Embedded Wizardry

Rebecca J. Passonneau[1], **Susan L. Epstein**[2,3], **Tiziana Ligorio**[3] and **Joshua Gordon**[1]
[1]Columbia University
New York, NY, USA
(becky|joshua)@cs.columbia.edu
[2,3]Hunter College
[3]The Graduate Center of the City University of New York
New York, NY, USA (susan.epstein@hunter|tligorio@gc).cuny.edu

Abstract

This paper presents a progressively challenging series of experiments that investigate clarification subdialogues to resolve the words in noisy transcriptions of user utterances. We focus on user utterances where the user's specific intent requires little additional inference, given sufficient understanding of the form. We learned decision-making strategies for a dialogue manager from run-time features of our spoken dialogue system and from observation of human wizards we had embedded within it. Results show that noisy ASR can be resolved based on predictions from context about what a user might say, and that dialogue management strategies for clarifications of linguistic form benefit from access to features from spoken language understanding.

1 Introduction

Utterances have literal meaning derived from their linguistic form, and pragmatic intent, the actions speakers aim to achieve through words (Austin, 1962). Because the channel is usually not noisy enough to impede communication, misunderstandings that arise between adult human interlocutors are more often due to confusions about intent, rather than about words. Between humans and machines, however, verbal interaction has a much higher rate of linguistic misunderstandings because the channel is noisy, and machines are not as adept at using spoken language. It is difficult to arrive at accurate rates for misunderstandings of form versus intent in human conversation, because the two types cannot always be distinguished (Schlangen and Fern'andez, 2005). However, one estimate of the rate of misunderstandings of literal meaning between humans, based on text transcripts of the British National Corpus, is in the low range of 4% (Purver et al., 2001), compared with a 30% estimate for human-computer dialogue (Rieser and Lemon, 2011). The thesis of our work is that misunderstandings of linguistic form in human-machine dialogue are more effectively resolved through greater reliance on context, and through closer integration of spoken language understanding (SLU) with dialogue management (DM). We investigate these claims by focusing on noisy speech recognition for utterances where the user's specific intent requires little additional inference, given sufficient understanding of the form.

This paper presents three experiments that progressively address SLU methods to compensate for poor automated speech recognition (ASR), and complementary DM strategies. In two of the experiments, human *wizards* are embedded in the spoken dialogue system while run-time SLU features are collected. Many wizard-of-Oz investigations have addressed the noisy channel issue for SDS (Zollo, 1999; Skantze, 2003; Williams and Young, 2004; Skantze, 2005; Rieser and Lemon, 2006; Schlangen and Fern'andez, 2005; Rieser and Lemon, 2011). Like them, we study how human wizards solve the joint problem of interpreting users' words and inferring users' intents. Our work differs in its exploration of the role context can play in the literal interpretation of noisy language. We rely on knowledge in the backend database to propose candidate linguistic forms for noisy ASR.

Our principal results are that both wizards and our

Proceedings of the SIGDIAL 2011: the 12th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 248–258,
Portland, Oregon, June 17-18, 2011. ©2011 Association for Computational Linguistics

SDS can achieve high accuracy interpretations, indicating that predictions about what the user might be saying can play a significant role in resolving noise. We show it is possible to achieve low rates of unresolved misunderstanding, even at word error rates (WER) as poor as 50%-70%. We achieve this through machine learned models of DM actions that combine standard DM features with a rich number and variety of SLU features. The learned models predict DM actions to determine whether a reliable candidate interpretation exists for a noisy utterance, and if not, what action to take. The results support an approach to DM design that integrates the two problems of understanding form and intent.

The next sections present related work, our library domain and our baseline SDS architecture. Subsequent sections discuss the SLU settings across the three experiments, and present the experimental designs and results, discussion and conclusion.

2 Related Work

Previous WOz studies of wizards' ability to process noisy transcriptions of speaker utterances include the use of real (Skantze, 2003; Zollo, 1999) or simulated ASR (Kruijff-Korbayová et al., 2005; Williams and Young, 2004). WOz studies that directed their attention to the wizard include efforts to predict: the wizard's response when the user is not understood (Bohus 2004); the wizard's use of multimodal clarification strategies (Rieser and Lemon, 2006; Rieser and Lemon, 2011); and the wizard's use of application-specific clarification strategies (Skantze, 2003; Skantze, 2005). WOz studies that address real or simulated ASR reveal that wizards can find ways to not respond to utterances they fail to understand (Zollo, 1999; Skantze, 2003; Kruijff-Korbayová et al., 2005; Williams and Young, 2004). For example, they can prompt the user for an alternative attribute of the same object. Our work differs in that we address clarifications about the words used, and rely on a rich set of SLU features. Further, we compare behavior across wizards. Our SDS benefits from models of the most skilled wizards.

To limit communication errors incurred by faulty ASR, an SDS can rely on strategies to detect and respond to incorrect recognition output (Bohus, 2004). The SDS can repeatedly request user confirmation to avoid misunderstanding, or ask for confirmation using language that elicits responses from the user that the system can handle (Raux and Eskenazi, 2004). When the user adds unanticipated information in response to a system prompt, two-pass recognition can rely on a concept-specific language model to improve the recognition of the domain concepts within the utterance containing unknown words, and thereby achieve better recognition (Stoyanchev and Stent, 2009). An SDS could take this approach one step further and use context-specific language for incremental understanding of noisy input throughout the dialogue (Aist et al., 2007).

Current work on error recovery and grounding for SDS assumes that the primary responsibility of a dialogue management strategy is to understand the user's intent. Errors of understanding are addressed by ignoring the utterances where understanding failures occur, asking users to repeat, or pursuing clarifications about intent. These strategies typically rely on knowledge sources that follow the SLU stage. The RavenClaw dialogue manager, which represents domain-dependent (task-based) DM strategy as a tree of goals, triggers error handling by means of a single confidence score associated with the concepts hypothesized to represent the user's intent (Bohus and Rudnicky, 2002; Bohus and Rudnicky, 2009). Features for reinforcement learning of MDP-based DM strategies include a few lexical features and a measure of noise analogous to WER (Rieser and Lemon, 2011). The WOz studies reported here yield learned models of specific actions in response to noisy input, such as whether to treat a candidate interpretation as correct, or to pursue one of many possible clarification strategies, including clarifications of form or intent. These models rely on relatively large numbers of features from all phases of spoken language understanding, as well as on typical dialogue management features.

3 CheckItOut

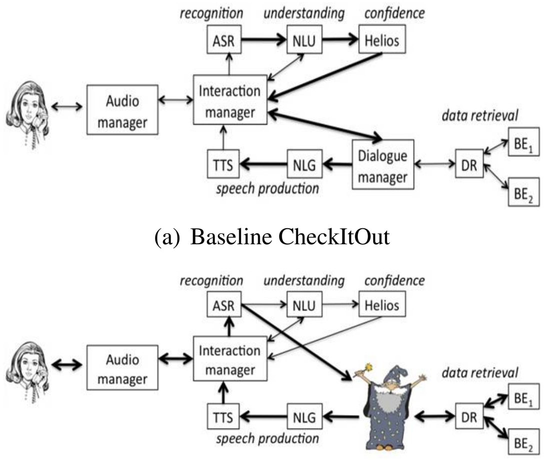
3.1 Domain

Our domain of investigation simulates book orders from the Andrew Heiskell Braille and Talking Book Library, part of the New York Public Library and the Library of Congress. Patrons order books by telephone during conversation with a librarian, and receive them by mail. Patrons typically have identifying information for the books they seek, which they get from monthly newsletters. In a corpus of eighty two calls recorded at the library, we found that most book requests by title were very faithful to the actual title. Challenges to SLU in this domain include the size of the database, the size of the vocabulary, and the average sentence length.

While large databases have been used for investigations of phonological query expansion (Georgila et al., 2003), much of the research on DM strategy relies on relatively small databases. A recent study of reinforcement learning of DM strategy modeled as a Markov Decision Process reported in (Rieser and Lemon, 2011) relies on a database of 438 items. In (Gordon and Passonneau, 2011) we compared the SLU challenges faced by CheckItOut and the Let's Go bus schedule information system, both of which rely on the same architecture (Raux et al., 2005). The Let's Go corpus contained 70 bus routes names and 1300 place names, and a mean utterance length of 4.4 words. The work reported here uses the full 2007 version of Heiskell's database of 71,166 books and 28,031 authors, and a sanitized version of its 2007 patron database of 5,028 active patrons. Authors and titles contribute 45,636 distinct words, with a 10.43% overlap between the two. Average book title length is 5.4 words; 26% of titles are 1-2 words, 44% are 3-5 words, 20% are 6 to 10. Consequently, our domain has relatively long utterances. The syntax of book titles is much richer than typical SDS slot fillers, such as place or person names.

To achieve high-confidence SLU, we integrate voice search into the SLU components of our two SDS experiments (Wang et al., 2008).[1] Our custom voice search query relies on Ratcliff/Obershershelp (R/O) pattern matching (Ratcliff and Metzener, 1988), the ratio of the number of matching characters to the total length of both strings. This simple metric captures gross similarities without overfitting to a specific application domain. The criteria for selecting R/O derive from our first offline experiment, described in Section 4.2.

For an experiment focused only on a single turn exchange beginning with a user book request, we queried the backend directly with the ASR string. For a subsequent experiment on full dialogues, we queried the backend with a modified ASR string, because the SDS architecture we used permits backend queries to occur only during the dialogue management phase, after natural language understanding. The next section describes this architecture.



(a) Baseline CheckItOut

(b) Embedded Wizard

Figure 1: CheckItOut information pipeline

3.2 Architecture

CheckItOut, our baseline SDS, employs the Olympus/RavenClaw architecture developed at Carnegie Mellon University (CMU) (Raux et al., 2005; Bohus and Rudnicky, 2009). SDS modules communicate via message passing, controlled by a central hub. However, the information flow is largely a pipeline, as depicted in Figure 1(a). The PocketSphinx recognizer (Huggins-Daines et al., 2006) receives acoustic data segmented by the audio manager, and passes a single recognition hypothesis to the Phoenix parser (Ward and Issar, 1994). Phoenix sends one or more equivalently ranked semantic parses to the Helios confidence annotator (Bohus and Rudnicky, 2002), which selects a parse and assigns a confidence score. The Apollo interaction manager (Raux and Eskenazi, 2007) monitors the three SLU modules–the recognizer, the semantic parser, and the confidence annotator–to determine whether the user or SDS has the current turn. To a limited degree, Apollo can override the early segmentation decisions based solely on pause length.

[1] In concurrent work on a new SDS architecture, we use ensembles of SLU strategies (Gordon and Passonneau, 2011; Gordon et al., 2011).

Confidence-annotated concepts from the semantic parse are passed to the RavenClaw DM, which decides when to prompt the user, present information to her, or query the backend database.

A wizard server communicates with other modules via the hub, as shown in Figure 1(b). For each wizard experiment, we constructed a graphical user interface (GUI). Wizard GUIs display information for the wizard in a manageable form, and allow the wizard to query the backend or select communicative actions that result in utterances directed to the user. Figure 1(b) shows an arrow from the speech recognizer directly to the wizard: the recognition string has been vetted by Apollo before it is displayed to the wizard.

4 Experiments and Results

The experiments reported here are an off-line pilot study to identify book titles under worst case recognition (Title Pilot), an embedded WOz study of a single turn exchange involving book requests by title (Turn Exchange), and an embedded WOz study of dialogues where users followed scenarios that included four books at a time (Full WOz). To evaluate the impact of learned models of wizard actions from the Full WOz wizard data, we evaluated CheckItOut before and after the dialogue manager was enhanced with wizard models for specific actions.

4.1 Experimental Settings

All three experiments use the full database for search. To control for WER, the knowledge sources for speech recognition and semantic parsing vary across experiments. For each experiment, Table 1 indicates the acoustic model (AM) used, the number of hours of domain-specific spontaneous speech used for AM adaptation, the number of titles used to construct the language model (LM), the type of LM, the type of grammar rules in the Phoenix book title subgrammar, and average WER as measured by Levenstein word edit distance (Levenshtein, 1996).

For the first two experiments, we used CMU's Open Source WSJ1 dictation AMs for wideband (16kHz) microphone (dictation) speech. For Full WOz we adapted narrowband (8kHz) WSJ1 dictation speech with about eight hours of data collected from Turn Exchange and two hours of scripted spontaneous speech typical of CheckItOut dialogues.

Logios is a CMU toolkit for generating a pseudo-corpus from a Phoenix grammar. It produces a set of strings generated by Phoenix production rules, which in turn are used to build an LM (Carnegie Mellon University Speech Group, 2008). Before we explain the three rightmost columns in Table 1, we first briefly describe Phoenix, the Phoenix book title subgrammar, and how we combine title strings with a Logios pseudo-corpus.

Phoenix is a context-free grammar (CFG) parser that produces one or more semantic frames per parse. A semantic frame has slots, where each slot is a concept with its own CFG productions (subgrammar). To accommodate noisy ASR, the parser can skip words between frames or slots. Phoenix is well-suited for restricted domains, where a frame represents a particular type of subdialogue (e.g., ordering a plane ticket), and slots represent constrained concepts (e.g., departure city, destination city). Phoenix is not well-suited for book titles, which have a rich vocabulary and syntax, and no obvious component slots. The CFG rules for the Turn Exchange book title subgrammar consisted of a verbatim rule for each book title. Rules that consisted of a bag-of-words (BOW; i.e., unordered) for each title proved to be too unconstrained.[2] In Turn Exchange, interpretation of ASR consisted primarily of voice search; the highly constrained CFG rules (exact words in exact order) had little impact on performance. For baseline CheckItOut dialogues, and for Full WOz, we required more constrained grammar rules that would preserve Phoenix's robustness to noise.

To avoid the brittleness of exact string CFG rules, and the massive over-generation of BOW CFG rules, we wrote a transducer that mapped dependency parses of book titles to CFG rules. When ASR words are skipped, book title parses can consist of multiple slots. We used MICA, a broad-coverage dependency grammar (Bangalore et al., 2009) to parse the entire book title database. When a set of titles is selected for an experiment, the corresponding MICA parses are transduced to the relevant CFG productions, and inserted into a Phoenix grammar. Productions for the *author* subgrammar

[2] BOW Phoenix rules for book titles are used in a more recent Olympus/RavenClaw system inspired in part by CheckItOut (Lee et al., 2010), with a database of 15,088 eBooks.

Exp.	AM	Adapted	# Titles for LM	LM	Grammar rules	WER
Title Pilot	WSJ1 16kHz	NA	500	unigram	NA	0.76
Turn Exchange	WSJ1 16kHz	NA	7,500	trigram	title strings	0.71
Full WOz	WSJ1 8kHz	10 hr.	3,000	Logios + book data	Mica-based	0.50 (est)

Table 1: SLU settings across experiments

consist largely of a first name slot followed by a last name slot. The remaining portions of the Phoenix CheckItOut grammar consist of subgrammars for *book request* prefixes and affixes (e.g., "I would like the book called"), for *confirmations* and *rejections*, *phone numbers*, *book catalogue numbers*, and miscellaneous additional concepts. The set of subgrammars excluding the book title and author subgrammars (*book requests, confirmations*, and so on; the grammar *shell*) are the same for all experiments. The MICA-based book title grammar also provides several features (e.g., number of slots in a parse) for machine learning.

The Title Pilot LM consisted of unigram frequencies of the 1400 word types from a random sample (without replacement) of 500 titles. For Turn Exchange, a trigram LM was constructed from 7,500 titles randomly selected from the 19,708 titles that remained after we eliminated one-word titles and titles with below average circulation. For Full WOz, 3,000 books were randomly selected from the full book database (with no more than three titles by the same author, and no one-word titles). Logios was used on the grammar shell to generate an initial pseudo-corpus, which was combined with the book title and author strings to generate a full pseudo-corpus for the trigram LM (denoted as "Logios + book data" in Table 1).

4.2 Title Pilot

The Title Pilot (Passonneau et al., 2009) was an offline investigation of how reliance on prior knowledge in the database might facilitate interpretation of noisy ASR. It demonstrates that given the context of things a user might say, ASR that is otherwise unintelligible becomes intelligible.

Three males each read 50 randomly selected titles from the LM subset of 500 (see Table 1). Their average WER was 0.75, 0.83 and 0.69, respectively. Three undergraduates (A, B, C) were each given one of the sets of 50 recognition strings from a different speaker. Each also received a plain text file listing all the titles in the database, and word frequency statistics for the book titles. Their task was to try to find the correct title, and to provide a brief description of their overall strategy.

A was accurate on 66.7% of the titles he matched, B and C on 71.7%. We identified similar strategies for A and B, including number of exact word matches, types of exact word matches (e.g., content words were favored over stop words), rarity of exact word matches, and phonetic similarity. Analysis of C's responses showed dependency on number and types of exact word matches, and on miscellaneous strategies that could not be grouped. Through inspection, we determined that similarity in length and number of words were important factors. From this experiment, we concluded that humans are adept at interpreting noisy ASR when provided with context; that voice search (queries to the backend with ASR) would prove useful, given an appropriate similarity metric; and that there would likely always be uncertain cases that might lead to false hits. As we discuss below, two of seven Turn Exchange wizards were fairly adept, and five of six Full WOz wizards were very adept, at avoiding false hits from voice search.

4.3 Turn Exchange

The offline Title Pilot suggested that voice search could lead to far fewer non-understandings, given some predictions as to the actual words a noisy ASR string might represent. The next experiment addressed, in real time, the question of what level of accuracy might be achieved through an online implementation of voice search for book requests by title (Passonneau et al., 2010; Ligorio et al., 2010b). We embedded wizards into the CheckItOut SDS to present them with live ASR, and to collect runtime recognition features. On the GUI, variations in the display fonts for ASR and voice search returns cued the wizard to gross differences in word-level recognition confidence, and similarities between an ASR string and each candidate returned by the search. Learned models of wizard actions indicated that recognition features such as acoustic model fit and speech rate, along with various measures of similarity between the ASR output string and candidate titles, number of books ordered thus far (RecentSuccess), and number of relatively close candidate matches, were useful in modeling the most accurate wizards. These results show that DM strategy for determing what actions to take, given an interpretation of a user request, can depend on subtle recognition metrics.

In Turn Exchange, users requested books by title from embedded wizards. Speech input and output was by microphone and headset, with wizards and users seated in separate rooms, each using a different GUI. Seven undergraduates (one female and six males, including two non-native speakers of English) participated as paid subjects. Each of the 21 possible pairs of students met for five trials. A trial had two sessions. In the first, one student served as wizard and the other as user for a session in which the user requested 20 books by title. In the second session, the students reversed roles. We collected 4,192 turn exchanges.

The GUI displayed the ASR corresponding to the user utterance, with confident words in bolder font. The wizard could query the backend with some or all of the ASR. Voice search results displayed a single candidate above a high R/O threshold with all matching words in boldface, or three candidates of moderate similarity with matching words in medium bold, or five to ten candidates of lower similarity in grayscale. There were four available wizard actions: to offer a candidate title to the user in a confident manner (through Text-to-Speech), to offer a title tentatively, to select two or more candidates and ask a free-form question about them (here the user would hear the wizard's speech), or to give up. The user indicated whether an offered candidate was correct, or indicated the quality and appropriateness of a wizard's question. A prize would go to the wizard who offered the most correct titles.

The top ranked search return was correct 65.24% of the time. The two wizards who most often offered the top ranked return (81% and 86% of the time) both achieved 69.5% accuracy. The two best wizards (W4 and W5) could detect search returns that did not contain the correct title, thus avoiding false hits. On average, they offered the top return only 73% of the time and both achieved the highest accuracy (83.4%).

Several classification methods were used to predict the four wizard actions: firm offer, tentative offer, question, and give up. Features (N=60) included many ASR metrics, such as word-level confidence, AM fit, and three measures of speech rate; various measures of the average similarity or overlap between the ASR string and the candidate titles from the R/O query; the dialogue history; the number of candidates titles returned; and so on. The learned classifiers, including C4.5 decision trees (Quinlan, 1993), all had similar performance. Learned trees for W4 and W5 both had F measures of 0.85. Decision trees give a transparent view of the relative importance of features; those nearer the root have greater discriminatory power. Common features at the tops of trees for all wizards were the type and size of the query return, how often the wizard had chosen the correct title in the last three title cycles, the average of the maximum number of contiguous exact word matches between the ASR string and the candidate titles, and the Helios confidence score.

We trained an additional decision tree to learn how W4 (the best wizard) chose between offering a title versus asking a question (F=0.91 for making an offer; F=0.68 for asking a question). The tree is distinctive in that it splits at the root on a measure of speech rate. If the ASR is short (as measured both by the number of recognition frames and the words), W4 asks a question if the query return is not a single title, and either RecentSuccess=1 or ContiguousWord-Match=0, and the acoustic model score is low. Note that shorter titles are more confusable. If the ASR is long, W4 asks a question when ContiguousWordMatch=1, RecentSuccess=2, and either CandidateDisplay = NoisyList, or Helios Confidence is low, and there is a choice of titles.

4.4 Full WOz

The third experiment was a full WOz study demonstrating that embedded wizards could achieve high task success by relying on a large number of actions that included clarifications of utterance form or intent. Here we briefly report results on task success and time on task in a comparision of baseline CheckItOut with an enhanced version, CheckItOut+, that incorporates learned models of wizard actions. The evaluation demonstrates improved performance with more books ordered, more correct books ordered, and less elapsed time per book, or per correct book.

For Full WOz (Ligorio et al., 2010a), CheckItOut relied on VOIP (Voice over Internet Protocol) telephony. Users interacted with the embedded wizards by telephone, and wizards took over after CheckItOut answered the phone. After familiarization with the task and GUI, nine wizards auditioned and six were selected. There were ten users. Both groups were evenly balanced for gender. Users were directed to a website that presented scenarios for each call. The scenario page gave the user a patron identity and phone number, and author, title and catalogue number information for four books they were to order. Each user was to make at least fifteen calls to each wizard; we recorded 913 usable calls.

A single trainer prepared the original nine wizard volunteers one at a time. First, each trainee practiced on data from the experiments described above. Next, the trainer explained the wizard GUI and demonstrated it, serving as wizard on a sample call. Finally, the trainee served as wizard on five test calls with guidance from the trainer. The trainer chose the six most skilled and motivated trainees as wizards.

The GUI had two screens, one for user login and one for book requests. Users identified themselves by scenario phone number. The book request screen had a scrollable frame displaying the ASR for each user utterance. Separate frames on the GUI displayed the query return, dialogue history, basic actions (e.g., querying the backend with a custom R/O query, or prompting the user for a book), and auxiliary actions (e.g., removing a book from the order in progress). Finally, wizards could select among four types of dialogue acts: signals of non-understanding, or clarifications about the ASR, the book request or the query return. A dialogue act selected by the wizard was passed to a template-based natural language generator, and then to a Text-to-Speech component. Due to their complexity, calls could be time consuming. A clock on the GUI indicated call duration; wizards were instructed to finish the current book request and then terminate the call after six minutes.

A wizard's *precision* is the proportion of books she offer that correctly match the user's request; five of the six wizards had precision over 90%. A wizard's *recall* is the number of books in the scenario that she correctly identified. The two best wizards, WA and WB, had the highest recall, 63% and 67% respectively.

The number of book requests per dialogue was tallied automatically. Some dialogues were terminated before all scenario books could be requested. Also, a wizard who experienced problems with a book request could abandon the current request and prompt the user for a new book. The user could resume the abandoned book request later in the dialogue. In such cases, the abandoned and resumed requests for the same book would count as two distinct book requests. Given these facts, the ratio of number of correct books to number of book requests yields only an approximate estimate of how many scenario books were correctly identified. WA correctly identified 2.69 books per call from 3.64 requests per call, yielding a total success rate of 73.9% per book request, and 67.25% per 4-book scenario. WB correctly identified 2.54 books per call from 4.44 requests per call, yielding success rates of 57.21% per request and 63.50% per 4-book scenario. WA and WB had quite distinct strategies. WA persisted with each book request and exploited a wide range of the available GUI actions, with the greatest number of actions per book request among all wizards (N=8.24). WB abandoned book requests early and moved on to the next book request, exploited relatively fewer GUI actions, and had the fewest actions per book request (N=5.10).

From 163 features that characterize the ASR, search, current user utterance, current turn exchange, current book request, and the entire dialogue, we learned models for three types of wizard actions: select a non-understanding prompt, perform a search, or select a prompt to disambiguate among search returns. We used three machine learning methods for classification: decision trees, logistic regression and support vector machines. Table 2 gives the accuracies and overall F measures for decision trees that model WA and WB. (All learning methods have similar performance.)

Of note here is the range of features that predict when the best wizards selected a non-understanding, shown in Table 3. In addition, the two models depend partly on different features. Trees for the other actions in Table 2 have similarly diverse features.

Wizard	Action	Acc	F
A	Non-Understanding	0.71	0.71
B	Non-Understanding	0.73	0.73
A	Disambiguate	0.80	0.81
B	Disambiguate	0.86	0.87
A	Search	0.94	0.95
B	Search	0.93	0.94

Table 2: Performance of learned trees

To evaluate the benefit of learned models of wizard actions for SDS, we conducted two data collections where subjects placed calls following the same types of scenarios used in Full WOz. For our baseline evaluation of CheckItOut, 10 subjects were recruited from Columbia University and Hunter College. Each was to place a minimum of 50 calls over a period of three days; 562 calls were collected. For each call, subjects visited a web page that presented a new scenario. Each scenario included mock patron data for the caller to use (e.g., name, address and phone number), a list of four books, and instructions to request one book by catalogue number, one by title, one by author, and one by any of those methods. At three points during their calls, subjects completed a user satisfaction survey containing eleven questions adapted from (Hone and Graham, 2006).

CheckItOut+ is an enhanced version of our SDS in which the DM was modified to include learned models for three decisions. The first determines whether the system should signal non-understanding in response to the caller's last utterance, and executes before voice search would take place. The second determines whether to perform voice search with the ASR (i.e., before the parse, in contrast to CheckItOut). The third executes after voice search, and determines whether to offer the candidate with the highest R/O score to the user. The evaluation setup for CheckItOut+ also included 10 callers who were to place 50 calls each; 505 calls were collected.

Here we report results that compare the number of books ordered per call, the number of correct books per call, the elapsed time per book ordered, and elapsed time per correct book. T-tests show all differences to be highly significant. (A full discussion of the evaluation results will appear in future publications.) Callers to CheckItOut+ nearly always ordered four books (3.998), compared with 3.217 for the baseline ($p < 0.0001$). There was an increase of correct books in the order from 2.40 in the baseline to 2.70 in CheckItOut+ ($p < 0.0001$). The total elapsed time per call increased by only 13 seconds from 210.93 to 223.96 ($p < 0.0175$). Given that CheckItOut+ callers ordered more books and more correct books, CheckItOut+ performed much faster. The elapsed time per ordered book decreased from 65.57 to 56.01 seconds, and decreased from 87.89 to 82.95 seconds per correct books.

Feature	WA	WB
# books ordered so far	Y	Y
% unparsed ASR words	Y	N
Avg. word confidence	Y	N
# explicit confirms in call	Y	Y
# MICA slots per concept	Y	N
# searches in call	Y	N
Most recent wizard action	N	Y
Most frequent concept in call	N	Y
Speech rate	N	Y
# user utts. this request	N	Y
# author searches in call	Y	Y
Normalized LM score this utt	Y	Y

Table 3: Features that predict wizards' non-understanding

5 Discussion

Spoken language understanding has been relatively under-investigated in SDS design. Our experiments suggest that tighter integration of all phases of SLU with dialogue management can lead to more robust system behavior. We illustrate here with an example of WA's strategic questioning in which a non-understanding is avoided, and WA builds on partial understanding to identify the user's objective.

In response to ASR `MARY .JO. EARTH`, where the `'.'`s bracket an unconfident word, WA's search returned three authors with first name `Mary`, and last names that had moderate character overlap with `.JO. EARTH`. WA first asked whether the book was by `Mary O'Hara`. When the user responded negatively, WA confirmed the first name, prompted for the last name, and got `SURE` as the ASR. WA finally confirmed that the book was by *Mary Stewart*. Although it took four turns, WA was able to identify the correct book.

In general, the Full WOz corpus contains a very high proportion of wizard questions. In the total corpus of 20,415 caller utterances, there were 11,562 wizard questions. The types of questions

S1: What's the next book?
U1: .BARBARA. THREE THREE
S2: Is the author barbara freethy?
U2: YES
S3: Would you like 'some kind of wonderful' by BARBARA FREETHY?
U3: YES

(a) Example 1

S1: Sorry, I misunderstood. Can you repeat the author?
U1: DIDN'T I AM THE .GOLDEN. ARM
S2: Is the title 'THE man with THE golden ARM '?
U2: NO
S3: Sorry, I misunderstood. Can you repeat the title please?
U3: .A. .AMBLING. .THE. .GAME. .EDELMAN. STORY
S4: Is the title 'up and running the jami goldman STORY '?
U4: YES

(b) Example 2

Figure 2: Sample Clarification Subdialogues

wizard's ask not only often lead to successful concept identification, they also avoid prompting the user to repeat what they said. Previous work has presented results showing that the hyperarticulation associated with user repetitions often leads users to slow their speech, speak more loudly, and pronounce words more carefully, which hurts recognition performance (Hirschberg et al., 2004).

Figure 2 illustrates two clarification subdialogues from CheckItOut+. The first illustrates how prior knowledge about what a user might say provides sufficient constraints to interpret ASR that would otherwise be unintelligible. The first word in the ASR for the caller's first utterance is bracketed by '.', which again represents low word confidence. The high confidence words `THREE THREE` are phonologically and orthographically similar to the actual author name, *Freethy*. Note that from the caller's point of view, the same question shown in S3 could be motivated by confusion over the words alone, as in this case, or confusion over the words and multiple candidate referents (e.g., *Barbara Freethy* versus *Freeling*).

The second clarification subdialogue illustrates how confusions about the linguistic input can be resolved through strategies that combine questions about words and intents. The prompt at system turn 3 indicates that the system believes that the caller provided a title in user turn 1, which is incorrect. The caller responds with the title, however, which provides an alternative means to guess the intended book, Jami Goldman's memoir *Up and Running*.

6 Conclusion

The studies reported here are premised on two hypotheses about the role spoken language understanding plays in SDS design. First, prior knowledge derived from the context in which a dialogue takes place can yield predictions about the words a user might produce, and that these predictions can play a key role in interpreting noisy ASR. Here we have used context derived from knowledge in the application database. Similar results could follow from predictions from other sources, such as an explicit model of the *alignment of linguistic representations* proposed in the work of Pickering and Garrod (e.g., (Pickering and Garrod, 2006). Second, closer integration of spoken language understanding and dialogue management affords a wider range of clarification subdialogues.

Our results from the experiments reported here support both hypotheses. Our first experiment demonstrated that words obscured by very noisy ASR (50% $\leq$ WER $\leq$ 75%) can be inferred by reliance on what might have been said, predictions that came from the database of entities in the domain. We assume that an SDS that interacts well when ASR quality is poor will perform all the better when ASR quality is good. Our second experiment demonstrated that two of five human wizards were able to achieve high accuracy in on-line resolution of noisy ASR, when presented with no more than ten candidate matches. Run-time recognition features not available to the wizards were nonetheless useful in modeling the ability of the two best wizards to avoid false hits. Our third experiment demonstrated that wizards could achieve high task success on full dialogues where callers requested four books, and an enhancement of our baseline SDS with learned models of three wizard actions led to improved task success with less time per subtask. The variety of features that contribute to learned models of wizard actions demonstrates the advantages of embedded wizardry, as well as the benefit of DM clarification strategies that include features from all phases of SLU.

Acknowledgments

The Loqui project is funded by the National Science Foundation under awards IIS-0745369, IIS-0744904 and IIS-084966. We thank those at Carnegie Mellon University who helped us construct CheckItOut through tutorials and work sessions held at Columbia University and Carnegie Mellon University, and who responded to numerous emails about the Olympus/RavenClaw architecture and component modules: Alex Rudnicky, Brian Langner, David Huggins-Daines, and Antoine Raux. We also thank the many undergraduates from Columbia College, Barnard College, and Hunter College who assisted with tasks that supported the implementation of CheckItOut, including the telephony.

References

Gregory Aist, James Allen, Ellen Campana, Carlos Gomez Gallo, Scott Stoness, Mary Swift, and Michael K. Tanenhaus. 2007. Incremental dialogue system faster than and preferred to its nonincremental counterpart. In *COGSCI 2007*, pages 779–74.

John L. Austin. 1962. *How to Do Things with Words*. Oxford University Press, New York.

Srinivas Bangalore, Pierre B. Boullier, Alexis Nasr, Owen Rambow, and Benoîit Sagot. 2009. Mica: a probabilistic dependency parser based on tree insertion grammars. In *NAACL/HLT*, pages 185–188.

Dan Bohus and Alex Rudnicky. 2002. Integrating multiple knowledge sources for utterance-level confidence anno-tation in the CMU Communicator spoken dialogue system. Technical Report CS-02-190, Carnegie Mellon University, Department of Computer Science.

Dan Bohus and Alex Rudnicky. 2009. The RavenClaw dialog management framework. *Computer Speech and Language*, 23:332–361.

Dan Bohus. 2004. *Error awareness and recovery in conversational spoken language interfaces*. Ph.D. thesis, Carnegie Mellon University, Computer Science.

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

Kallirroi Georgila, Kyrakos Sgarbas, Anastasios Tsopanoglou, Nikos Fakotakis, and George 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*, 6:145–59.

Joshua Gordon and Rebecca J. Passonneau. 2011. An evaluation framework for natural language understanding in spoken dialogue systems. In *7th LREC*.

Joshua Gordon, Rebecca J. Passonneau, and Susan L. Epstein. 2011. Helping agents help their users despite imperfect speech recognition. In *Proceedings of the AAAI Spring Symposium 2011 (SS11): Help Me Help You: Bridging the Gaps in Human-Agent Collaboration*.

Julia Hirschberg, Diane Litman, and Marc Swerts. 2004. Prosodic and other cues to speech recognition failures. *Speech Communication*, 43(1-2):155–75.

Kate S. Hone and Robert Graham. 2006. Towards a tool for the subjective assessment of speech system interfaces (sassi). *Natural Language Engineering, Special ISsue on Best Practice in Spoken Dialogue Systems*, 6(3-4):287–303.

David Huggins-Daines, Mohit Kumar, Arthur Chan, Allen W. Black, Mosur Ravishankar, and Alex I. Rudnicky. 2006. PocketSphinx: A free, real-time continuous speech recognition system for hand-led devices. In *Proceedings of ICASSP*, volume I, pages 185–188.

Ivana Kruijff-Korbayová, Nate Blaylock, Ciprian Gerstenberger, Verena Rieser, Tilman Becker, Michael Kaisser, Peter Poller, and Jan Schehl. 2005. An experiment setup for collecting data for adaptive output planning in a multimodal dialogue system. In *10th ENLG*, pages 191–196.

Cheongjae Lee, Alexander Rudnicky, and Gary Geunbae Lee. 2010. Let's buy books: finding ebooks using voice search. In *IEEE-SLT 2010*, pages 442–447.

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

Tiziana Ligorio, Susan L. Epstein, and Rebecca J. Passonneau. 2010a. Wizards' dialogue strategies to handle noisy speech recognition. In *IEEE-SLT 2010*.

Tiziana Ligorio, Susan L. Epstein, Rebecca J. Passonneau, and Joshua Gordon. 2010b. What you did and didn't mean: Noise, context and human skill. In *COGSCI 10*.

Rebecca J. Passonneau, Susan L. Epstein, and Joshua Gordon. 2009. Help me understand you: Addressing the speech recognition bottleneck. In *Proceedings of the AAAI Spring Symposium 2009 (SS09): Agents that Learn from Human Teachers*, pages 23–25.

Rebecca J. Passonneau, Susan L. Epstein, Tiziana Ligorio, Joshua Gordon, and Pravin Bhutada. 2010. Learning about voice search for spoken dialogue systems. In *NAACL-HLT 2010*, pages 840–848.

Martin J. Pickering and Simon Garrod. 2006. Alignment as the basis for successful communication. *Research on Language and Communication*, 4(2):203–228.

Matthew Purver, Jonathan Ginzburg, and Patrick Healey. 2001. On the means for clarification in dialogue. In *Proceedings of the 2nd SIGdial Workshop on Discourse and Dialogue*, pages 116–125.

J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

John W. Ratcliff and David Metzener. 1988. Pattern matching: the gestalt approach.

Antoine Raux and Maxine Eskenazi. 2004. Non-native users in the Let's Go! spoken dialogue systems. In *HLT/NAACL*, pages 217–224.

Antoine Raux and Maxine A. Eskenazi. 2007. A multi-layer architecture for semi-synchronous event-driven dialogue management. In *ASRU 2007*, pages 514–519.

Antoine Raux, Brian Langner, Allan W. Black, and Maxine Eskenazi. 2005. Let's Go Public! taking a spoken dialogue system to the real world. In *Interspeech - Eurospeech 2005*, pages 885–888.

Verena Rieser and Oliver Lemon. 2006. Using machine learning to explore human multimodal clarification strategies. In *COLING/ACL*, pages 659–666.

Verena Rieser and Oliver Lemon. 2011. Learning and evaluation of dialogue strategies for new applications: Empirical methods for optimization from small data sets. *Computational Linguistics*, 37:153–96.

David Schlangen and Raquel Fern'andez. 2005. Speaking through a noisy channel – experiments on inducing clarification behaviour in human-human diaogue. In *8th Annual Converence of the International Speech Communication Association (INTERSPEECH 2007)*, pages 1266–1269.

Gabriel Skantze. 2003. Exploring human error handling strategies: Implications for spoken dialogue systems. In *Proceedings of ISCA Tutorial and Research Workshop on Error Handling in Spoken Dialogue Systems*, pages 71–76.

Gabriel Skantze. 2005. Exploring human recovery strategies: Implications for spoken dialogue systems. *Speech Communication*, 45:325–41.

Svetlana Stoyanchev and Amanda Stent. 2009. Predicting concept types in user corrections in dialog. In *EACL Workshop SRSL*, pages 42–49.

Ye-Yi Wang, Yu Dong, Yun-Cheng Ju, and Alex Acero. 2008. An introduction to voice search. *IEEE Signal Processing Magazine: Special ISsue on Spoken Language Technology*, 25(3):28–38.

Wayne Ward and Sunil Issar. 1994. Recent improvements in the CMU spoken language understanding system. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 213–216.

Jason D. Williams and Steve Young. 2004. Characterizing task-oriented dialog using a simulated ASR channel. In *ICSLP/Interspeech*, pages 185–188.

Teresa Zollo. 1999. A study of human dialogue strategies in the presence of speech recognition errors. In *Proceedings of the AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, pages 132–139.