Help Me Understand You: Addressing the Speech Recognition Bottleneck

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

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

Abstract

This paper focuses on the ways dialog systems might learn better strategies to handle automatic speech recognition errors from the way people handle such errors. In the well-known Wizard of Oz paradigm to study human-computer interaction, a user participates in dialog with what she believes to be a machine, but is actually another person, the wizard. The Loqui project ablates its wizards, removing human capabilities one at a time. This paper details a pilot experiment to develop specifications for Loqui's wizard ablation studies. In the pilot task, a speaker requests books in a library application. The key finding here is that, when bolstered by a very large database of titles, humans are remarkably successful at interpreting poorly recognized output. Their repertoire of clever, domain-independent methods depends upon partial matches, string length, word order, phonetic similarity, and semantics. The long term goals of this work are to provide dialog systems with new ways to ask users for help, and to provide users with greater understanding of system functionality. Once implemented, these methods should substantially reduce human frustration with automated dialog systems, and improve task success.

SYSTEM:	`Welcome to the CMU Let's Go! Bus Information System. How may I help you?`
CALLER:	I'd like to go to Forbes and Murray.
SYSTEM:	`To Forbes and Murray. Where are you leaving from?`
CALLER:	Carnegie Mellon.
SYSTEM:	`From Carnegie Mellon. There are five buses running between Carnegie Mellon and Forbes and Murray: the 501,the 59U, the 61A, the 61B, and the 61C. What else can I do for you?`
CALLER:	When is the next one?
SYSTEM:	`The next 61C going to Forbes and Murray will leave CMU at 5:13p.m.`

Figure 1: Sample dialog from the telephone-based Let's Go dialog system developed at Carnegie Mellon University (Raux et al., 2003). The task is to provide route and schedule information about Pittsburgh Port Authority Transit buses.

Challenges in Dialog Systems

Task-oriented or information-seeking dialog between a person and a machine can be frustrating. Users want human-machine dialog to be *successful,* that is, they want to accomplish their task or to get their questions answered. They also need human-machine dialog to be *habitable*, that is, they need their tacit knowledge about how to engage in human-human dialog to carry over. Otherwise, what is the point of designing a human-computer interaction interface that relies on what humans do so naturally? Even in a successful dialog system like the one illustrated in Figure 1, however, users often chafe at the rigid paths the system pursues to achieve the dialog goals.

Figure 1 illustrates Let's Go, a system that has a relatively low word error rate (*WER*), and a relatively simple domain where users request bus route and schedule information. Even in a system like Let's Go, where the output of the automated speech recognizer may be a perfect transcription of the caller's speech – as it is here – users become easily frustrated when they are misunderstood. We believe this is in part because systems try too hard to understand a user's words and do not try hard enough to guess at what the user intends to communicate (with notable exceptions, cf. (Skantze, 2003)). The performance of automated speech recognition (*ASR*) then becomes a crucial limiting factor for dialog system performance. User frustration also arises because users have poor models of how human-machine dialog differs from human-human dialog. For example, Bohus (2004) reports that users often fail to discover important functionalities of the dialog systems with which they interact.

WOz (Wizard of Oz) is a well-known paradigm for studying human-computer interaction. A user participates in dialog with what she believes to be a machine, but is actually another person, the *wizard*. Mediated through a computer interface, the wizard receives input from the user and generates responses. Conventional WOz dialog does exhibit the same kinds of errors that occur in human-machine dialog. To investigate intelligent recovery from ASR errors, the *Loqui* project *ablates* its wizards, removing human capabilities one at a time. The pilot experiment reported here contributes to the development of specifications for wizard ablation experiments, as proposed in (Levin and Passonneau, 2006)

Figure 2 shows the range of corpora Loqui addresses. The first box represents a conventional wizard. The second represents a wizard who sees ASR instead of hearing speech. The third represents an additional ablation condition where the wizard who sees ASR must respond with dialog actions the system has available to it. Dialog between an ablated wizard and a user permits us to examine how human wizards handle obstacles to fluent human-system dialog, given the performance limitations of standard modules such as the automatic speech recognizer, the dialog manager (the brains of a dialog system), and other system components. The contrasting datasets will highlight ways in which wizards take better advantage of intermediate stages of processing in ways that systems could be engineered to do.

The pilot experiment presented here directly addresses how to guess at what a user intends to communicate, instead of guessing at the user's words. The task is to request books by title from a library. The experiment has human wizards play the role of automated librarian, by guessing which book a user requests. The wizard is given intentionally poor ASR for a spoken request, along with a large set of titles from a library database. We believe the ways a wizard guesses at the intended title and asks questions of the user can inform a new model for an automated dialog participant, one that seeks to understand the user's intent while making clear its own capabilities and shortcomings.

The next section motivates the experiment with illustrations of ASR output for several book titles in the CheckItOut dialog system. It also suggests problem-solving strategies a wizard might pursue to find the caller's intended title in the library database. Subsequent sections detail related work, provide background on the CheckItOut dialog system, describe the pilot book title experiment, present the results, and discuss their implications. The final section illustrates how this pilot experiment shaped the design of the infrastructure for a current, large-scale book title experiment.

Reasoning about Imperfect ASR

The CheckItOut dialog system addresses the more routine requests to librarians at the Andrew Heiskell Braille and Talking Book Library from its 5028 active patrons. Heiskell is part of the New York Public Library and the Library of Congress National Library Service for the Blind and Physically Handicapped (NLS). Heiskell's database includes holdings and patron information. CheckItOut substitutes artificial data for personal identification information about Heiskell's patrons, but is otherwise realistic. The holdings include 71,166 titles, comprising 29,794 distinct words; the 28,043 author names include 19,108 distinct words.

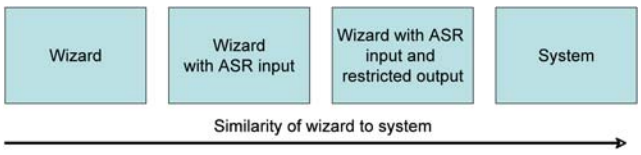

Figure 2: Wizard ablation generates a spectrum of corpora between a user and a wizard, a user and an ablated wizard, or a user and a system.

Patrons order their books by telephone and receive them by mail. (Henceforward a library patron is a *caller*.) When the line is busy or the library is closed, callers' requests go to a voicemail system with limited storage capacity. Most callers prefer to converse with a librarian, who often knows the most active callers by name and is familiar with their reading preferences. The Heiskell Library allowed us to observe how librarians handle calls, and to record approximately 175 calls. (The annotated corpus of human-human calls and our human-wizard and human-system calls will be made available to the research community at the end of this project.) In the dialogs that CheckItOut models, librarians have the caller identify herself, accept multiple book requests, confirm which books are available, and arrange to mail them to the caller.

Heiskell's callers request books in a variety of ways: by title, catalog number, author, genre. The easiest to process are requests by *RC number* (Recorded Cassette; requests for Braille format materials are relatively rare). The RC number (assigned by the NLS) is a unique identifier of four to six digits. ASR can be engineered to be very accurate for sequences of numbers. The next most common requests, those by title, are more difficult for ASR, due to the size of the vocabulary. Through access to the database, CheckItOut can retrieve titles similar to what it believes it has heard. For example, it can compare the recognizer's transcription of a book title against known titles.

To illustrate how a wizard might infer a title from poor quality ASR, we set the ASR component in CheckItOut to perform poorly (as described in the experimental design section). Table 1 lists book titles (in italics) read into CheckItOut, followed by the ASR output. The four examples in Table 1 illustrate transcription errors that vary in their degree of similarity to words in the title that was read.

Wizards can identify candidate titles with features of the ASR other than the precise words themselves, such as the number of words and the phonetic similarity, as we now explain. In example 1, the pronunciation of "the night" and "than 9" differ mainly in the final consonant: 't' versus 'n'. The database contains 37 titles that begin "Into the" but of these, only 18 are also three words long, and 'n' is the first

Table 1: Book titles (in italics) and their noisy ASR output.

1	*Into the Night* `Into than 9`
2	*Helen and Teacher: The Story of Helen Keller and Anne Sullivan Macy* `Helen an teacher distort tell until an am Sullivan Macy`
3	*Map of Bones* `Nah don't bones`
4	*I Lived to Tell it All* `Elusive total man`

letter of the last word in only two of those. In example 2, the initial consonant sounds of "the story" and "distort" (spelled 'th' and 'd') differ by only one phonetic feature; also, although the first post-consonantal vowels are spelled differently -- 'e' versus 'i' – they are pronounced the same, as the reduced vowel known as *shəwa.* Even without this similarity, a wizard can find 32 titles in the database containing the name Helen. (17 contain Helen Keller.) Only one also contains "Macy," which is the last word in that title and in the ASR string. Example 3 is more difficult to resolve. There is very little similarity between "map of" and "nah don't," apart from the fact that "m" and "n" are both nasal consonants. Eighteen titles end in "bones," but only three of them are three words long. Example 4 is the most problematic. No word in the ASR matches any word in the title. They share the same number of syllables, but it would be difficult to search the database for syllabic sequences that share only some of their sounds, such as the "l" and "v" of "I lived" versus "elusive."

Given these challenges, the experiment described here has two goals. The first is to determine how often a wizard can find the correct title given imperfect ASR. The second is to determine what kinds of situations provide an opportunity for the caller and wizard to engage in further discussion about the title to help the wizard choose among multiple candidates.

Related Work

Dialog manager implementation has recently changed from a manual process to one that applies machine learning strategies to dialog corpora (Levin, Pieraccini and Eckert, 2000; Torres, Sanchis and Segarra, 2003; Young, 2002). These approaches include learning a stochastic dialog manager from corpora that simulate relevant levels of representation for both the human and system participants (Scheffler and Young, 2002), learning dialog policies (Tetreault, Bohus and Litman, 2007), and learning error recovery strategies (Bohus, 2004). Loqui seeks to learn dialog strategies from corpora, but also considers how to identify which corpora are the best to learn from, and how to use wizard ablation to design human-wizard corpora that target dialog phenomena specific to human-system dialog.

Simulated corpora are often relied upon because they are easier to collect than human-wizard or human-human corpora. To assess the quality of simulated corpora, one group prepared a matched corpus of simulated dialogs and human-wizard dialogs (Griol et al., 2008), and compared them with a method designed to assess simulated corpora (Schatzmann, Georgila and Young, 2005). As reliance on simulated corpora for learning dialog strategies increases, better assessments of simulated corpora are essential (Ai and Litman, 2006). Most of the work cited above studies task-oriented dialog. Ai and Litman compare real-real, simulated-simulated, and simulated-real tutorial dialog corpora. They argue that the evaluation method in (Schatzmann, Georgila and Young, 2005) can discriminate real from simulated dialogs, but cannot support strong conclusions about how "realistic" the simulated corpora are.

Among the few human-wizard studies in which the wizard receives ASR instead of speech input (e.g., (Skantze, 2003; Zollo, 1999)), none have used multiple ablation conditions. (Zollo, 1999) is the earliest study we know of that investigates how human wizards perform when they are presented with ASR output instead of speech. In seven dialogs with different human-wizard pairs in which the wizard and the human were to develop an evacuation plan, the overall WER was 30%. Wizards produced utterances indicating a failure to understand in only 35% of the 227 cases of incorrect ASR. To compensate for the imperfect ASR, wizards ignored words that were not salient in the domain, and hypothesized words based on phonetic similarity.

In another experiment that explored human error handling strategies for ASR input, eight subjects played the role of system users and eight different subjects performed as operators (Skantze, 2003). This resembled a human-wizard study, but with the modification that "users" knew they were speaking with human operators. Although the WER was 43%, there were very few misunderstandings, in part because wizards could often ignore the ASR errors and continue the dialog. Operators were very good at detecting when they had enough information to infer what

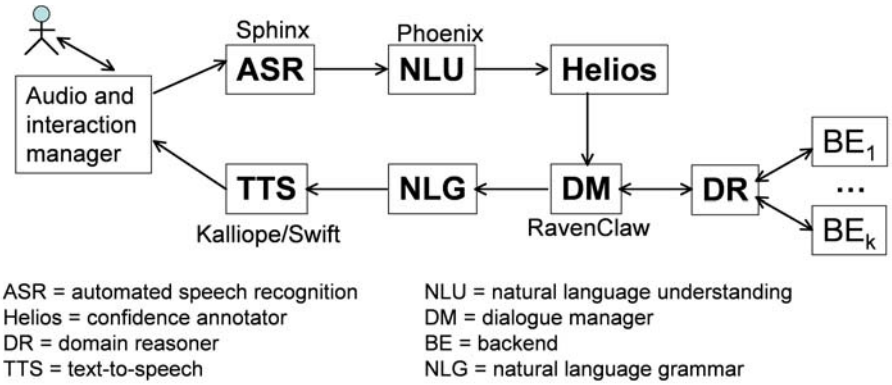

Figure 3: The Olympus/RavenClaw dialog system. Errors in automated speech recognition impact every module in the system.

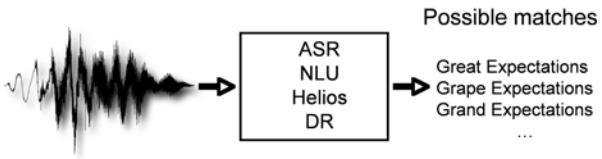

Figure 4: Sound is transformed to text that may match multiple titles in the database.

was said. The three most common operator strategies were to continue the task, to ask a task-related question, and finally, to signal non-understanding. The prevalence of the first two strategies led users to believe that they were almost always understood. In fact, signaling non-understanding was correlated inversely with how well users thought they themselves performed the task.

CheckItOut and Olympus/RavenClaw

CheckItOut incorporates the *Olympus/RavenClaw* architecture (Bohus et al., 2007; Bohus and Rudnicky, 2003). *Olympus* is a domain-independent dialog system architecture. *RavenClaw* is a dialog manager toolkit that provides an out-of-the-box environment with domain-independent error handling and requires a domain-dependent dialog task tree. Together they have been the basis for 11 research dialog systems at half a dozen sites (Bohus 2004). Figure 3 is a schematic for Olympus/RavenClaw, where the person at the left provides speech input. The sound waveforms from this speech go to the ASR module, which relies on two data sources: an *acoustic model* that maps sound segments to words and a *language model* of word sequences that are likely to be encountered. The output of the ASR is a text string. As in Figure 4, sound is transformed to text that may match multiple titles in the database

The ASR output goes to the natural language understanding (*NLU*) module, which produces a semantic representation of the text string. Together, the ASR and the NLU recognize what is said. The NLU forwards its output to Helios, which provides a confidence annotation on the NLU module's semantic output. This annotation is based on a variety of knowledge sources, such as the ASR module's confidence scores on the individual source words. The annotated NLU output then goes to the RavenClaw dialog manager, which determines what to say next given its dialog strategies, the current state of the dialog, and its interaction with the domain reasoner. CheckItOut's domain reasoner provides the dialog manager with access to its backend database. Once the dialog manager has determined what to say, CheckItOut generates text through a template-driven natural language generator. Finally, a text-to-speech module transforms the text into speech, the system's spoken participation in a dialog.

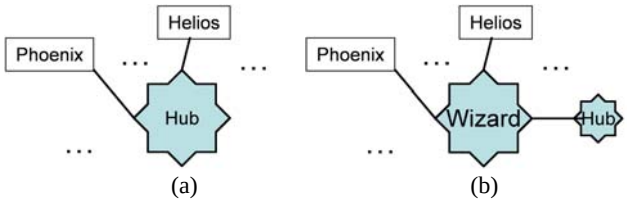

Figure 5: Rewiring (a) the Olympus/RavenClaw star topology to (b) incorporate the wizard.

Although Figure 3 suggests a pipeline, system modules actually communicate via frames passed through a hub, shown as a star topology in Figure 5(a). CheckItOut makes the wizard a required gateway to the hub as shown in Figure 5(b). Thus the wizard can intervene with respect to any message. The focus of current development is on which messages to display to the wizard, in what form, and what interventions to allow. In particular, given imperfect ASR, we seek novel ways for the system to guess at what a caller means, and novel forms of clarification subdialog.

A Title Recognition Experiment

Experimental Design

The participants in the experiment were three undergraduates (referred to here as A, B, and C), familiar with CheckItOut but not with Heiskell's holdings. This off-line pilot study used three individuals to determine whether wizards would differ in their success rates or in the kinds of strategies on which they rely.

CheckItOut's ASR was trained on the words in a randomly chosen subset of s titles, using the Olympus/RavenClaw tool *Logios* (Group, 2008). To choose a good value for s, several were examined. With s = 100, the ASR hypotheses were sufficiently good to support unambiguous matches. With s = 1000, the ASR output was impenetrable. The examples presented here, therefore, are drawn from the sample with s = 500 (1400 unique words).

The resulting language model for the speech recognizer contained only unigram frequencies (single words). The deliberate omission of bigram and trigram frequencies insured an impaired word error rate. In later experiments, baseline ASR performance will improve with a language model that is based ngram sequences of two or more words. Here, the overall performance of the ASR matters less than the creation of sets of 50 titles for which the ASR is far from perfect but far better than random noise.

Each participant received, in text format, CheckItOut's ASR output from a single speaker on a randomly chosen subset of size 50 from s, plus a text file listing all 71,166 book titles, the frequency with which each individual word in s appeared in s, and the borrowing frequency for each book. Participants were asked to identify or guess the title in each instance, and to describe how they went about it as descriptively as possible. We provided a simple search mechanism, but they were permitted to search the text file in any way they chose, with no time limit. The participants' responses were evaluated for correctness and their strategies examined.

Table 2: Distribution of responses when subjects guess the true title from the ASR output.

	Participant A		Participant B		Participant C	
Category	**Count**	**%**	**Count**	**%**	**Count**	**%**
Correct	30	66.7	33	71.7	33	71.7
Ambiguous	0.0	0.0	4	8.7	0.0	0.0
Incorrect	7	15.5	1	2.2	13	28.3
No response	8	17.8	8	17.4	0.0	0.0
Total	45	100.0	46	100.0	46	100.0

Results

The ASR rendered 9% of all 150 titles perfectly, leaving A to identify 45, and B and C to identify 46 each. Responses were categorized as correct, incorrect, *no response* (for titles where the participant offered no guess), or *ambiguous*. Ambiguous responses were titles where the participant indicated that he could not decide between a small set of titles that roughly matched the string length and a small set that matched all or part of a content word in the ASR hypothesis. Only B applied ambiguous. When he did so, it was only to ASR hypotheses of at most three words; it produced at most four titles, and only once was any alternative correct. C always provided a title response, and therefore had more explicitly incorrect responses, as shown in Table 2.

While all three participants confronted nearly the same number of ASR strings that were not exact title matches, the quality of the ASR they began with seems to have been different. The average WER compares ASR quality for each set of titles. WER is a measure frequently used to evaluate speech recognizers. It normalizes string edit distance by the length of the string. Here we use Levenshtein's string edit distance on word tokens (Levenshtein, 1996). The overall WER for the 150 titles was 0.69. Table 3 identifies the speaker who generated the ASR, and the average WER after removal of the correct titles.

The WER differences are likely due to differences in the speech quality of the reader. A and B have a non-standard American English pronunciation that is likely not represented in the recognizer's acoustic models. D is a fourth person, whose pronunciation is standard American English. In addition, D had already worked with the recognizer enough to bias his speech towards better recognition performance. Despite ASR output with higher WERs, however, B and C had a higher percentage correct, probably due to the strategies on which they relied.

Table 3: ASR details and word error rate as provided to each participant. The number of short (one or two words) ASR outputs, and the longest single ASR are given for each participant.

	Reader	WER	Short ASR	Longest ASR
A	D	0.69	12	10
B	A	0.75	7	19
C	B	0.83	8	15

Table 4: Distribution of responses by matching strategies.

	A		B		C	
Strategy	**#**	**%**	**#**	**%**	**#**	**%**
Word hits	11	24	17	37	13	28
Rarity	5	11	3	7	0	0
Word hits +location	2	4	3	7	13	28
Word hits +rarity	1	2	5	11	2	4
Word hits + rarity + location	11	24	5	11	0	0
Phonetic	8	18	6	13	1	2
Semantic	1	2	1	2	0	0
Other	6	13	6	13	17	37
Totals	45	99	46	100	45	100

There are two striking observations from the distribution of the participants' guesses into the categories shown in Table 2. First, they show a very similar percentage of correct responses. Second, the correct hit rate of about 70% suggests that application of the participants' strategies would substantially improve CheckItOut's understanding.

The participants relied on similar sets of strategies, which fall into three categories: lexical, phonetic, and semantic. Length similarity between the ASR and potential titles always applied. Lexical strategies include properties of words in the ASR hypothesis: one or more words are exact matches to words in the title (*word hits*), the word is rare (*rarity*), or the position of the word in the string discriminates possible hits (*location*). A match is *phonetic* if no word is a direct match, but words that sound or are spelled similarly do match and produce a hit (e.g., "too" and "two," or "than" and "then"). A match is *semantic* if no word is an identical match, but words that are semantically related to it do match (e.g., "truck" and "train"). Table 4 shows that these strategies were often combined. "Other" includes additional combinations of lexical, phonetic, or semantic strategies.

Clearly there will be an interaction between the usefulness of a strategy and the length of the ASR hypothesis. Table 3 shows the number of *short* (one-word or two-word) ASR outputs and the greatest number of words in an ASR output. Short ASR output presents particular challenges; fewer words offer fewer clues. The 9 one-word titles produced by the ASR for 150 books matched one-word titles in the database 60% of the time. When a one-word title had no match, participants had recourse only to their imaginations. (For example, A tried "exist" for "exit.") Two-word titles produced by the ASR permitted more creativity. Six of them matched three-word titles whose first word was "The." The remainder encouraged partial matching, usually biased toward nouns or unusual words. For example, when the ASR offered up "Kind Tailors," A searched for titles with both or either of them, and decided that "kind" and "nine" sounded similar enough to choose "The Nine Tailors" correctly. B used both location and word hits on his two-word titles, and thereby found multiple matches. For example, the ASR output "Tandem Betrayal" produced one "unlikely" possibility with "tandem," but titles ending in "betrayal" suggested the correct "Ten-

Table 5: A noisy two-word ASR output has five two-word title matches based on a single word.

Title	Women's *Place*
ASR output	*Wilderness Place*
Guess 1	*Wilderness* Peril
Guess 2	*Wilderness* Tips
Guess 3	*Wilderness* Trek
Guess 4	Waverly *Place*
Guess 5	Women's *Place*

der Betrayal." C eliminated an extraneous second word based on title similarity to the first word. Nonetheless, matching even on a single word in a two-word ASR output was not always successful. For example, the ASR output for "Road to Wealth" was "Roll Dwell" and the output for "The Odes of Pindar" was "People Exit." Phonetics failed C in the latter, where his guess based on initial and final sounds with "very low confidence" was "Poultry in the Pulpit."

Table 5 is an example drawn from one of B's responses that was classified as ambiguous. The correct title appears in the first row. The guesses include three matches on the first word and one two matches on the second word. The correct title is one of the two with exact matches on the second word. The wizard's task here seems clear: to resolve whether either word is correct, and if so, to disambiguate the relevant subset. One can, however, imagine a broad range of subdialogs to address this task. This is the kind of clarification dialog we intend to elicit from our ablated wizard, and to acquire by machine learning.

All three participants intuitively focused upon content words rather than *stop words* (those with high frequency and low information content, such as an article or preposition). Given an ASR hypothesis of three or more words, participants interpreted several matching content words in a retrieved title as sufficient evidence. Whenever search on several content words produced only a single hit in the database, participants assumed that hit was correct. For example, given "China Life Could Techniques No Hard Above Dragon," a search for just "china" and "dragon" yielded "China Live: Two Decades in the Heart of the Dragon." Similarly, "Cosmos Coyote Can William Bloom Ice" matched three of the six words "Cosmos Coyote and William the Nice." The more content words that matched, the more comfortable the subjects were with the hit.

The right content words to search on are by no means obvious, however. For example, to match "Trouble Pennsylvania an This Session Reilly Mystery,**"** B searched first on "Pennsylvania" and "mystery" (to no avail), but a search on "Reilly" and "mystery" produced the correct "Trouble in Transylvania: A Cassandra Reilly Mystery." Similarly, ASR output "Big Politics Diplomacy Revolution or Antes" instigated A's unsuccessful search for "Big Politics" and then for "Politics" and "Diplomacy." The only hit, and therefore the guess, was "The Policy of Diplomacy: Revolution, War and Peace 1989 -1992." The incorrect guess, "Successful Job Search Strategies for the Disabled," was produced for the ASR output "Job Search RC 8 1 Disability"; the correct title was "Job Search Handbook for People with Disabilities." Occasionally our participants outsmarted themselves with these devices. For example, in response to ASR output that read "Portable Western Reader," A assumed that "Western" was probably misheard, and searched only on "Portable" and "Reader." The hit on "The Portable Western Reader" surprised him.

For titles of three to five words, A liked matches based on word or syllable count. For example, in response to "What Heart Into" he searched for some combination of those words and (incorrectly) chose "What the Heart Knows" because it was "about the same number of words." B added to the mix the word's location (at the beginning or the end of the title, searched for with regular expressions). For example, to decipher "Baby Us Eyes Projects" he retrieved titles that began with "baby" or ended with "projects," and found "the only title that looked even remotely possible," "Ideas for Science Projects."

Similar sounds were used to sort through multiple possibilities retrieved from recognized nouns. For example, "Gates of Care Venice" instigated a search for combinations of "gates," "care," and "Venice," and then titles that ended in Venice. A chose "The Ghetto of Venice" because it sounded like the ASR output. A similar process matched "Want sinking ship to" with "Dance on a Sinking Ship," and "It Double Insight Liston" with "The Devil and Sonny Liston." For "Affair Summer Tumbling Sun Our Dies of Benjamin Franklin," B diligently looked at the many titles that ended in "Benjamin Franklin," and "scoured for the initial 'f' sound" to find "The First American: The Life and Times of Benjamin Franklin." Only C considered borrowing frequency; he used "popularity" to select among alternatives 9 times, but this device succeeded only once. Subsequent experiments will select *s* more realistically, biased on title circulation frequency.

Discussion

This paper recounts Loqui's first probe experiment to support the design of an ablated wizard module and a domain-specific reasoner. The probe was directed at one of the nodes in our dialog manager's task hierarchy, book request by title, but the results will be relevant to other nodes that involve retrieval from the database. The findings illuminate two types of knowledge relevant to this node: search methods to query the database and filtering methods to select among likely hits.

For the search methods, the distribution in Table 2 shows that a person can find the correct title from imperfect ASR approximately two-thirds of the time. This bodes well for the hypothesis that CheckItOut can learn new dialog strategies from human wizards who reason about imperfect ASR. Unless there is something special about our participants, the pilot results suggest that our ablated wizards are likely to achieve a much higher success rate at title retrieval based on imperfect ASR than a system that uses a conventional dialog strategy. Moreover, to the degree that one can automatically learn these strategies from

corpora, or engineer them directly, it should be possible for our final version of CheckItOut to achieve a high success rate for certain types of user queries.

To model some of our participants' successful methods, we intend to build search techniques directly into our baseline system. Recall that on one-word strings, attempts to match the database succeeded two-thirds of the time. In this experiment, attempts to match one or two-word strings (25% of the ASR responses, e.g., "Photo Finish") to the database succeeded 36% of the time. We will provide the same database query strategy to our baseline system and our wizard module. For short titles with only one hit, this will allow the system to speak to the caller more informatively. For example, a response like "I think you said, 'Century.' Is that correct?" infuriates many users. A more helpful clarification request would be "I think you said, 'Century' but we have no books with that title." This indicates to the caller that the system has already tried to find the title and needs the user's assistance.

We conclude from these results that when a wizard indicates to the system that the caller's utterance is intended to be a title, the system should query the database and portray its hypothesis to the wizard in one the following ways:

• A single title that is roughly the same length as the ASR hypothesis with exact content word matches emphasized (e.g., in boldface).

• No more than five titles, each roughly the same length as the system's hypothesis, with exact matches in some parts of the title, a high-confidence ambiguous return.

• A *pseudo-string* (a cloud around one or more icons for words from the hypothesis) to show words that match many titles of about the same length as the hypothesis.

• A symbol for query failure.

Each case provides an opportunity for clarification or collaborative problem-solving with the caller. For example, in the first display, the wizard can present the title to the caller with more or less confidence, depending on factors such as how many words in the title are an exact match to the similarly positioned word in the ASR. The second display provides a filter, where the wizard can choose among titles, possibly with the caller's assistance. The third display potentially allows the wizard to re-elicit part of the title in a manner that avoids asking the caller to repeat herself, or that indicates the motivation for such a request. In general, when callers to automated dialog systems are asked to repeat their utterances, they hyperarticulate, and thereby confuse the recognizer.

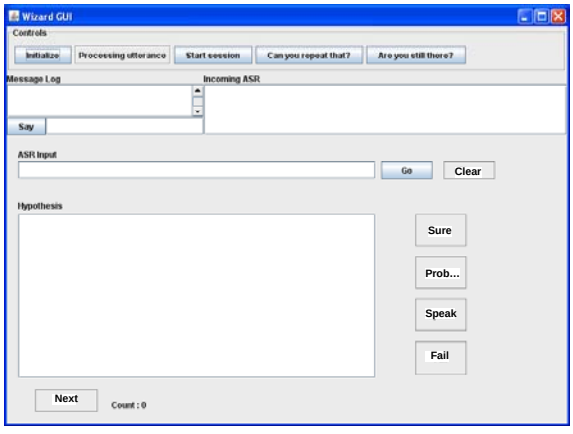

Figure 6: Wizard interface for follow-on book title experiment conducted online.

Current and Future Work

To test and extend this work, we have begun a large-scale experiment in which two people perform the same task online. Participants alternately play the roles of caller and wizard. The caller and the wizard sit in separate rooms and communicate via GUIs. The wizard GUI gets input directly from the CheckItOut ASR, domain reasoner, and the backend database, via the altered star topology illustrated in Figure 5.

The wizard interface for the new experiment is shown in Figure 6. During each session, a caller requests twenty titles. After a session is initiated, and before the caller terminates it, the interface supports a sequence of three turns:

1. The caller speaks a title. The ASR transcription appears on the wizard's screen in the upper right.
2. The wizard initiates a default backend query that compares ASR strings to titles using a string comparison function. The query return appears as a list of one or more titles in the hypothesis pane on the lower left. The wizard then selects a response to the caller from four buttons on the lower right. The wizard can offer a retrieved title with high confidence; can offer a retrieved title with lower confidence; can ask the caller a question; or can give up on the current title.
3. The caller then judges the wizard's response. If the wizard offered a title (with high or low confidence), the caller indicates whether the retrieved title is correct or not. If the wizard asked a question, the caller judges the question from a menu to indicate whether the caller would be able to answer the question. If the wizard gives up, the caller gives acknowledgement. All messages from the caller other than the ASR (e.g., "ready to start") appear in a message log in the upper left pane, which flashes green when a new message appears.

Conclusion

Our key finding is that people are remarkably successful at interpreting ASR output when bolstered by a very large database of possible strings. To do so, they rely upon a repertoire of clever, domain-independent methods that depend upon partial matches, string length, word order, phonetic similarity, and semantics. Once implemented, these methods will substantially improve a dialog system's ability to deal with poor ASR, and will support less frustrating human-computer dialog. When the system does not under-

stand, these methods both provide new ways to ask the user for help and provide the user with greater understanding of system functionality.

Because Heiskell is part of the New York Public Library and the National Library Service, any improvements CheckItOut can point towards for its patrons have potentially wide-ranging impact. Moreover, the use of these methods is not limited to a database of books — it should be deployable against any targeted database. For dialog systems in other domains, these strategies would apply to queries against a database of items the speaker targets, and the dialog manager would have to recognize when to apply this type of strategy.

Acknowledgements

This research was supported in part by the National Science Foundation under IIS-0745369 and IIS-0744904. We thank the staff of the Heiskell Library and our CMU collaborators Alex Rudnicky and Brian Langner for their continued support and cooperation, and our tireless research assistants, Carnegie Castillo, Kenneth Cordero, William Ng, Davis Quintanilla, and Allan Zelener, for their painstaking and thoughtful analyses.

References

Carnegie Mellon University Speech Group. 2008. The Logios Tool. https://cmusphinx.svn.sourceforge.net/svnroot/cmusphinx/trunk/logios

Ai, H. and D. J. Litman 2006. Comparing Real-Real, Simulated-Simulated, and Simulated-Real Spoken Dialogue Corpora. In *Proceedings of AAAI Workshop Statistical and Empirical Approaches for Spoken Dialogue Systems*, Boston, USA.

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. In *Proceedings of 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. In *Proceedings of Eurospeech 2003*.

Griol, D., L. F. Hurtado, E. Segarra and E. Sanchis 2008. Acquisition and evaluation of a dialog corpus through WOz and dialog simulation techniques. In *Proceedings of Conference on Language Resources and Evaluation (LREC) 2008*.

Group, C. M. U. S. 2008. The Logios Tool.

Levenshtein, A. 1996. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady* 10(8): 707-710.

Levin, E. and R. Passonneau 2006. A WOz variant with contrastive conditions. In *Proceedings of Interspeech Satellite Workshop, Dialogue on Dialogues: Multidisciplinary Evaluation of Speech-based Interactive Systems*, Pittsburgh, PA.

Levin, E., R. Pieraccini and W. Eckert 2000. A Stochastic Model of Human-Machine Interaction for Learning Dialog Strategies. *IEEE Trans. on Speech and Audio Processing* 8(1): 11-23.

Schatzmann, J., K. Georgila and S. Young 2005. Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems.,. In *Proceedings of SIGDial Workshop' 05*, 45–54. Lisbon, Portugal.

Scheffler, K. and S. J. Young 2002. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In *Proceedings of HLT 2002*, San Diego, USA.

Skantze, G. 2003. Exploring human error handling strategies: Implications for spoken dialogue systems. In *Proceedings of Error Handling in Spoken Language Dialogue Systems*, International Speech Communication Association.

Tetreault, J. R., D. Bohus and D. J. Litman 2007. Estimating the Reliability of MDP Policies: A Confidence Interval Approach. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, Rochester, NY.

Torres, F., E. Sanchis and E. Segarra 2003. Development of a stochastic dialog manager driven by semantics In *Proceedings of EuroSpeech'03*, 605–608.

Young, S. 2002. The Statistical Approach to the Design of Spoken Dialogue Systems., Cambridge University Engineering Department.

Zollo, T. 1999. A study of human dialog strategies in the presence of human recognition errors. In *Proceedings of AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*.