussed focus on eliciting the corresponding language (lexical form) for a given semantic form. Successful implementations of such methods may enable a number of natural language processing tasks. For instance, for data-driven natural language generation, the methods are sufficient to collect the required data, as the set of semantic forms of interest are known to the system developer. For other tasks, such as developing a spoken dialog system, the distribution of instantiated semantic forms, which is required as an input for our methods, is an important aspect of the corpus that must be collected. This distribution over semantic forms may still be authored, or may be collected by transcribing and annotating interactions of a dialog system with real users.

7. CONCLUSION

We presented a case study of structured natural language elicitation via crowdsourcing. Specifically, we investigated three text-based elicitation approaches for collecting language corresponding to a given semantic form. We studied the feasibility and accuracy of these approaches, and analyzed and discussed some of the biases these methods might introduce in the elicited language. The experiments and analysis we have conducted indicate that the proposed methods can be used to efficiently elicit natural language that matches given semantic forms. At the same time, the study has brought to the fore several challenges and opportunities for using crowdsourcing methods to acquire natural language data. Progress in these areas can lead to faster development cycles, less overhead, and increased performance. By providing on-demand low-cost access to human intelligence, crowdsourcing approaches can enable natural language systems that learn continuously and improve over their lifetimes.

8. ACKNOWLEDGMENTS

We thank Ashish Kapoor, Kristina Toutanova, and Chris Brockett for useful suggestions and advice in the development of this work. This research was performed during an internship by William Wang at Microsoft Research.

9. REFERENCES

[1] Marge, M., S. Banerjee and A. I. Rudnicky, “Using the Amazon Mechanical Turk for Transcription of Spoken Language”, In *Proc. of ICASSP*, 2010.

[2] Yang, Z., B. Li, Y. Zhu, I. King, G. Levow, and H.M. Meng, Collection of user judgments on spoken dialog system with crowdsourcing, In *Proc. of SLT*, 2010.

[3] Lane, I., M. Eck, K. Rottmann and A. Waibel, Tools for Collecting Speech Corpora via Mechanical-Turk., In *Proc. Of Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010.

[4] Alonso, O., D. E. Rose, and B. Stewart. Crowdsourcing for relevance evaluation, In *Proc. of SIGIR Forum 42*, 2008.

[5] Zaidan, O., C. Callison-Burch, Crowdsourcing Translation: Professional Quality from Non-Professionals, In *Proc. of ACL*, 2011.

[6] Burrows, S., M. Potthast, and B. Stein. Paraphrase Acquisition via Crowdsourcing and Machine Learning. In *ACM TIST* (to appear), 2012.

[7] Dolan, W. B., C. Brockett. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proc. of The Third International Workshop on Paraphrasing*, 2005.

[8] Ward, W., and B. Pellom, The CU Communicator System, In *Proc. of IEEE ASRU*, 1999.

[9] Gandhe, S., D. DeVault, A. Roque, B. Martinovski, R. Artstein, A. Leuski, J. Gerten, and D. Traum, From Domain Specification to Virtual Humans: An integrated approach to authoring tactical questioning characters, In *Proc. of Interspeech*, 2008.

[10] Aleven, V., Sewall, J., McLaren, B. M., & Koedinger, K. R. Rapid authoring of intelligent tutors for real world and experimental use. In *Proc. of ICALT*, 2006.

[11] Allen, J. F., B. W. Miller, E. K. Ringger, and T. Sikorski. 1996. A robust system for natural spoken dialog, In *Proc. of the ACL*. 1996.

[12] Gorin, A. L., G. Riccardi, J. H. Wright, How may I help you? In *Speech Communication*, 1997.

[13] Lathrop, B. et al. A Wizard of Oz framework for collecting spoken human-computer dialogs: An experiment procedure for the design and testing of natural language in-vehicle technology systems, In *Proc. ITS*, 2004.