SPOKEN DIALOGUE GRAMMAR INDUCTION FROM CROWDSOURCED DATA

Elisavet Palogiannidi[1], *Ioannis Klasinas*[1], *Alexandros Potamianos*[2], *Elias Iosif*[1]

[1]School of ECE, Technical University of Crete, Chania 73100, Greece
[2]School of ECE, National Technical University of Athens, Zografou 15780, Greece
epalogiannidi@gmail.com iklasinas@isc.tuc.gr apotam@gmail.com iosife@telecom.tuc.gr

ABSTRACT

We design and evaluate various crowdsourcing tasks for eliciting spoken dialogue data. Task design is based on an array of parameters that quantify the basic characteristics of the elicitation questions, e.g., how open-ended is a question. The crowdsourced data are used for and evaluated on the unsupervised induction of semantic classes for speech understanding grammars. We show that grammar induction performance is significantly affected by the crowdsourcing task parameters, e.g., paraphrasing tasks prime high lexical entrainment and result in poor corpus/grammar quality. The task parameters along with perplexity filters are used for corpus selection achieving grammar induction performance that is comparable to that of using in-domain spoken dialogue data.

Index Terms— Crowdsourcing, Spoken Dialogue Systems, Grammar Induction

1. INTRODUCTION

A significant obstacle for the further adoption of speech interfaces is the significant development time needed to port an existing system into a new domain or language. The portability of a Spoken Dialog System (SDS) is heavily depended on the availability of grammars for the domains/languages of interest. Tool-assisted grammar development can alleviate the need for manual labor enabling the rapid porting of SDS. Such tools span from grammar editing environments [1] to algorithms of grammar induction [2, 3]. Induction algorithms can be broadly divided into two main categories, namely, top-down (resource-based) and bottom-up (data-driven). The Grammatical Framework Resource Grammar Library (GFRGL) [4] is a resource mainly following the top-down approach, which allows for the creation of multilingual grammars according to a high-level formalism. The basic idea is to hide the linguistic details (e.g., morphology) from the grammar developer. An example of an end-to-end SDS based on GFRGL is presented in [5]. The main drawback of the top-down paradigm is the requirement of pre-existing knowledge, which might not be available for under-resourced languages. This is tackled by the bottom-up paradigm that relies (mostly) on textual data for the induction of grammars. The key idea is the automatic induction of domain concepts (also referred to as semantic classes) that consist of semantically similar words/phrases. Semantic similarity is estimated according to the distributional hypothesis of meaning: "similarity of context implies similarity of meaning" [6]. Several similarity metrics have been used for this task [3], while different clustering algorithms have been employed for the creation of semantic classes, e.g., hard [2] and soft [7] agglomerative clustering. In recent work [8] the performance of the bottom-up approach was investigated using various schemes for harvesting and filtering web data.

Crowdsourcing is a very popular method for various natural language and speech processing tasks [9, 10, 11]. Examples include sentence translation from one language to another or gathering annotations on bilingual lexical entries [12, 13], as well as paraphrasing applications [14, 15]. Reports of utilizing crowdsouring for evaluating an SDS can be found in [16, 17, 18, 19]. Crowdsourcing methods are especially relevant for resource-poor languages [13, 14].

In this paper, we investigate various methods (tasks) to elicit spoken dialogue text data via crowdsourcing for grammar induction. Each task is characterized by a list of factors, e.g., how open ended a prompt is, degree of politeness, prompt length. The quality of crowdsourced data is analyzed in terms of richness, in-domainess, degree of entrainment, as well as in terms of the performance of the grammar induction algorithm. Various data selection algorithms are investigated using language perplexity and task factors to select the most informative parts of the gathered corpus and boost grammar induction performance. The main difference with traditional crowdsourcing experiments, e.g., [12], is the different elicitation methods investigated here. Also, in contrast to [16, 17, 18, 19], the focus is not on evaluating SDS, but on creating a corpus useful for the development of a SDS. In addition, we propose to parametrize the crowsourcing task making it amenable to machine learning algorithms for corpus selection.

2. CROWDSOURCING TASKS AND PARAMETERS

In order to elicit realistic SDS data we designed four crowdsourcing tasks that simulate SDS interaction. Hence, the majority of the tasks follows a *question and answer* structure. Specifically the following tasks were created: 1) *Answers* is a task for collecting answers from questions (SDS prompts), 2) *Paraphrasing* is used for collecting paraphrases of an (underlined) portion of a sentence (corresponding to a prompt or user input), 3) *Complete the dialogues* (Compl. Dlg.) where task contributors must insert suitable answers and questions to incomplete dialogues and 4) *Fill in* where task contributors must fill in the missing part of a sentence, i.e., complete a sentence. Illustrative examples of the four elicitation methods are shown in Fig. 1 for a travel domain. Empty fields must be filled in by the contributor.

2.1. Task Parameters

It is reasonable to assume that the way the questions are structured in each task determine the quality of the acquired data. In order to validate this assumption a number of task characteristics were defined and their impact on the gathered corpus was investigated. In Table 1, we list the parameters we used along with associated values and in which task each one is applicable. Specifically:

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

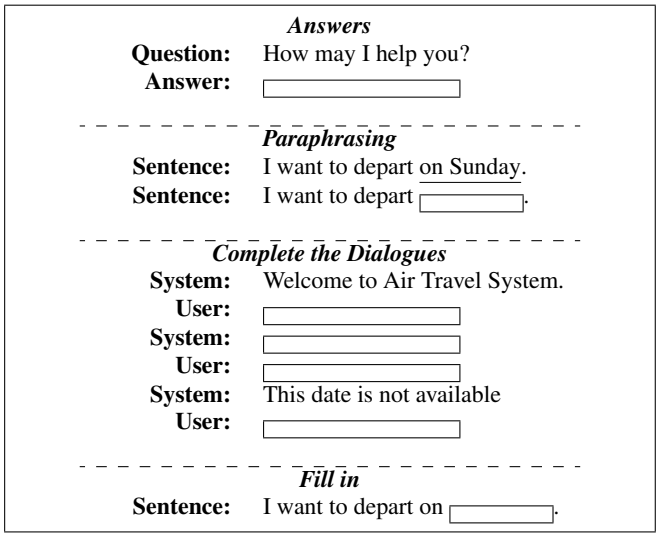

Fig. 1. Examples of the four crowdsourcing tasks.

Op-en parameter expresses how specific is the expected reply to a question. Questions with high open-endedness are commonly used in the beginning of a dialogue and have more abstract meaning (e.g, "*How may I help you?*", Op-en = 4) in contrast to questions with low open-endedness (e.g, "*Please provide the day of arrival?*, Op-en = 0).

Pol. takes higher values for questions that use politeness markers, such as "please".

#Wor. is the length of the question in words.

#Con-req. quantifies the concepts that are requested in each question, e.g, "*When do you want to depart and which airport do you prefer?*", requests departure date and departure airport thus #Con-req. = 2. For open-ended questions we assume that the number concepts requested is fixed large number (#Con-req. = 10).

#Wor-par. is the length of the portion of the sentence that is being paraphrased (in words).

Position is the position in the sentence (left, right or center) of the fragment that is being paraphrased or filled-in, e.g., for the example in Fig. 1 the Position is "right".

Freedom is the ratio of the empty fields over the total fields in a dialogue. For example in Fig. 1 there are four empty fields and Freedom is $4/6$ or 67%.

During the design phase, we have created questions that adequately sample the task parameter space for each task. First, we created a number of "skeleton" questions per task and we assigned them values for each parameter. Next, we created four new questions with close meaning to the initial, by changing one of the parameter values and keeping the rest constant. Repeating this process for all the skeleton questions, we ended up with 300 questions and 40 incomplete dialogues that were used for the four crowdsourcing tasks.

The majority of these parameters try to gauge how much freedom is given to the workers. Note that each task by design allows for higher or lower freedom. Specifically, *Paraphrasing* allows the lowest freedom, providing the most piece of information, while *Complete the dialogues* allows the highest freedom. Typically, higher freedom results in a richer corpus, while lower freedom results in a more in domain corpus. Examining the reactions (answers) of the contributors, we can extract conclusions about the optimal values for each parameter and task. These conclusions can be used for future task design, as well as for filtering the crowd sourced corpus as detailed in Section 5.

Parameter	Values	Tasks
Open-ended (Op-en)	0-4	All
Politeness (Pol.)	0-5	All
#Words (#Wor.)	≥ 1	All
#Concepts-requested (#Con-req.)	≥ 1	All
#Words-paraphrase (#Wor-par.)	≥ 2	Paraphrasing
Position in sentence (Position)	left,right, center	Paraphrasing, Fill in
Dialogue Freedom (Freedom)	0-100 %	Compl.Dlg

Table 1. Parameters, range of values, and associated tasks.

2.2. Experimental Procedure

We have focused on designing rules for a finite-state SDS grammar in the travel domain for English. Only a subset of the grammar rules were targeted, namely eliciting data for 1) Date and 2) Departure-city concepts. The Crowdflower platform [20] was used to gather the data. The major problem during this process was the assurance of quality control, because, the nature of the tasks we designed didn't allow for the use of Gold Standard Data[1]. First, we used the flagging mechanism in order to exclude contributors providing irrelevant data from participating again to any crowdsourcing tasks we upload. In addition we experimented with varying the payments, starting from 2 cents and converging to 0.6 cents per unit (Human Intelligent task), as well as restricting the maximum number of units that a contributor could submit.

3. CORPUS FILTERS

The collected corpus has been filtered using both perplexity and task parameter constraints. The end-goal here is to select a corpus that performs best for grammar induction as detailed in Section 5.

3.1. Perplexity

Perplexity has been used as criterion for corpus selection for both web-harvested and crowdsourced corpora, e.g., see [8]. Given a sentence W_1^I of length I and a probability model P, the perplexity is computed from the formula $ppl = 10^{-logP(W_1^I)/I}$. Sentences with high probability have low perplexity and come from a distribution that is similar to that of the model. The probability model in this work has been built from the crowdsourced corpus proper. The goal here is to exclude out-of-domain sentences that have especially high perplexity.

3.2. Perplexity and task parameters combination

In order to investigate if we can achieve better filtering results combining information from the task design (task parameters) and the resulting answer (perplexity) we used a linear combination scheme, computing for each sentence s a score $LC(s)$:

$$LC(s) = \sum_{i=1}^{n} \lambda_i x_i' : \sum_i \lambda_i = 1 \quad (1)$$

[1]Crowdflower's mechanism for automatic quality control

where x_i are the features used and λ_i the coefficients. For perplexity it is reasonable to assume that the best sentences are the ones with the smaller perplexity. For the task features this is not necessarily true and they were first transformed using the formula $x'_i = |x_i - x_{i0}|$ where x_{i0} is the value of x_i for which the best results are achieved[2]. In addition, since the dynamic range of each feature is different, they were normalized to a 0 to 1 scale. The sentences were then ranked in ascending order with respect to the computed scores and the top 20% were extracted and used to run the grammar induction module. To optimize the λ_i values, the Simplex algorithm was used [21].

4. LEXICAL ENTRAINMENT ANALYSIS

Entrainment, a common phenomenon in SDS, expresses the fact that a person adapts his behavior to his conversational partner. Lexical entrainment has been examined in [22, 23, 24]. Our goal here is to analyze for which tasks lexical entrainment is more prominent, i.e., find under which conditions the contributor has a tendency to copy chunks of the input. Intuitively, high lexical entrainment is a problem as it results in a corpus that provides little new information. For the lexical entrainment experiments we worked on two corpora, one for the sentences contributors provided (user's side) and one for the corresponding questions we designed (system's side)[3].

We estimated lexical entrainment implementing one of the metrics presented on [22]:

$$Entr1(c) = \sum_{w \in C} entr(w) \tag{2}$$

$$entr(w) = -\mid \frac{count_{s1}(w)}{ALL_{s1}} - \frac{count_{s2}(w)}{ALL_{s2}} \mid \tag{3}$$

where, w is the word, c is the class of words, $s1$ is the system side, $s2$ is the user side, and ALL is the total number of words. Note that Entr1 takes values in $(-\infty$,0] with 0 denoting the perfect match on lexical items and $-\infty$ the perfect mismatch.

5. RESULTS

431 unique contributors participated in the crowdsourcing experiment (69 of them were flagged). A total of 250 crowdsourcing jobs were run over a period of 14 days. The resulting corpus consists of 21,760 sentences[4] (167,558 words) and its total cost was $450.

The quality of the corpus and corpus filtering algorithms is evaluated on the performance of the grammar induction algorithm in [8], using a handcrafted golden grammar from the ATIS corpus. The overall induction algorithm consists of three main modules: 1) identification of multi-word terms, e.g., "New York", 2) induction of terminal rules, e.g., <City> = ("New York", "Boston", ...), and 3) induction of non-terminal rules, e.g., <DepartureCity> = ("depart from <City>", "leave <City>", ...). In this work, we assume that the multi-word terms are known and focus on the induction of terminal rules. Following the distributional hypothesis of meaning, the semantic distance between two words (or terms) was estimated as the *Manhattan-norm* (M) of their respective bigram probability distributions of left and right contexts [3]. The pairwise word similarities were used for building a similarity matrix that was given as input to an agglomerative clustering algorithm. In particular, we used the CLUTO toolkit [25], while we experimented with various number of clusters. Each resulting cluster was assumed to correspond to an induced terminal rule. For evaluation purposes, each rule was mapped to the corresponding (best match) ground-truth rule. Class-weighted precision, recall, and F-measure[5] were used as evaluation metrics in relation to a ground-truth grammar for the travel domain. In addition, the number of terminal concepts (e.g. <City>) and terminal instances (e.g. <City> = "New York") are reported.

5.1. Corpus Analysis: Task and Parameters

For each crowdsourcing task a corpus was created using the respective data, while the corresponding individual corpora were merged into a single corpus (All).

The corpora were filtered according to the parameter values of the corresponding crowdsourcing tasks. The grammar induction algorithm was applied over the resulting corpora, while the F-measure was used for evaluating the induced rules (i.e., terminal concepts). The F-measure for the grammar induction is presented in Table 2 along with the numbers of the respective terminal concepts and their instances. These are shown for various corpora created via different crowdsourcing tasks, as well as for different methods for corpus filtering and parameters[6] relevant to the aforementioned crowdsourcing tasks.

Regarding **Op-en** and **#Con-req** the highest F-measure is obtained when moderate values are used. This observation suggests that the utilization of open/specific prompts and the request of limited/detailed information (counted as the number of concepts) in prompts do not necessarily help the workers provide more rich responses. The request for a single concept only seems to improve the variability of answers as indicated by the high number of concepts (11) and the respective instances (145) that are included in the induced grammar. Regarding the politeness factor the highest F-measure is interestingly achieved by the less polite prompts. In addition, relatively short prompts (i.e., #Words<=7) appear to obtain higher performance compared to lengthier prompts. Regarding the **Position** parameter better performance is observed when both right and left context were provided indicating the need for surrounding pragmatics. The highest F-measure for the *Complete the Dialogues* task is obtained for **Freedom** $> 66\%$. This mid-level degree of freedom was observed to enable the workers to provide diverse responses, while preserving the overall thematic coherence of answers.

5.2. Lexical Entrainment

As in [22] we have computed entrainment for the class with the 25 most frequent words in the created corpus. The results are presented in Table 3. We observe that the highest entrainment is observed for the *Paraphrasing* task. High entrainment consistently results in low F-measure performance for grammar induction, especially for*Paraphrasing*. This backs the idea mentioned in Section 4, that high entrainment hurts performance. High entrainment values imply that the answers are following a lexical usage pattern similar to that of the prompt. As a result there is little variability in the crowdsourced answers and consequently little new information to be exploited by the grammar induction algorithm. For the rest of the tasks there is no clear conclusion. While the entrainment value ranges from -0.39 to -0.23, the corresponding F-measure shows

[2]As discussed in the next section, sentences collected using mid-range values of task parameters typically produced better grammar induction performance.

[3]User's side corpus contained almost the half words of system's side

[4]Available at `http://users.isc.tuc.gr/~epalogiannidi/icassp_2014.html`

[5]Precision and recall are omitted due to lack of space.

[6]Only the most conclusive parameter values are reported due to space limitations.

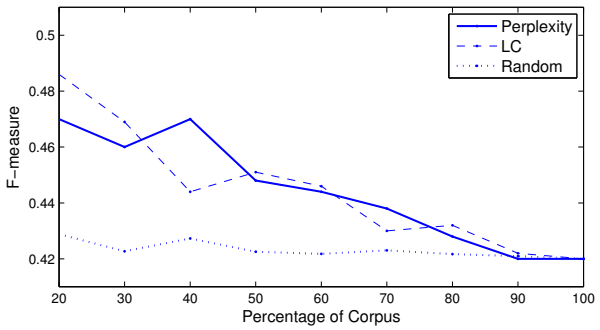

Fig. 2. F-measure for increasing size corpora, using Perplexity, LC-20% and Random filter

very little variation, from 0.46 to 0.48. So, although entrainment is a good indicator for grammar induction performance, the elicitation method, and degree of open-endedness also play a role, i.e, low entrainment can indicate both rich input, as well as out-of domain input.

5.3. Corpus Selection

Given the composite corpus (i.e., merged corpora that were created by all crowdsourcing tasks) two naive filtering steps were applied: (i) exclusion of corpus created via the paraphrasing task due to its limited richness, and (ii) exclusion of data provided by the flagged contributors. The performance of the grammar induction algorithm using the resulting corpora is presented in Table 2. The initial (i.e., non-filtered) corpus is denoted as "All", while "NP" and "'FF" stand for the corpora after applying (i) and (ii), respectively. The performance yielded by "NP" and "'FF" is shown to be at least as good as the performance obtained when using the initial corpus.

The results for perplexity based filtering as well as the its linear combination with the task features are presented in Fig. 2. The *Random* is the average of 20 runs selecting randomly incremental subsets of the corpus. In *Perplexity* the sentences are ranked in ascending order according to perplexity and the top 20%, 30% etc. were picked and the resulting F-measure plotted. Finally, for the *LC* (Linear Combination) described in Subsection 3.2 the features used are perplexity and **Op-en**, **Pol** and **#Wor**, the latter ones chosen among all task features because they are available for all tasks. As described in Subsection 3.2 the values used for the transformation of the task features are $|\textbf{Op-en} - 2|$, $|\textbf{Pol} - 2|$ and $|\textbf{\#Wor} - 7|$.

Comparing perplexity to random selection we see a large increase in F-measure for small subsets of the corpus. This is an important finding, because it means it is possible to perform filtering without relying on external tools, just by utilizing the gathered corpus. Unfortunately the linear combination does not improve results on using perplexity alone. It outperforms perplexity for the 20% and 30% subsets of data but is much worse for 40%. We experimented on optimizing on the combined F-measure of the 20% and 40% corpora, however the results did not improve. This might be due to the fact that the gathered corpus is small and does not offer sufficient data to utilize the task performance features.

6. CONCLUSIONS

In this paper we designed crowdsourcing tasks for collecting text data relevant to SDS for an array of parameters. We also investigated the performance of various filtered versions of the crowdsourced corpora for a grammar induction task. The quality of corpora was also analyzed using entrainment. The main result is that task design greatly influences the gathered corpus and grammar induction performance. Specifically, we observe that the *Paraphrasing* task gave significantly worse performance comparing to the rest elicitation tasks. *Complete the dialogues* gives best performance similar to that of the *Answers* and *Fill in* tasks. We showed that elicitation via the *Paraphrasing* task primes the contributor to lexical entrainment and as a result the responses are less varied. Another important finding is that the task parameters are also good indicators of corpus quality and performance. Higher degree of freedom leads to more varied responses and better performance, however, too much freedom also results into out-of-domain input (and eventually hurts performance). In addition, we have shown that performance can be improved using simple perplexity based filtering or task parameter-based filtering. In summary, we showed that crowdsourcing can be used to gather data for grammar induction, achieving comparable performance to web-harvested [8] or manually created in-domain corpora. In future crowdsourcing experiments, we will expand on the use of machine learning to design elicitation methods and select corpora for grammar induction.

Corpus	Words	F-meas.	Terminal Concepts	Terminal Instances
All	**167558**	0.42	8	173
Flag Filter(FF)	139828	0.43	9	**175**
No *Paraphrasing*(NP)	126058	**0.44**	**11**	167
Answers	16175	0.46	7	**123**
Paraphrasing	41500	0.35	8	46
Complete the Dialogues	47108	**0.48**	**10**	122
Fill in	**62775**	0.47	**10**	98
Open-ended=0	18735	0.42	8	66
Open-ended=2	**48141**	**0.45**	**11**	119
Open-ended=4	46137	**0.45**	6	**120**
Politeness=0,1	29740	**0.47**	9	115
Politeness=2,3	**116899**	0.42	**10**	**159**
Politeness=4,5	20919	0.42	7	79
#Concepts-requested=1	**66848**	0.42	**11**	**145**
#Concepts-requested=2	44116	**0.48**	9	100
#Concepts-requested>2	33365	0.41	9	65
#Words<=7	76845	**0.45**	10	**153**
#Words>7	**90713**	0.42	**11**	138

Table 2. Grammar induction performance for various tasks and filtering methods.

Task	Entr1	F-measure
Answers	-0.39	0.46
Paraphrasing	-0.14	0.35
Complete the dialogues	-0.23	0.48
Fill in	-0.35	0.47

Table 3. Entrainment metric per task, and corresponding F-measure for grammar induction.

Acknowledgements This work has been been partially funded by the PortDial project ("Language Resources for Portable Multilingual Spoken Dialog Systems") supported by the EU Seventh Framework Programme (FP7), grant number 296170.

7. REFERENCES

[1] Y.-Y. Wang and A. Acero, "Rapid development of spoken language understanding grammars," *Speech Communication*, vol. 48, no. 3-4, pp. 390–416, 2006.

[2] H. Meng and K.-C. Siu, "Semi-automatic acquisition of semantic structures for understanding domain-specific natural language queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 1, pp. 172–181, 2002.

[3] A. Pargellis, E. Fosler-Lussier, C. H. Lee, A. Potamianos, and A. Tsai, "Auto-induced semantic classes," *Speech Communication*, vol. 43, no. 3, pp. 183–203, 2004.

[4] A. Ranta, "Grammatical Framework: A Type-Theoretical Grammar Formalism," *Journal of Functional Programming*, vol. 14, no. 02, pp. 145–189, Mar. 2004.

[5] "Talk project, deliverable 1.6," `http://talk-project.eurice.eu/`.

[6] Z. Harris, "Distributional structure," *Word*, vol. 10, no. 23, pp. 146–162, 1954.

[7] E. Iosif and A. Potamianos, "A soft-clustering algorithm for automatic induction of semantic classes," in *Proc. of Interspeech*, 2007, pp. 1609–1612.

[8] I. Klasinas, A. Potamianos, E. Iosif, S. Georgiladakis, and G. Mameli, "Web data harvesting for speech understanding grammar induction.," in *Proc. of Interspeech*, 2013, pp. 2733–2737.

[9] I. McGraw, J. Glass, and S. Seneff, "Growing a spoken language interface on amazon mechanical turk.," in *Proc. of Interspeech, Florence*, 2011, pp. 3057–3060.

[10] C. Callison-Burch and M. Dredze, "Creating speech and language data with amazon's mechanical turk," in *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 1–12.

[11] W. Wang, D. Bohus, E. Kamar, and E. Horvitz, "Crowdsourcing the acquisition of natural language corpora: Methods and observations," in *"Proc. of IEEE Spoken Language Technology Workshop (SLT)"*, 2012, pp. 73–78.

[12] V. Ambati and S. Vogel, "Can crowds build parallel corpora for machine translation systems?," in *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 62–65.

[13] A. Irvine and A. Klementiev, "Using mechanical turk to annotate lexicons for less commonly used languages," in *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 108–113.

[14] M. Denkowski, H. Al-Haj, and A. Lavie, "Turker-assisted paraphrasing for english-arabic machine translation," in *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 66–70.

[15] O. Buzek, P. Resnik, and B. Bederson, "Error driven paraphrase annotation using mechanical turk," in *Proc. of NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 217–221.

[16] A. Raux, B. Langner, D. Bohus, A. Black, and M. Eskenazi, "Lets go public! Taking a spoken dialog system to the real world," in *Proc. of Interspeech*, 2005.

[17] Z. Yang, Y. Li, B.and Zhu, I. King, G. Levow, and H. Meng, "Collection of user judgments on spoken dialog system with crowdsourcing," in *Proc. of IEEE Spoken Language Technology Workshop (SLT)*, 2010, pp. 277–282.

[18] F. Jurcıcek, S. Keizer, M. Gašic, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Real user evaluation of spoken dialogue systems using amazon mechanical turk," in *Proc. of Interspeech*, 2011, pp. 3061–3064.

[19] Y. Zhu, Z. Yang, H. Meng, B. Li, G. Levow, and I. King, "Using finite state machines for evaluating spoken dialog systems," in *Proc. of IEEE Spoken Language Technology Workshop (SLT)*, 2010, pp. 478–483.

[20] "Crowdflower crowdsourcing service," `http://crowdflower.com/`.

[21] J. A Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[22] A. Nenkova, A. Gravano, and J. Hirschberg, "High frequency word entrainment in spoken dialogue," in *Proc. of ACL Human Language Technologies: Short Papers*, 2008, pp. 169–172.

[23] G. Parent and M. Eskenazi, "Lexical entrainment of real users in the let's go spoken dialog system.," in *Proc. of Interspeech*, 2010, pp. 3018–3021.

[24] S. Stoyanchev and A. Stent, "Lexical and syntactic priming and their impact in deployed spoken dialog systems," in *Proc. of ACL Human Language Technologies: Short Papers*, 2009, pp. 189–192.

[25] "Cluto: A clustering toolkit," `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview`.