Seeing What You Said: How Wizards Use Voice Search Results

Rebecca J. Passonneau[1], Susan L. Epstein[2,3], Joshua B. Gordon[4] and Tiziana Ligorio[2]

[1]Center for Computational Learning Systems, Columbia University
[2]Department of Computer Science, Hunter College of The City University of New York
[3]Department of Computer Science, The Graduate Center of The City University of New York
[4]Department of Computer Science, Columbia University
becky@cs.columbia.edu, susan.epstein@hunter.cuny.edu, joshua@cs.columbia.edu, tligorio@gc.cuny.edu

Abstract

A Wizard-of-Oz experiment investigates how voice search could affect dialogue management strategies. The study design has two novel components. First, a single turn exchange is examined, rather than a full dialogue. Second, wizards partner with a dialogue system, so internal system features unavailable to the wizard can be used to model wizard actions. Wizards see the output of automated speech recognition (ASR) for a book title request, plus a ranked list of candidate titles from a backend query. The features that contribute most to a regression model of the wizards' actions prove to be the utterance level confidence score on the ASR, and the backend return type. People who compare ASR strings to candidate titles can select the correct one if it is there, and do so more confidently when the backend return has higher confidence.

Introduction

For at least the past decade, the quality of automated speech recognition (*ASR*) within spoken dialogue systems (*SDSs*) has been acknowledged as a limiting factor for user satisfaction, task success and other measures of performance (Litman, Walker and Kearns, 1999; Walker et al., 1997). Information-seeking and transaction-based systems (Georgila, et al. 2003, Johnston, et al. 2002, Levin, et al. 2000, Raux, et al. 2006, Zue, et al. 2000) query a backend database for information or to perform actions. The dialogue manager typically maintains system initiative, and aims for short, unambiguous user utterances through carefully designed prompts. This supports maximally accurate backend queries while minimizing clarification subdialogues. CheckItOut, a transaction-based SDS that handles telephone requests for library books, is a mixed initiative system. It accesses a library database where the mean length of the book title field is five words and the median is nineteen. Multiword book titles in the context of book request dialogue acts present an unusual challenge for SDS, particularly with mixed initiative. To address this challenge, we query the backend with ASR for book titles, rather than a semantic interpretation resulting from a natural language understanding phase. This amounts to integrating voice search into SDS.

This paper presents preliminary results of an experiment investigating how voice search could affect dialogue management strategies. The principal findings pertain to three cases of backend return. Humans who compare ASR strings to candidate book titles are justifiably confident in selecting a title when the backend return has high confidence. When the backend has only moderate confidence, our subjects select a title with justifiably less confidence. When the backend return has low confidence, subjects correctly select a title only about a third of the time, and are tentative when they do so.

Voice search has been investigated primarily to access the web via mobile devices (Franz & Milch 2002; Paek & Yu 2008). In our experiment, ASR output is used to query a database of book titles. Often only a few returned titles (*candidates*) will both be roughly the same length as the ASR string and match one or more content words (i.e., nouns, verbs, adverbs and adjectives). For example, for the title *Billy Phelan's Greatest Game,* the ASR output in our experiment was "`billies villains greatest`." A simple query method using that string returned three candidate titles:

- Billy Phelan's Greatest Game
- Baseball's Greatest Games
- More like Us: Making America Great Again

Our subjects' task is to guess which of the candidates returned by the backend query is correct, if any, and to formulate a question if they cannot select a candidate.

The experiment described here relies on the Wizard of Oz (*WOz*) paradigm. In WOz studies, a human subject interacts with a *wizard*, whom she believes to be a computer but is actually a person. Our subjects perform as wizards or as mock callers, using a graphical user interface (*GUI*) rather than a telephone. This work employs two novel adaptations of WOz. First, we examine a single turn exchange, comparable to the well-known notion of adjacency pair (Sacks, Schegloff and Jefferson, 1974), rather than a full dialogue. Second, wizards operate as partners with our dialogue system, which allows us to model their behavior with system-internal features unavailable to wizards, as well as with features representing the wizards' dialogue state.

After an overview of related work, this paper describes CheckItOut and three types of subdialogue likely to arise with voice search around book title requests. Subsequent sections describe the experimental design and results of the experiment, particularly the factors that account for wizards' decisions. The final two sections discuss implications for future work and summarize the contributions presented here.

Related Work

ASR quality, as measured by word error rate (*WER*), typically falls in the range [0.25, 0.65], depending upon such factors as vocabulary size, perplexity of the language model, and diversity of the user population by gender, age, and native language. The impact of WER on SDS performance can also vary considerably, depending on whether the system maintains initiative and on the design of system prompts. CMU's Let's Go!, which provides bus route information to the public from data provided by the Port Authority of Allegheny County, maintains system initiative. The average WER reported in (Raux et al., 2005) was 0.60, due in part to a user population that included elderly and non-native speakers, and in part to the conditions under which users access the system. Callers often called from noisy street locations, or from indoor locations with background noise, such as televisions

Approaches to error-ridden ASR either try to improve the recognizer's data or algorithms, for example through speaker adaptation (Raux, 2004), or try to compensate for transcription errors through error handling dialogue strategies (Bohus, 2004). For the directory service application in (Georgila et al., 2003), users spell the first three letters of surnames, and ASR results are expanded using frequently confused phones. (Stoyanchev and Stent, 2009) add a two-pass recognition architecture to Let's Go! to improve concept recognition in post-confirmation user utterances.

Turn segmentation and disfluencies also affect recognizer performance. A long pause, for example, is likely to be interpreted as the end of the speaker's turn, even if it occurs within the utterance of a long book title. The Let's Go! architecture now has an explicit representation of the conversational floor, the real-time events that take place when speakers seize or cede the next turn (Raux and Eskenazi, 2007). To detect utterance boundaries, an interaction manager uses information from the speech recognizer, a semantic parser, and *Helios*, an utterance-level confidence annotator.

The goal of a WOz study is to elicit behaviors likely to appear when a system replaces the wizard. Work on the impact of ASR errors in full human-wizard dialogues clearly demonstrates that wizards do not aim for full interpretation of every utterance (Rieser, Kruijff-Korbayová and Lemon 2005, Skantze 2003, Williams and Young 2004, Zollo 1999). Zollo collected seven dialogues with different human-wizard pairs whose task was to develop an evacuation plan. The overall WER was 30% and there were 227 cases of incorrect ASR. Nonetheless, wizard utterances indicated a failure to understand for only 35% of them. Instead, wizards ignored words not salient in the domain and hypothesized words based on phonetic similarity. In another study where both users and wizards were treated as subjects, and both knew there was no dialogue system, 44 direction-finding dialogues were collected involving 16 subjects (Skantze, 2003). Despite a WER of 43%, wizard operators signaled misunderstanding only 5% of the time. For the 20% of non-understandings, operators continued a route description, asked a task-related question, or requested a clarification of what had been said.

Simulated ASR controls for the degree of transcription errors, allow collection of dialogues without building or tuning a speech recognizer, and can deliberately deprive the wizard of prosody (Rieser, Kruijff-Korbayová and Lemon, 2005; Williams and Young, 2004). A typist transcribes the user utterances, and errors are introduced systematically. In (Williams and Young, 2004), 144 dialogues were collected simulating tourist requests for information, and WER was constrained to be high, medium or low. High WER decreased full understandings and increased unflagged misunderstandings (where the wizard did not show evidence of detecting the misunderstanding). Under medium WER, a task-related question in response to non-understanding or misunderstanding more often led to full understanding in the next wizard turn than a repair did. Under high WER, when wizards followed a non-understanding or misunderstanding with a task-related question instead of a repair, unflagged misunderstanding significantly increased.

The present experiment is a step towards *wizard ablation*, described in (Levin and Passonneau, 2006), in which the wizard relies on system inputs or outputs, rather than human ones. The hypothesis is that behaviors elicited from wizard/subject pairs in an ablated wizard study will be more pertinent for investigating dialogue strategies given the current state-of-the art in component technologies, such as speech recognition. Here we ablate the input channel to the wizard, so that the wizard has access only to the output of the speech recognizer, not the caller's speech.

In an offline pilot study for this experiment (Passonneau, Epstein and Gordon, 2009), three speakers each read fifty book titles to generate three sets of ASR transcription. Each set was presented to one of three wizards who were asked to find the correct title by searching a plain text file of more than 70,000 titles. WER ranged from 0.69 to 0.83, depending on the speaker. Despite this high WER, on average wizards were able to find the correct title 74% of the time.

The current experiment provides a benchmark for the performance of voice search techniques within the context of CheckItOut, and data on the types of subdialogue to expect for book requests by title. Our initial goals are to identify the contexts in which wizards perform well at selecting the correct title, and especially, to characterize the contexts where they do not, as these are the contexts likely to benefit the most from strategic dialogue management.

CheckItOut

CheckItOut handles book requests made to librarians at the Andrew Heiskell Braille and Talking Book Library. Heiskell is a branch of the New York Public Library and part of the National Library System (*NLS*). Patrons request materials by telephone and receive them by mail. Heiskell and other NLS libraries could greatly benefit from a system that automates some of the borrowing requests.

CheckItOut draws on the Olympus/Ravenclaw architecture and dialogue management framework (Bohus et al., 2007; Bohus and Rudnicky, 2003). *Olympus* is a domain-independent dialogue system architecture based upon the earlier CMU Communicator (Rudniky, 2000). *Ravenclaw* (Bohus, 2004) is a dialogue management framework that

separates the domain-dependent task structure from domain-independent error-handling and clarification strategies. Olympus/Ravenclaw has been the basis for about a dozen research dialogue systems in different domains.

CheckItOut has domain-specific code for the task structure of the dialogue. The backend accesses a sanitized version of Heiskell's database of 5028 active patrons, and its full book database with 71,166 titles and 28,031 authors. Titles and author names contribute 54,448 words to the vocabulary.

In a dialogue with CheckItOut, a caller identifies herself, requests books, and is told which are available for immediate shipment and which will go on reserve. The caller can request a book by catalogue number, by title, or by author. We recorded and transcribed 82 calls to the library. Approximately 44% of the book requests were by number, 28% by title or a combination of title and author, and the remainder represented a range of more general book requests. Because patrons receive monthly newsletters listing new titles, they request books with knowledge of the bibliographic data or catalogue numbers. As a result, most title requests from patrons are nearly exact matches to the actual title. For present purposes, we assume they request the exact title or nearly so.

We exploited the Galaxy message passing architecture of Olympus/Ravenclaw to insert a wizard server into CheckItOut. This makes it possible to pass messages from the system to a wizard GUI, or from the wizard GUI to the system. By embedding our wizard within the system, we can examine how wizard actions relate to information available to the system at runtime. Because CheckItOut relies on the same version of Olympus as Let's Go!, we can access features used by the interaction manager mentioned above. This allows us to test whether system features available during the speech recognition phase can be used to model wizards' decisions.

We used PocketSphinx 0.50 for speech recognition, and microphone bandwidth acoustic models from Let's Go!. Like the user population of Let's Go!, patrons of the Andrew Heiskell library include many elderly and non-native speakers. Our target population differs in that patrons qualify for access to Heiskell because they cannot read books in printed format. Many patrons are legally blind, or lack the motor skills to manipulate a book. In separate work, we are evaluating the recognition performance on speech from our transcribed corpus of patron-library calls to determine the utility of additional iterations of acoustic training.

To present challenging cases to our wizards we aimed for a relatively high but not intractable WER. We sought a WER similar to that managed by wizards in the offline pilot study, but with a model that covered the titles in the database. WER was computed using Levenshtein distance (Levenshtein 1996). A statistical language model assigns a probability distribution to possible word sequences. To select a language model, we first manipulated WER by constructing several bigram language models of varying sizes. We randomly selected 10,000 titles (~11K words) from the library database, and then selected from it subsets of size 7,500 (~9K words), 5,000 (~6.8K words) and 1,000 titles (~2K words). For each of the four sets of titles, we constructed a bigram language model. For each language model size, one male and one female each read a set of 50 titles used in our offline pilot. From this, we determined that a language model based on 7,500 titles would yield the desired WER.

To model real-world conditions more closely, titles with below average circulation were eliminated before we selected a set to build the language model for our experiment. We also eliminated one-word titles and those containing non-alphanumeric characters. A random sample of 7,500 was chosen from the remaining 19,708 titles to build a bigram language model. It contained 9,491 unique words. The 4,200 titles in the experimental materials were drawn from the 7,500 titles used in constructing the language model. Average WER for the book title requests in our experiment was 0.69.

Experimental Design

For the current study, we implemented a backend query that returns a ranked list of candidate titles, given the ASR transcription of a caller's book title request. The number of titles in the backend return depends on similarity scores between the ASR string and titles in the database. For the similarity score, we used Ratcliff/Obershelp (*R/O*) pattern recognition, which is the number of matching characters divided by the total number of characters (Ratcliff and Metzener, 1988),. Matching characters are those in the longest common subsequence, then recursively in the longest subsequences in the unmatched regions. For the ASR "`billies villains greatest`" the candidate titles and their R/O scores were:

- Billy Phelan's Greatest Game (0.69)
- Baseball's Greatest Games (0.44)
- More like Us: Making America Great Again (0.44)

Based on our offline pilot, we hypothesized that there would be four distinct cases: a single close match, a small set of competing matches, a larger set of more evenly matched candidates with low but better than random similarity, and no candidates above a low, non-random threshold. The R/O thresholds we selected to yield these four cases here were:

- *Singleton*: a single, good candidate ($R/O \geq 0.85$)
- *AmbiguousList*: a list of two to five moderately good candidates ($0.85 > R/O \geq 0.55$)
- *NoisyList:* a list of six to ten poor but non-random candidates ($0.55 > R/O \geq 0.40$)
- *Empty:* no titles returned at all ($R/0 < 0.40$)

In each candidate in a list, words that matched a word in the ASR appeared in a darker font, with all other words in gray-scale that reflected the degree of character overlap. For AmbiguousList, the darkest font was dark black; for NoisyList it was medium black. Note that our focus here is not on the backend query, but on the distinct types of returns. Certainly, a more finely tuned query could be implemented.

In each *session*, the caller was given a list of 20 titles to read. The acoustic quality of titles read from a list is unlikely to approximate that of a patron asking for a title. Therefore, before each session the caller was asked to read a brief synopsis of each book (taken from the library database) and to number the titles to reflect some logical grouping, such as genre or topic. Titles were then requested in that order.

Participants did two sessions at a time, reversing roles in between. They were asked to maximize a score designed to elicit cooperative behavior and to foster the development of useful strategies. For each individual title request, or *title cycle*, the wizard scored +1 for a correctly identified title, +0.5 for a thoughtful question, and -1 for an incorrect title. The caller received +0.5 for each successfully recognized title. No time limit was imposed on either the session or an individual title cycle. Figure 1 lists the 8 steps in a title cycle.

Seven undergraduate students at Hunter College participated. Two were non-native speakers of English (one Spanish, one Romanian). Each of the 21 pairs of students met for 5 trials. During each trial, one student served as wizard and the other as caller for a session of 20 title cycles, then reversed roles for a second session. The maximum number of title cycles is thus 4,200 (21 pairs × 5 trials × 2 sessions × 20 titles). Participants were allowed to end a session early. We collected data for 4,172 title cycles.

Wizard and caller sat in separate rooms where they could not overhear one another. Each was provided with a headset with microphone, and a GUI. (Audio input on the wizard's headset was disabled.) Both GUIs accepted input from a mouse. The wizard GUI also accepted input from a keyboard.

The wizard GUI presented a live feed of each ASR hypothesis, weighted by grayscale to reflect acoustic confidence. The GUI also included a search field with which to query the database. The wizard selected an ASR string for entry into the search field. Because a long title could be split by the endpointer that segments utterances, wizards could optionally select a sequence of ASR strings. Wizards could also manually edit the search field, but were encouraged not to do so. The search result was presented as a list of candidate titles on the GUI, in descending order of the (unrevealed) similarity score from the backend's retrieval function. Words in returned titles were darkened in proportion to their lexical similarity with the search terms. To offer a title to the caller, the wizard clicked on a title returned by the backend and then on a button labeled "Sure" or "Probably." Selected titles were presented to the caller through a text-to-speech component, prefixed with the word "probably" if the wizard had selected that button. To ask a question instead of selecting a candidate title, the wizard selected two or more titles the question pertained to, clicked a button labeled "Speak" and then spoke into the microphone. Questions could be of arbitrary length and content, and were recorded for offline analysis. The wizard GUI posted the success or failure of each title cycle before the next one began.

The caller GUI gave visual feedback to the caller on the full list of 20 titles to be read during the session. Titles in the list were highlighted green on success, red on failure, yellow if in progress, and not highlighted if still pending. If the caller heard a title selected by the wizard, the caller clicked on "Accept" or "Reject" to rate the wizard's accuracy. If the caller heard the wizard ask a question, the caller clicked on a judgment as to whether she could have answered it ("Can Answer" or "Cannot Answer"). Otherwise the caller clicked to indicate difficulty ("Problem") or uncertainty about the question's relevance ("Undecided").

1. ASR processes the speech and sends output to the wizard.
2. The wizard can ask the caller to repeat the title one time. The new ASR goes to the wizard.
3. The wizard queries the database either with the ASR string or with words she selects from it.
4. The database backend returns a list of candidates.
5. The wizard selects a candidate with or without high confidence, or selects one or more candidates and asks a thoughtful question intended to help identify the requested title, or gives up.
6. If the wizard selected a candidate, the caller judges its correctness. If the wizard asks a question, the caller judges its reasonableness.
7. The wizard is informed of success or failure, and prompts the caller for the next title.

Figure 1: The title cycle.

Evaluation of Wizard and Caller Behavior

Ideally, a wizard should identify the correct title when it is present among the candidates and, if possible, ask a clarifying question when it is not. Our wizards were uniformly very good (95.25% accurate; $\sigma = 1.45$) at detecting a title that was present. They fared less well, however, when the correct title was absent, a situation that occurred 28.36% of the time..

The backend never returned empty on any query, and NoisyLists were rare (2.83%). Responses were nearly evenly divided between a singleton title list (46.74%) and a list greater than one (53.26%). Moreover, every wizard saw a similar distribution of return types from the backend: singleton ($\mu = 278.57$, $\sigma = 21.16$), AmbiguousList ($\mu = 300.57$, $\sigma = 16.92$), and NoisyList ($\mu = 16.86$, $\sigma = 4.78$). The correct title was often (71.31%) in the list of candidates; 92.05% of the Singletons were the correct title, and 53.74% of the AmbiguousLists and NoisyLists contained it.

If the title was present in the backend response, wizards were very good at finding it. When the correct title appeared among the candidates on the wizard GUI (N=2986), the wizard identified it confidently (68.72%) or tentatively (26.53%), a remarkable total of 95.25% of the time. The difficulty of the wizards' task can be evaluated in part by the position of the title read by the caller within the backend response. If the backend returned multiple candidates (N=2222), the first was the correct one 41% of the time. Far less often it was the second (5.81%), third (2.61%), fourth (2.20%), or later. (The fifth through ninth accounted for 1.76%.) This should have helped the wizards, and indeed it did. In those cases where the first on the list was the correct title, wizards offered it 98.34% of the time (74.24% confidently, and 24.10% tentatively).

If the title was not present in the backend response (N=1186), however, wizards performed much less well. After the query return, the wizard was permitted one of four possible actions: *confident* (select a single title with "Sure"), *tentative* (select a single title with "Probably"), *questioning* (ask a question), or *mystified* (the wizard could not formulate a reasonable question and gave up). When the title was not present, the wizards asked a question only 22.32% of the time.

Typically our wizards were gamely tentative (67.71%) when the correct title was not among the hypotheses. Less often, they were confident (7.78%) or mystified (2.20%).

One would expect that the way the backend response appeared on the GUI would affect the wizard's action. "Appearance" here refers to the fact that any list was ranked by similarity to the ASR search string, and that words had distinct font color depending on the list type, and the degree of word overlap with the ASR. For each title, we coded the backend response to reflect the likelihood that the return contained the correct title (Singleton = 3, AmbiguousList = 2, and NoisyList = 1), and the wizard's response to reflect her certainty (confident = 3, tentative = 2, questioning = 1, and mystified = 0). The backend response proved somewhat correlated (R=0.59, $p < 2.2e\text{-}16$) with the wizard's response. Although a Singleton (N=1950) from the backend nearly always elicited a title from the wizard (85.38% confident, 13.74% tentative, 0.62% questioning, 0.26% mystified), an AmbiguousList (N=2104) from the backend substantially reduced the wizard's confidence (22.46% confident, 63.28% tentative, 13.32% questioning, 0.95% mystified). The response to NoisyStrings (N=118), was braver than might have been warranted: 9.32% confident, 52.54% tentative, 34.75% questioning, and 3.39% mystified. When the correct title was among the candidates, its *rank* (position in the list of candidates) was somewhat correlated (R=0.42) with the wizard's accuracy ($p < 2.2e\text{-}16$), that is, wizards were more likely to identify a title correctly when it was earlier on the list.

One would also expect wizards' confidence, and therefore their responses, would vary with the individual wizard. Figure 2 confirms this. The ratio of correct decisions to total decisions for each wizard was 0.69 (A), 0.67 (B), 0.66 (C), 0.67 (D), 0.69 (E), 0.69 (F) and 0.70 (G). Over all, the wizards were mostly confident (51.87%) or tentative (40.12%), rarely questioning (7.38%), and almost never mystified (0.62%). Nonetheless, one wizard almost never asked a question, and four did so only rarely. Confidence was correlated with correctness (0.65, $p < 2.2e\text{-}16$). Confident title choices (N=2164) were correct 94.73% of the time; tentative ones only 47.37%. Wizard response type also varied with the caller, as shown in Figure 3. The caller who elicited far more tentative responses and questions than any of the others was the Romanian speaker.

To understand how wizards made *correct* decisions (confident or tentative if the correct title was present, questioning or mystified if it was not), we coded wizards' correctness as correct = 1 and incorrect = 0. A linear regression model was then constructed with 10-fold cross-validation to predict wizard correctness from features available to the wizard or system. Initially we gathered 60 such features, including descriptions of the wizard GUI, how well the ASR matched the candidates and matched database entries, and how well the wizard had done thus far in the current session. Given their interdependence (e.g., different descriptions of the ASR string), preliminary processing examined correlations among the features and reduced the set to 28. The features and the feature selection process are described in detail in (Passonneau et al., Submitted).

The most significant feature in the linear regression model (root relative squared error = 73.60%) was CheckItOut's confidence in its understanding of the caller's reading of the title, which comes from the Helios confidence annotator. While this feature is not available to wizards, it is analogous to how much "sense" the ASR string made to the wizard, and could be used to constrain system behavior. In descending order, the other particularly salient features were the GUI display (Singleton, AmbiguousList, NoisyList), speech rate (faster led to lower accuracy), and on how many of the last three titles the wizard had succeeded. More candidates led to lower accuracy; more words in the ASR string led to higher accuracy. Among the features that made no contribution to the model were the wizard's or the caller's experience at the task (number of sessions to date), and the frequency with which a wizard requested the caller to repeat the title.

Discussion and Future Work

Voice search offered our wizards three types of contexts for book title requests. These translate to three opportunities for CheckItOut. When a single title was returned, wizards justifiably assumed that it was correct. In a full dialogue, CheckItOut could mimic librarians' behavior and simply report the status of the book, without confirming the title with the caller. When an AmbiguousList was returned, wizards made a tentative guess. Half the time, the title was there and the

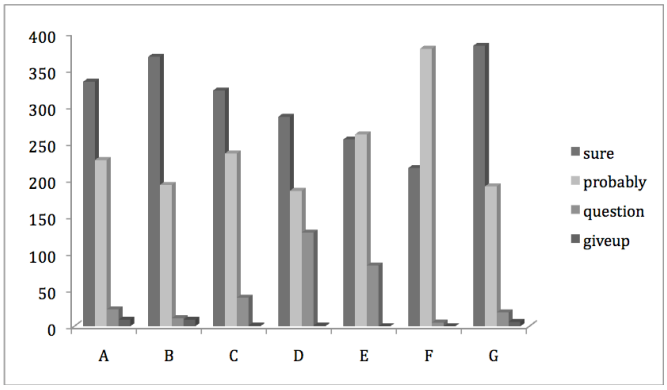

Figure 2: Distribution of acitons chosen by wizard.

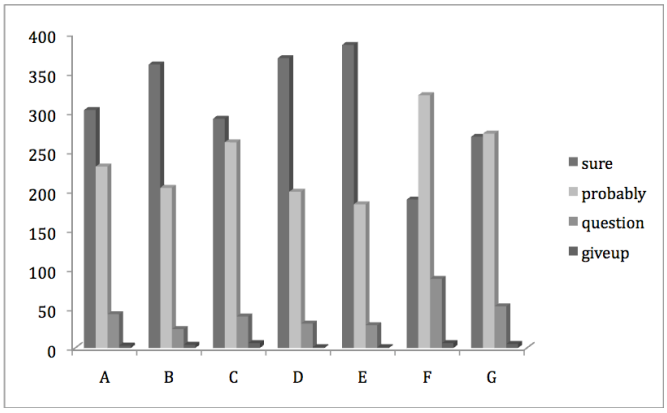

Figure 3: Distribution of actions elicited by caller.

guess was generally correct; the other half, the title was not. Here, CheckItOut could pursue one of two options: implicitly or explicitly confirm a title choice with the caller, or ask a disambiguating question. For example, given the ASR `a charge deaf`, one of our wizards was presented with two candidates: *A Charmed Death* and *A Changed Man*. She then asked "Did you say charmed or changed?" Finally, when the backend returned a NoisyList (six to ten titles), wizards often asked questions about specific words ("*Does it have orchid in it?*"), a strategy bound to be more successful, and appealing to users, than asking for a full repetition.

The focus here has been on the factors wizards attended to when they compared the ASR output to the list of candidates Extensive analysis of individual wizards is the subject of a subsequent paper currently under review (Passonneau et al., Submitted).We logged and computed many more features than those discussed here, including some that gauge the phonetic similarity of the ASR to the title. In addition, wizards and callers completed questionnaires after each session, which we will analyze, along with the wizards' questions, in future work.

Our experiment with voice search extends the WOz paradigm to allow the wizard access only to the ASR of user's utterances rather than to the acoustic input. We have shown that the integration of voice search into dialogue systems has significant promise. The accuracy of the wizards' title offers proved very high. A linear regression model based upon backend return type predicted response type (*confident, tentative, questioning, mystified*) very well. The clear differences in wizard performance bode well for our plans to learn the strategies that make a wizard proficient, and to incorporate those strategies in CheckItOut.

Acknowledgements

This research was supported in part by the National Science Foundation under IIS-084966, IIS-0745369, and IIS-0744904. We thank the staff of the Heiskell Library, our CMU collaborators Alex Rudnicky and Brian Langner and our statistical wizard Liana Epstein. Our undergraduate research assistants provided tireless enthusiasm and painstaking and thoughtful analyses.

References

Bohus, D. 2004. Error Awareness and Recovery in Task-Oriented Spoken Dialogue Systems. Pittsburgh, PA, Carnegie Mellon University.

Bohus, D., A. Raux, T. K. Harris, M. Eskenazi and A. I. Rudnicky 2007. Olympus: an open-source framework for conversational spoken language interface research. *Bridging the Gap: Academic and Industrial Research in Dialog Technology workshop at HLT/NAACL 2007*.

Bohus, D. and A. I. Rudnicky 2003. RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. *Eurospeech 2003*.

Georgila, K., K. Sgarbas, A. Tsopanoglou, N. Fakotakis and G. Kokkinakis 2003. A speech-based human-computer interaction system for automating directory assistance services. *International Journal of Speech Technology, Special Issue on Speech and Human-Computer Interaction* 6(2): 145-59.

Litman, D. J., M. A. Walker and M. S. Kearns 1999. Automatic detection of poor speech recognition at the dialogue level. I. *37th Annual ACL*, 309-316.

Passonneau, R., S. L. Epstein and J. B. Gordon 2009. Help Me Understand You: Addressing the Speech Recognition Bottleneck. *AAAI Spring Symposium on Agents that Learn from Human Teachers*, Palo Alto, CA, AAAI.

Passonneau, R., S. L. Epstein, T. Ligorio, J. Gordon, B. and P. Bhutada Submitted. Wizard strategies for resolving noisy ASR against database returns. *10th Annual Meeting on Discourse and Dialogue (SIGDIAL 2009)*.

Ratcliff, J. W. and D. Metzener 1988. *Pattern Matching: The Gestalt Approach, Dr. Dobb's Journal*.

Raux, A. 2004. Automated lexical adaptation and speaker clustering based on pronunciation habits for non-native speech recognition. *Interspeech 2004 (ICSLP)*, Jeju Island, Korea.

Raux, A. and M. Eskenazi 2007. A Multi-layer architecture for semi-synchronous event-driven dialogue management. *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 2007)*, Kyoto, Japan.

Raux, A., B. Langner, A. Black and M. Eskenazi 2005. Let's Go Public! Taking a spoken dialog system to the real world. *Interspeech 2005(Eurospeech)*, Lisbon, Portugal.

Rieser, V., I. Kruijff-Korbayová and O. Lemon 2005. A corpus collection and annotation framework for learning multimodal clarification strategies. *Sixth SIGdial Workshop on Discourse and Dialogue*. Lisbon**:** 97-106.

Rudniky, A. I., C. Bennett, et al. 2000. Task and domain specific modeling in the Carnegie Mellon Communicator System. *ICSLP 2000*, Beijing, China.

Sacks, H., E. A. Schegloff and G. Jefferson 1974. A simplest systemics for the organization of turn-taking for conversation. *Language* 50(4): 696-735.

Skantze, G. 2003. Exploring human error handling strategies: Implications for Spoken Dialogue Systems. *ISCA Tutorial and Research Workshop on Error Handling in Spoken Dialogue Systems*. Chateau-d'Oex-Vaud, Switzerland**:** 71-76.

Stoyanchev, S. and A. Stent 2009. Predicting concept types in user corrections in dialog. *EACL Workshop SRSL 2009*.

Walker, M. A., D. Litman, J., C. A. Kamm and A. Abella 1997. PARADISE: A Framework for Evaluating Spoken Dialogue Agents. *35th Annual ACL*, 271-280.

Williams, J. D. and S. Young 2004. Characterising Task-oriented Dialog using a Simulated ASR Channel. *Eight International Conference on Spoken Language Processing (ICSLP/Interspeech)*. Jeju Island, Korea**:** 185-188.