A “*K* Hypotheses + Other” Belief Updating Model

Dan Bohus, Alex Rudnicky

Computer Science Department,
Carnegie Mellon University,
Pittsburgh, PA, 15217
dbohus@cs.cmu.edu, air@cs.cmu.edu

Abstract

Spoken dialog systems typically rely on recognition confidence scores to guard against potential misunderstandings. While confidence scores can provide an *initial* assessment for the reliability of the information obtained from the user, ideally systems should leverage information that is available in subsequent user responses to *update* and *improve the accuracy* of their beliefs. We present a machine-learning based solution for this problem. We use a compressed representation of beliefs that tracks up to *k* hypotheses for each concept at any given time. We train a generalized linear model to perform the updates. Experimental results show that the proposed approach significantly outperforms heuristic rules used for this task in current systems. Furthermore, a user study with a mixed-initiative spoken dialog system shows that the approach leads to significant gains in task success and in the efficiency of the interaction, across a wide range of recognition error-rates.

Introduction

Today’s spoken language interfaces are still very brittle when faced with understanding-errors. The problem is present in all domains and interaction types and stems mostly from the speech recognition process. The recognition difficulties are further exacerbated by the conditions under which spoken dialog systems typically operate: spontaneous speech, large vocabularies and user populations, and large variability in input line quality. Under these circumstances, average word-error-rates of 20-30% (and up to 50% for non-native speakers) are quite common.

To guard against potential misunderstandings, spoken dialog systems rely on recognition confidence scores. Typically, the confidence score of the current hypothesis is compared to a set of predetermined thresholds. Based on the result of this comparison, the system will decide whether to accept the input, reject it, or engage in a confirmation action, such as explicit or implicit confirmation (see Figure 1). After the user provides a response to the confirmation action, the system has to update its belief about the concept that was confirmed. Very simple *heuristic update rules* are generally used for this task. For instance most systems consider a hypothesis grounded if they hear a *yes* response (e.g. yes, that’s right, etc) to an explicit confirmation; alternatively, if the response contains a negative marker (e.g. no, incorrect, etc), the hypothesis will be deleted. For implicit confirmations, most systems rely on the user to overwrite the concept if the confirmed value is incorrect.

We believe these heuristic approaches are suboptimal for a number of reasons. As example 2 in Figure 1 illustrates, users do not always overwrite slots. Previous studies (Krahmer et al, 2001; Bohus and Rudnicky, 2005a) have shown that user responses following implicit confirmations cover a wide language spectrum, and simple heuristic rules fall short on that account. Recognition errors can further complicate the problem (see example 2 in Figure 1). Last but not least, heuristic rules generally construct polarized beliefs (e.g. trust the value / don’t trust the value) that do not capture the degree of uncertainty the system should have, and therefore do not provide a robust basis for intelligent decision making.

While confidence scores can provide an *initial* assessment of reliability, we believe that spoken dialog systems should better leverage information from subsequent user turns and continuously *update* and *improve the accuracy* of their beliefs. Because corrections can appear at any point in the conversation, spoken dialog systems should update their beliefs not only after confirmation actions, but after each user turn, throughout the whole interaction.

In previous work (Bohus and Rudnicky, 2005a) we introduced and addressed a restricted version of this belief updating problem. We developed learning-based models for updating the confidence of the current top hypothesis

Example 1: Explicit confirmation

1 S: Starting at what time do you need the room?
2 U: `[STARTING AT TEN A_M / 0.45]`
Starting at ten a.m.
3 S: **Did you say you wanted the room starting at ten a.m.?**
4 U: `[GUEST UNTIL ONE / 0.89]`
Yes until noon

Example 2: Implicit confirmation

1 S: How may I help you?
2 U: `[THREE TO RESERVE A ROOM / 0.65]`
I’d like to reserve a room
3 S: **starting at three p.m.** … For which day do you need the conference room?
4 U: `[CAN YOU DETAILS TIME] - NONU/0.00`
I need a different time

Figure 1. Sample explicit and implicit confirmations. System turns are prefixed by (S); user turns are prefixed by (U); the speech recognition results and associated confidence scores are represented `[in between brackets]`

for a concept, in light of the user responses to system confirmation actions. The models significantly outperformed the heuristic rules used in current spoken dialog systems.

In this paper we generalize the previous models in several ways. First, we extend the models to use a more comprehensive, but still compact representation of beliefs: *k hypotheses + other*, where ***k*** is a small integer. Second, we extend the models to perform updates after all system actions (as opposed to only after confirmation actions). Finally, we include information in the models about the prior likelihood, as well as the confusability of various concept values.

Experimental results show that extended belief updating models significantly outperform heuristic update rules. Furthermore, the proposed models induce significant gains on several dialog performance metrics. Experiments with a mixed-initiative spoken dialog system show that the proposed models lead to significant gains in task success and to a significant reduction in the number of turns-to-completion for successful tasks.

We start by formalizing the problem.

The Belief Updating Problem

The *belief updating problem* introduced in the previous section can be formalized as follows.

• Let ***C*** denote a concept (or slot) that the system acquires from the user (e.g. date, start_time, end_time. etc);

• Let $B_t(C)$ denote the system's belief over the concept ***C*** at time ***t***, i.e. a representation of the system's uncertainty in the value of the concept ***C*** (e.g. start_time = {10 p.m. / 0.8 ; 2 p.m. / 0.2});

• Let ***SA(C)*** denote the system action at time ***t***, with respect to concept ***C***. Note that in a single turn the system might take different actions with respect to different concepts (e.g. in example 2 from Figure 1 the system engages in an implicit confirmation w.r.t. the date concept and a request w.r.t. the time concept);

• Finally, let ***R*** denote the user response to the system action, as this response is perceived by the system.

Then, the *belief updating problem* is stated as follows:

> given an initial belief over a concept $B_t(C)$, a system action ***SA(C)***, and a user response ***R***, construct an updated belief $B_{t+1}(C)$

$$B_{t+1}(C) \leftarrow f\,(\,B_t(C), SA(C), R\,)$$

Approach

We propose a data-driven approach for addressing this problem. We treat each concept independently, we use a compressed belief representation (described in detail below), and we induce the function ***f*** from training data using simple machine learning techniques.

In previous work (Bohus and Rudnicky, 2005a) we have addressed a simplified version of the belief updating problem. A first simplification concerned the belief representation. Ideally, the system's belief over a concept would be represented by a probability distribution over the full set of possible values for that concept. However, from a practical perspective, it is very improbable that a spoken dialog system would "hear" more than 3 or 4 conflicting values for a concept throughout a conversation (this observation is supported by empirical evidence – see Bohus and Rudnicky, 2005a). Under these circumstances, a compressed belief representation has the potential of greatly simplifying the problem without causing a significant degradation in performance. We therefore developed a belief updating approach where the system's belief over a concept was represented simply by the confidence score of the current top hypothesis for that concept (all other hypotheses were ignored; the remaining probability mass was accumulated in an *other* category). Secondly, we only focused our attention on updating beliefs after system confirmation actions. The restricted version of the belief updating problem was therefore stated as follows:

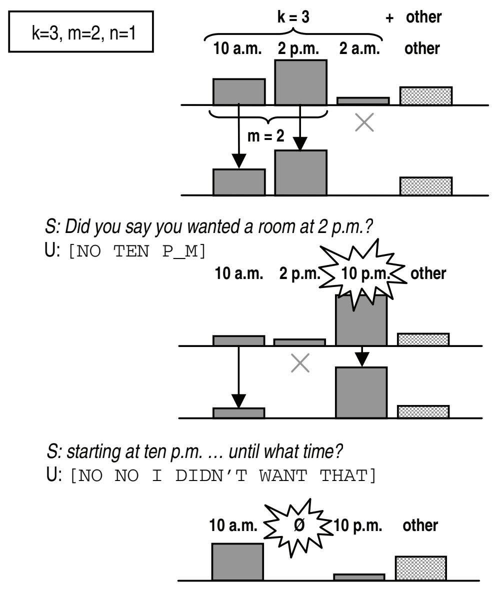
> given an initial confidence score for the top hypothesis ***h*** for a concept ***C***, construct an updated confidence score for ***h***, in light of the system confirmation action ***SA(C)***, and the follow-up user response ***R***.

We addressed this problem using a simple machine learning approach. As training data, we used a manually-labeled corpus of 449 dialogs. We identified a large number of potentially relevant features from different knowledge sources in the system. For each system confirmation action we trained a model tree with stepwise logistic regression models at the leaves. The models were evaluated using a 10-fold cross-validation process. The results, described in more detail in (Bohus and Rudnicky, 2005a) show that the learned models significantly outperform the heuristic update rules previous used in our system.

Encouraged by the positive results on the simpler version of the problem, we decided to remove the restrictions and generalize the models in several important ways.

1st extension: "*k* hypotheses + other" belief representation. First, we use a more comprehensive, but still compressed representation of beliefs – *k hypotheses + other*. In this representation, the system tracks up to ***k*** potential values for each concept (***k*** is a small fixed integer). During each update, the system retains only the ***m*** highest scoring hypotheses from the initial set of ***k*** hypotheses, and acquires up to ***n*** new hypotheses from the user response (***m***, ***n*** are small fixed integers, with ***m+n=k***). At each update, the system therefore constructs a new belief over the space of ***m*** initial hypotheses, ***n*** new ones and *other*. The approach allows for new hypotheses to be dynamically integrated into the compressed belief as they appear in subsequent user responses.

We illustrate this process in Figure 2. Two consecutive updates of the start_time concept are shown. Assume ***k***=3, ***m***=2, ***n***=1. Also assume that at time t the system already has 3 conflicting hypotheses for the start_time concept: 10 a.m., 2 p.m. and 2 a.m. The bars in the figure reflect the system's confidence in each of these values. The remaining



S: Did you say you wanted a room at 2 p.m.?
U: `[NO TEN P_M]`

S: starting at ten p.m. ... until what time?
U: `[NO NO I DIDN'T WANT THAT]`

Figure 2. Two consecutive belief updates of start_time

probability mass is assigned to the *other* category. The system engages in an explicit confirmation of the top hypothesis (2 p.m.) and updates the concept in light of the user response. During the update the system retains the top two (***m***=2) initial hypotheses (in this case 10 a.m. and 2 p.m.), and acquires one (***n***=1) new hypothesis from the user input – 10 p.m. The system therefore constructs a new belief over the space of ***m***=2 initial hypotheses, ***n***=1 new hypothesis, and *other*: {10 a.m., 2 p.m, 10 p.m., *other*}.

Next, the system engages in an implicit confirmation on the new top scoring hypothesis (10 p.m.). During this second update the system again retains the top 2 scoring hypotheses (this time 10 a.m. and 10 p.m.). Since no new hypothesis is present in the user response, we introduce a fictitious Ø hypothesis. The system will therefore construct an updated belief over the {10 a.m., 10 p.m., Ø, *other*} space. The probability mass assigned to the Ø hypothesis by the belief updating process will be automatically moved to the *other* category, as a post-processing step.

The system's belief in a concept is therefore modeled as a multinomial variable of degree ***k***+1: $\boldsymbol{B=<pi_1, \ldots, pi_m, pn_1, \ldots, pn_m, pother>}$, where $\boldsymbol{pi_a}$ is the probability for the initial hypothesis $\boldsymbol{a}$, $\boldsymbol{pn_b}$ is the probability for the new hypothesis $\boldsymbol{b}$, and $\boldsymbol{pi_1 + \ldots + pi_m + pn_1 + \ldots + pn_n + pother} = 1$. The representation abstracts over the actual values (e.g. 10 a.m., 10 p.m., etc), and allows for hypotheses to be dynamically added and dropped from the compressed belief representation. The actual values are stored separately, and they do not directly enter the belief (confidence) updating process. Using this representation, the belief updating problem can be cast a multinomial regression problem:

$$\boldsymbol{B_{t+1} \leftarrow B_t + SA(C) + R}$$

Note that the *top hypothesis* + *other* representation used in our previous work is equivalent to setting ***k*** to 1. In that case the target variable was binomial, and a logistic regression model sufficed.

2nd extension: updates after all system actions. In our previous work, we focused on updates after system confirmation actions. However, user corrections (either actual or false ones introduced by noisy recognition) can appear at any point in the dialog. We therefore extended the models to perform updates after all system actions.

We identified five types of actions that can be performed on any given concept ***C*** (note that multiple such actions can be coupled in a single system turn):

- **explicit confirmation**: the system asks a question to explicitly confirm a hypothesis for the concept ***C***;
- **implicit confirmation**: the system implicitly confirms a hypothesis for ***C*** and moves on to the next question;
- **unplanned implicit confirmation**: the system prompt includes the value for the concept ***C***. For instance, the system might respond "I found 10 rooms **this Friday** between **2** and **4 p.m.** Would you like a small room or a large one?" While the main purpose of this turn is to provide information to the user, the turn also includes the values for the date, start_time and end_time concepts, and therefore constitutes an implicit confirmation. However, this is not a planned error handling action; rather, it occurs as a side-effect of the prompt design;
- **request**: the system requests the concept ***C***;
- **other**: the system does not perform any action with respect to concept ***C***, but nevertheless a value is heard for this concept and an update is needed.

3rd extension: new features. Finally, we expanded the feature set to include information about the prior likelihood as well as the confusability of various concept values.

Experiments

Training Data

The data used for training the belief updating models was collected via a user study with the RoomLine system (RoomLine, 2003). RoomLine is a telephone-based, mixed-initiative, task-oriented spoken dialog system that can assist users in making conference room reservations. The system has information about the schedules and characteristics of 13 conference rooms in two buildings on campus and can engage in a negotiation dialog to identify the room that best matches the user's needs. During the data collection experiment 46 participants engaged in a maximum of 10 scenario-based conversations with the system. Sample interactions, as well as the 10 scenarios are available in (Bohus, 2006).

The collected corpus contains 449 dialogs, 8278 user turns and 11332 concept updates. The user utterances were orthographically transcribed and checked by a second annotator. Based on the transcriptions, we labeled the correct (user specified) concept values at each turn in the dialog.

Features

We identified a large number of potentially relevant features from different knowledge sources in the system. Given space constraints, we briefly summarize the feature set below. A more detailed description is available in (Bohus, 2006).

- **Initial belief features**: confidence scores for the ***m*** initial hypotheses, the identity of the updated concept;
- **Acoustic and prosodic features of the user response:** duration, pitch, speech rate, initial pause, etc.;
- **Lexical features of the user response:** number of words, the presence/absence of lexical terms highly correlated with user corrections;
- **Grammatical features of the user response:** number of grammar slots contained in the parse, the number of new and repeated slots, goodness-of-parse scores, etc.;
- **Dialog level features of the user response:** how well the response matches the current system expectation, the turn number, the current dialog state, etc.;
- **Priors:** features capturing the prior likelihood of various concept hypotheses. The RoomLine system operates with a set of 29 concepts. A domain expert (who did not have access to the corpus) manually constructed priors for 3 of the most important concepts in the system: the start_time, the end_time and the date for the reservation. For all other concepts, we assumed uniform priors;
- **Confusability scores:** features describing the confusability between various concept hypotheses. These confusability scores were determined empirically from the collected data.

Results

We trained and evaluated models in three different setups: <***k***=2,***m***=1,***n***=1>, <***k***=3,***m***=2,***n***=1> and <***k***=4,***m***=3,***n***=1>. We report results from the simpler model <***k***=2,***m***=1,***n***=1>. The larger models generated very similar results, and no significant performance improvements were observed.

For each system action, we trained a separate multinomial generalized linear (regression) model (Fahrmeir and Tutz, 2001). We trained the models in a stepwise approach: the next most informative feature (as measured by the improvement in the average log likelihood over the training set) was added to the model at each step. We used the Bayesian Information Criterion to prevent over-fitting.

We evaluated the models using a 10-fold cross-validation procedure. Figure 3 shows the error-rates for 3 different models. Given the multinomial nature of the target variable (k=2 + other), the error-rate is computed as for a 3-class classification problem; results using different metrics such as average (log)-likelihood of the correct hypothesis are omitted here, but are available in (Bohus, 2006). The *basic* model (BM) uses all features except for the priors and confusability scores. In contrast, the *full* model (FM) uses the full set of features. Finally, the *runtime* model (RM) uses only the features available at runtime, i.e. all features except the prosodic ones.

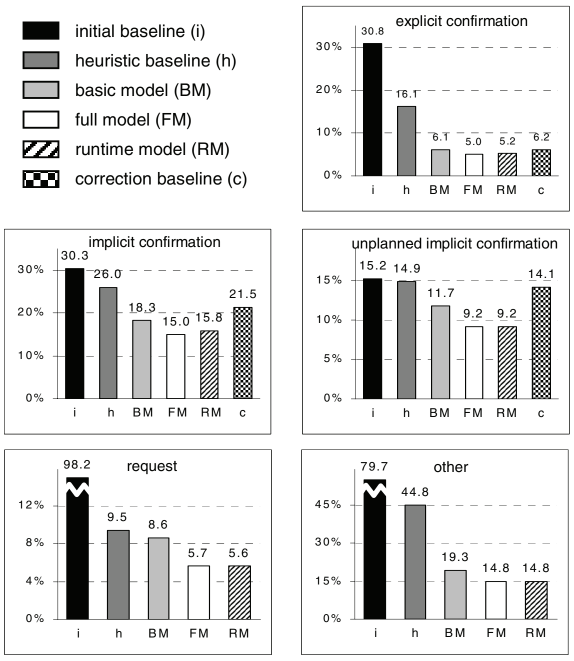


Figure 3. Error-rates for belief updating models

We compared each model's performance against 3 baselines: *initial*, *heuristic* and *correction*. The *initial* baseline reflects the error in the initial system belief, before the update is performed. The *heuristic* baseline reflects the error rate for the heuristic update rule currently used by the system. Finally, we also report a *correction* baseline. A corpus analysis (Bohus and Rudnicky, 2005a) has revealed that users do not always correct system errors. As a result, even if we knew precisely when the user is correcting the system, we would not be able to construct perfectly accurate beliefs in one time-step. The *correction* baseline assumes that the hypothesis undergoing the confirmation is correct, unless the user attempts to correct the system (note that this baseline is defined only for system confirmation actions).

As Figure 3 illustrates, the data-driven models significantly outperform the heuristic rules previously used in the system (h). For updates after system confirmation actions the performance surpasses even the *correction* baseline (c); this is possible because the models use information that is not available to the *correction* baseline, such as the prosody of the user response, barge-in, as well as priors and confusability for the various concept hypotheses.

For all actions, the *full* models (FM), which include the priors and confusability information, perform better than the *basic* models (BM). No significant degradation of performance is observed for the *runtime* models (RM). Another interesting observation is that the *basic* model (BM) does not produce statistically significant improvements over the heuristic baseline for the *request* action. For this action, the heuristic rule simply constructs a belief based on the confidence score of the recognized hypothesis. The fact that our *basic* belief updating model is not able to perform better implies that the confidence annotation model is well calibrated in our system (no further improvements are feasible). However, adding confusability and prior information (the *full* model – FM), leads to a significant decrease in error (from 8.6% to 5.7%). We believe these results highlight the importance of using high-level, domain-specific information such as priors and confusability.

In order to gain a better understanding of the relative contributions and predictive power our features, we inspected the set of selected features and their corresponding weights in each regression model. The following features were generally found to be very informative: priors and confusability features, confidence of the initial top hypothesis before the update, concept identity, barge-in, dialog expectation match, and the presence of repeated grammar slots. More details about the constructed models are available in (Bohus, 2006).

Impact on Global Dialog Performance

So far, we have shown that the proposed models significantly outperform the heuristic rules typically encountered in current spoken dialog systems. While these results are encouraging, our ultimate goal is to improve global dialog performance in a real, practical spoken dialog system. An important question therefore remains: do improvements on the local (one-step) belief updating task translate into improvements in global dialog performance?

To address this question, we implemented the *runtime* models in the RavenClaw dialog management framework (Bohus and Rudnicky, 2003) and conducted a new user study to assess their impact on dialog performance.

During this experiment 40 participants interacted with the RoomLine system (each performed up to 10 scenario-driven interactions during a time period of 40 minutes). The participants were randomly assigned into 2 groups: the *control* group interacted with a version of the RoomLine system that used heuristic update rules: after each *yes-type* answer to an explicit confirmation, the system would boost the top hypothesis to 0.95; after each *no-type* answer, the system erased the hypothesis. Whenever a new value for a concept appeared, the system performed a naïve probabilistic update (e.g. multiplying the distributions and renormalizing). Participants in the *treatment* group interacted with a version of the system that used the *runtime* belief updating models. In all other respects, the systems were identical.

Our previous data analysis indicated that improvements in belief updating performance are likely to translate in global performance improvements especially when the word-error-rate (WER) is high. At low WER, simply trusting the inputs leads to generally accurate beliefs and not many opportunities for improvement exist; however, as recognition performance degrades, it becomes more and more important for the system to accurately assess its beliefs. In light of this observation, we decided to include only non-native speakers of north-American English in this user study. Nevertheless, the average per user WER in this population ranged from 8.4% to 49.8%.

The corpus collected in this experiment contains 384 dialogs and 6389 user turns (the results we report here were computed after we excluded one participant who misunderstood 7 of the 10 scenarios; keeping this participant in the corpus leads to a stronger, but we believe less accurate result). The task success rate was higher in the treatment condition: 81.3% vs. 73.6%. To better understand the effect of the proposed belief updating models on task success (TS), we performed a logistic analysis of variance using the experimental condition as a fixed effect and WER as a covariate. The resulting model is:

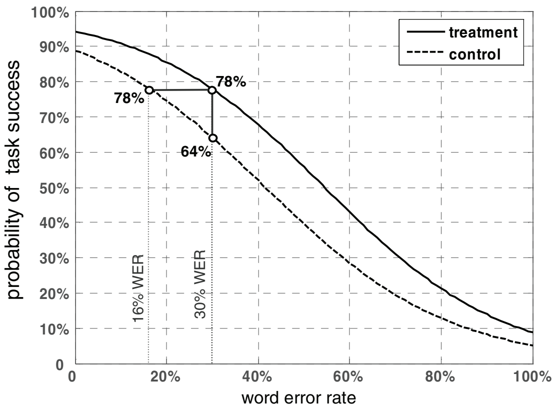


Figure 4. Improvement in task success at different word-error rates

$$\text{Logit(TS)} \leftarrow 2.09 - 0.05\cdot\text{WER} + 0.69\cdot\text{Condition}$$

This model shows that both the WER ($p<10^{-4}$) and the experimental condition (p=0.0095) have a significant impact on task success. Based on this fitted model, we plot the probability of task success as a function of WER in the two experimental conditions in Figure 4. As this figure illustrates, the proposed models lead to gains in task success across a wide range of WER. As expected, the improvements are larger at high WER. For instance, at a WER of 30%, the belief updating models produce a 14% absolute improvement in task success (from 64% to 78%). To attain the same task success with the *control* system, we would need a 16% WER. In effect, we have cut the WER in half. The largest improvement in task success, 16.3%, is obtained at a WER of 47%. At very high WER, the improvements decrease again, as it becomes much harder to improve global performance under these conditions. The analysis of variance shows than on average the proposed models improve the log-odds of task success by the same amount as a 13.6% absolute reduction in WER.

The proposed belief updating models also exert a significant impact on dialog efficiency. An analysis of variance using the experimental condition as a fixed effect, WER as a random effect and task duration for successful tasks (in number of turns) as the main effect shows that:

$$\text{Duration} \leftarrow -0.21 + 0.013\cdot\text{WER} - 0.106\cdot\text{Condition}$$

Both WER ($p<10^{-4}$) and the experimental condition (p=0.0003) have a significant effect on task duration. The improvements due to the belief updating models are equivalent to a 7.9% absolute reduction in WER.

Discussion and Future Work

To date, various machine learning methods have been proposed for detecting turn-level misunderstandings (Walker et al, 2000) and corrections (Hirschberg et al, 2001) in spoken dialog systems. Our work bridges these ideas and provides a unified approach that allows systems to continuously monitor and update their beliefs throughout the conversation.

Independent of the gains in effectiveness and efficiency, the proposed method has several other desirable properties. First, the proposed approach *learns from data*, *tracks multiple hypotheses*, and *integrates information across multiple turns* in the conversation. The idea of updating beliefs through time appears in previous work. Higashinaka et al. (2003) keep track of multiple dialog-states and resolve the ambiguities using empirical probabilities of dialog-act / dialog-act and dialog-act / dialog-state sequences. Before that, Paek and Horvitz (2000) used a handcrafted Dynamic Bayesian Network to continuously update the belief over a user's goal in a command-and-control application. We take inspiration from their work and we automatically induce the models from data. The advantage of the proposed machine learning technique (generalized linear models) is that it allows us to consider a very large number of features from different knowledge sources in the system, and it does not require expert knowledge about the potential relationships between these features. The most informative features are automatically selected and weighted accordingly.

Second, the proposed approach is *sample efficient* and *scalable*. The approach focuses on a local (one-turn) rather than a global optimization. Furthermore, the beliefs over each concept are updated independently (we are not modeling any inter-concept dependencies at this point). Although we sacrifice theoretical global optimality, learning is feasible even with relatively little training data. We have shown this approach can lead to significant gains in task success and dialog efficiency in a real-world, fairly complex spoken dialog system. RoomLine operates over a space of 29 concepts with cardinalities ranging between two to several hundred possible values. Our study indicates that good local decisions can sum up to improvements in overall performance. In the future, we plan to integrate a data-driven decision making component with this belief updating framework (Bohus and Rudnicky, 2005b). We believe this approach will help overcome the scalability issues that have hindered the deployment of learning-based optimization of dialog management (Singh et al, 2000) in real-world spoken dialog systems (Paek, 2006).

Last but not least, the approach is *portable*. The proposed belief updating framework was implemented as part of a generic dialog management engine, decoupled from any particular dialog task (we are planning to reuse it in other RavenClaw-based dialog systems). More importantly, the approach does not make any strong assumptions about the dialog management approach used (e.g. form-filling, plan-based, task-oriented, information state update, etc), and consequently does not tie a developer into any particular type of dialog controller.

We believe further performance improvements are possible. We plan to train and evaluate models that incorporate more information from the n-best list ($\boldsymbol{n} > 1$). Furthermore, we plan to add other high-level pragmatic features (e.g. features that capture domain-specific constraint violations). Another question we intend to address is: what is the trade-off between training set size and performance gains? Finally, we plan to investigate the transferability of these models across different domains.

Acknowledgements

This work grew out of initial discussions between the first author and Tim Paek and Eric Horvitz at Microsoft Research. We would like to thank them for their very useful feedback and contributions at various stages in this work. In particular, the idea of abstracting over the distribution of slot values was initially proposed by Tim Paek. We would also like to thank Antoine Raux and the other members of the Dialogs on Dialogs reading group at CMU for useful discussions and feedback.

References

Bohus, D., and Rudnicky, A. 2005a. *Constructing Accurate Beliefs in Spoken Dialog Systems.* in Proceedings of ASRU-2005, San Juan, Puerto Rico.

Bohus, D., and Rudnicky, A. 2005b. *Error Handling in the RavenClaw Dialog Management Architecture,* in Proceedings of HLT-EMNLP 2005, Vancouver, Canada.

Bohus, D., 2006. *www.cs.cmu.edu/~dbohus/blf_upd/*

Fahrmeir, L., and Tutz, G., 2001 *Multivariate Statistical Modeling Based on Generalized Linear Models*, in Springer Series in Statistics, ISBN 0387951873

Hirschberg, J., Litman, D., Swerts, M., 2001 *Identifying User Corrections Automatically in Spoken Dialogue Systems*, in Proceedings of NAACL'01, Pittsburgh, PA, June 2001

Krahmer, E., Swerts, M., Theune, M. and Weegels, M. 2001, *Error Detection in Spoken Human Machine Interaction*, in International Journal of Speech Technology, Vol.4, No.1, 19-29.

Paek, T., and Horvitz, E., 2000, *DeepListener: Harnessing expected utility to guide clarification dialog in spoken language systems*, in Proceedings of ICSLP'2000, Beijing, China, 2000.

Paek, T., 2006, *Reinforcement Learning for Spoken Dialogue Systems: Comparing Strengths and Weaknesses for Practical Deployment*, MSR Technical Report MSR-TR-2006-62.

RoomLine, 2003 – *http://www.cs.cmu.edu/~dbohus/RoomLine*

Higashinaka, R., Nakano, M., and Aikawa, K., 2000 – *OCorpus-based discourse understanding in spoken dialogue systems*, in Proceedings of ACL-2003, Sapporo, Japan.

Singh, S., Litman, D., Kearns, M., Walker, M., 2000 – *Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System*, in Journal of Artificial Intelligence Research, vol. 16, pp 105-133, 2000.

Walker, M., Wright, J., Langkilde, I., 2000 – *Using Natural Language Processing and Discourse Features to Identify Understanding Errors in a Spoken Dialogue System*, in ICML-2000.