Efficient Probabilistic Tracking of User Goal and Dialog History for Spoken Dialog Systems

Antoine Raux[1], *Yi Ma*[2]

[1]Honda Research Institute USA, Mountain View, CA, USA.
[2]Ohio State University, Columbus, OH, USA.
araux@honda-ri.com, may@cse.ohio-state.edu

Abstract

In this paper, we describe Dynamic Probabilistic Ontology Trees, a new probabilistic model to track dialog state in a dialog system. Our model captures both the user goal and the history of user dialog acts using a unified Bayesian Network. We perform efficient inference using a form of blocked Gibbs sampling designed to exploit the structure of the model. Evaluation on a corpus of dialogs from the CMU Let's Go system shows that our approach significantly outperforms a deterministic baseline, exploiting long N-best lists without loss of accuracy.
Index Terms: dialog systems, belief tracking, gibbs sampling

1. Introduction

One of the main functions of task-oriented spoken dialog systems is understanding the user intention. For instance, a bus schedule information system might need to understand where the user is leaving from and where they want to go in order to provide the time of the next bus. To achieve this, each user utterance is first recognized by an automatic speech recognizer (ASR) and parsed by a natural language understanding module (NLU). Subsequently, the system must combine the semantic output of the NLU across several utterances to infer the global user intention. We refer to this latter task as *belief tracking*. While traditional dialog systems rely on structures such as frames and heuristic rules to track the user goal, recently, probabilistic alternatives have been proposed [1, 2, 3, 4], with the expected benefit of enhanced flexibility and robustness in the face of ASR and NLU errors. These methods maintain probability distributions over the user goal, but offer little information on other aspects of dialog state such as the history of dialog acts.

In this paper, we introduce Dynamic Probabilistic Ontology Trees (D-POT), which extend our previous work on Probabilistic Ontology Trees [4] by capturing dialog history within the Bayesian Network. D-POT thus allows more accurate belief estimation as well as provides additional information that can be useful to dialog managers.

2. Belief Tracking for Dialog Systems

2.1. Problem Statement

Probabilistic belief tracking in spoken dialog systems (also referred to as belief updating and dialog state update) refers to the process of maintaining a probability distribution over possible dialog states based on noisy observations of user actions. The dialog state immediately after turn T, s_T contains information such as user goal, which we assume constant for the duration of the dialog, and dialog history (i.e. user and system actions from the beginning of the dialog up to now). In addition, the goal is often decomposed into a set of concepts. The problem is thus to estimate $P(s_T|(o_t)_{t=1}^T)$, the distribution over possible dialog states s_T given all previous observations $(o_t)_{t=1}^T$. o_t is typically obtained from the ASR and NLU modules and is comprised of an N-best list of concept values including information such as confidence scores.

2.2. Prior Work

While the general probabilistic framework described above is well established, recent work focused on scalability and tractability issues. For example, [5] and [6] both rely on partitioning the belief space according to concept values observed in the ASR input. These approaches give the same probability to all goals within a given partition, which makes them highly tractable. However, this simplification prevents the system from using prior knowledge about dependencies among concepts to infer possible values of *unseen* concepts. For instance, knowing the origin and destination of a bus trip can allow a system to infer the route and confirm it (rather than having to request it).

[3] relies on Loopy Belief Propagation, a general inference method for Bayesian Networks. In order for the belief update to work in real time, they make strong assumptions about concept dependencies. The only relationship that can hold between concepts is that of applicability. This limits the ability of the system to exploit prior knowledge.

[2] proposes to use particle filters, a sampling-based inference method, to track user intention in a technical assistance dialog system. We follow a similar path, using another well known sampling-based approach: Gibbs sampling. However, in [2], the design of the Bayesian Network is very specific to the troubleshooting task, whereas we present a model that is applicable to any form-based system.

3. Dynamic Probabilistic Ontology Trees

3.1. General Structure

A Dynamic Probabilistic Ontology Tree (D-POT) is a Bayesian Network composed of two parts:

1. a static sub-network that is tree-shaped and represents the user goal in the dialog (the *Goal Network*)
2. a set of sub-networks that capture the evidence provided by each utterance as recognized by the ASR/NLU (the *Evidence Networks*)

3.2. The Goal Network

We assume that the user goal is constant throughout the dialog and representable as a set of concept/value pairs. The

concepts capture atomic pieces of information relevant to the application. Each concept can take any of a discrete set of values. For example, in a bus information system, the goal might be decomposed in three concepts: `Bus` (the bus number), `Orig` (the origin stop), and `Dest` (the destination stop), and one instance of user goal might be `Bus=61C, Orig=CMU, Dest=OAKLAND`, capturing a specific trip the user wishes to get information about.

The role of the Goal Network is to capture the user goal for the dialog. Each concept is represented by a node C^i and related concepts are connected by an edge. We assume that the Goal Network has a tree structure, whose root we denote C^0. We do not make any assumptions regarding the semantics of the Goal Network's edges. A natural class of Goal Networks are Probabilistic Ontology Trees [4], whose edges represent *is-a* and *has-a* relationships. Figure 2 a) represents an example Goal Network for a bus schedule information domain with the three concepts described above. At any point in the dialog, the joint distribution over the Goal Network represents the system's belief about the user goal.

3.3. The Evidence Networks

The role of the Evidence Networks is to model user utterances and how they relate to the user goal. A new Evidence Network is attached to the D-POT at each turn t, consisting of a single dialog act node α_t and one observation node C_t^i for each goal concept C^i (e.g. Bus_t). α_t represents alternative dialog acts (or dialog act sequences) for this user utterance. Examples of individual dialog acts are `Provide:Bus`, when the user provides some value for concept `Bus`, and `AnswerConfirm:Dest=CMU` when the user responds (positively or negatively) to a request to confirm their destination as `CMU`. For ASR hypotheses that combine several dialog acts (e.g. "No, Downtown" would correspond to `AnswerConfirm:Dest=CMU, Provide:Dest`), the corresponding α_t value represents the complete dialog act sequence. For each turn, the domain of α_t is set dynamically based on the dialog acts occurring in the ASR N-best list. A key aspect of the model is that the values of α_t are *independent of the user goal* (i.e. they do not contain any information about the true value of the concepts). Thus, all α_t nodes are marginally independent of the Goal Network. This independence assumption allows inference over the network to be tractable (see Section 3.4). The conditional probabilities $P(C_t^i|C^i, \alpha_t)$ capture the evidence that dialog act α_t contributes to the system's knowledge of each concept and are estimated from the ASR/NLU output, including confidence scores.

Through the Evidence Networks, D-POT tracks the complete dialog act history. The distribution over α nodes can thus be used to estimate the probability of events such as "the user has confirmed concept `Bus` at some point in the dialog", or "the user has provided concept `Dest` three times or more so far", which are essential cues for the dialog manager to generate the next system question or confirmation.

3.4. Approximate Inference using Blocked Gibbs Sampling

Compared to a POT [4], the structure of the D-POT has two consequences. First, it contains undirected cycles due to the Evidence Networks, which means that simple exact inference techniques such as those described in [4] are no longer applicable. Second, because a new Evidence Network is added at each dialog turn, the D-POT keeps growing in size rendering exact inference quickly intractable. To compensate for these issues, we use blocked Gibbs sampling [7], which extends traditional Gibbs sampling by sampling from whole sub-networks (blocks) rather than individual nodes.

We treat the Goal Network and each Evidence Network as distinct blocks. In order to sample from the whole Goal Network given the observations and α_t nodes, we first use a recursive algorithm to compute at time T the posterior of each concept C^i, $P(C^i|Parent(C^i), (\alpha_t)_{t=1}^T, (C_t^i)_{t=1}^T)$. The algorithm, whose details we omit here for lack of space, is similar in nature to Algorithm 2 of [4]. Given this probability table, we sample from the Goal Network starting from the root node and subsequently using forward sampling. We then sample from each α_t node using the following equation:

$$P(\alpha_t|(C_t^i)_{i=0}^N, (C^i)_{i=0}^N) \quad = \quad K \cdot P(\alpha_t) \cdot \prod_{i=0}^{N} P(C_t^i|\alpha_t, C^i)$$

where $P(\alpha_t)$ and $P(C_t^i|\alpha_t, C^i)$ are read from the CPTs of the network and K is a normalizing constant.

3.5. Example

Figure 1 provides an example of the system's behavior for a two-turn interaction in the bus domain introduced in section 3.2. In the first turn, the user specifies the bus route (`61C`) and origin (`CMU`). However, the ASR/NLU's first hypothesis is incorrect (`[Orig=MURRAY,Dest=NEGLEY]`) while the second one is correct. Based on this input, the system constructs the α_1 node with two values: `[Provide:Orig,Provide:Dest]` and `[Provide:Bus,Provide:Orig]`. At this point, the top belief of the system corresponds to the incorrect top ASR/NLU hypothesis. In the next turn, the user specifies their destination (`OAKLAND`). This time the top ASR/NLU hypothesis is correct. A new α_2 node is generated with 2 values : `[Provide:Dest]` and `[Provide:Bus]`. Given the evidence from the two turns, the system updates its belief incorporating evidence from the second hypothesis of turn 1 (i.e. $\alpha_1 =$ `[Provide:Bus,Provide:Orig]`) and the first hypothesis of turn 2 ($\alpha_2 =$ `[Provide:Dest]`), which make up the most probable explanation of the evidence given the network.

4. Evaluation

4.1. Experimental Setup

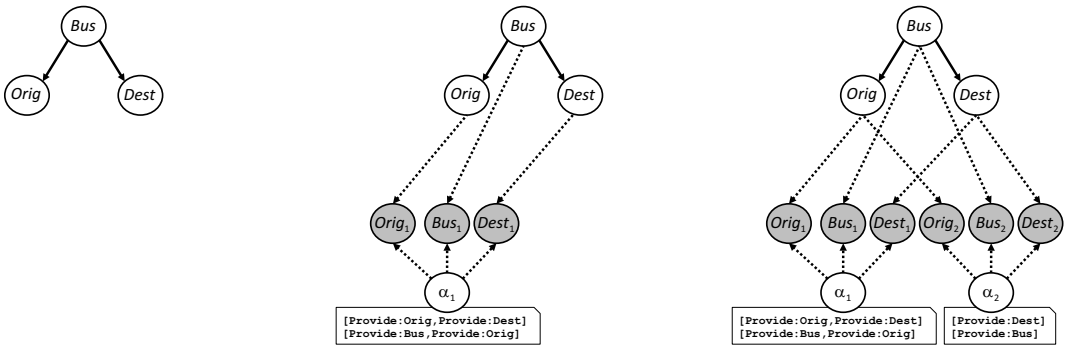
We evaluated the proposed model on a publicly available corpus of dialogs from the CMU Let's Go Bus Information System. The corpus is the same as the one described in [8]. It contains 4305 dialogs manually transcribed and semi-automatically annotated with dialog acts, split into a training set of 4010 dialogs and a test set of 260 dialogs (combining the development and test sets of [8]). The original system used to collect the data makes systematic use of explicit confirmations. However, because we want to evaluate the ability of our model to track the user goal from noisy data, we eliminate all the confirmation turns from the corpus (i.e. we skip them when running a dialog). This results in a total of 904 turns in the test set.

To conduct our batch experiments, we first run ASR on the whole corpus. We use Nuance 8.5 for ASR, with a class-based, context-dependent statistical language model trained on the utterances from the training set[1]. We tune the parameters of the

[1] Although the original Let's Go system uses Sphinx for ASR, we decided to use Nuance because it provides N-best lists and better confidence scores, which are critical to our approach.

True utterance	Top 2 ASR/NLU Hyps	Top 2 Belief Hyps
S: What can I do for you?	[Orig=NEGLEY,Dest=MURRAY]	[Bus=N/A,Orig=NEGLEY,Dest=MURRAY]
U: 61C from CMU.	[Bus=61C,Orig=CMU]	[Bus=61C,Orig=CMU,Dest=N/A]
S: Where to?	[Dest=OAKLAND]	[Bus=61C,Orig=CMU,Dest=OAKLAND]
U: Oakland.	[Bus=77H]	[Bus=N/A,Orig=NEGLEY,Dest=OAKLAND]

Figure 1: Example dialog in a simplified bus information domain.



(a) Initially (b) After Turn 1 (c) After Turn 2

Figure 2: Example Dynamic POT for 3 concepts for the dialog of Figure 1. Observed nodes are shaded in gray. Solid arrows represents edges of the Goal Network and dashed arrows edges of the Evidence Networks.

engine and set it to return N-best lists of length 100 without any confidence-based rejections.

4.2. Parameter Estimation and Tuning

In Let's Go, the Goal Network has 5 concepts: `Bus Number`, `Departure Neighborhood`, `Departure Stop`, `Arrival Neighborhood`, and `Arrival Stop`. As in [8], all the CPTs (e.g. $P(DepartureNbhd = OAKLAND|BusNum = 61C)$) are estimated from the user queries of the training set. We also use the training corpus to estimate dialog act prior probabilities for the α nodes. Because we define the value set of each α node dynamically based on the ASR N-best list, we simply keep the raw counts of dialog act combination in the training set, and normalize them at runtime for each α node (with add-one smoothing).

The observation node CPTs are computed from the utterance-level confidence scores returned by Nuance[2]. While it is not possible to directly derive $P(C_t^i|C^i, \alpha_t)$, we approximate these likelihood scores using the following function:

$$P(C_t^i|C^i = c, \alpha_t = a) \quad = \quad (\frac{\nu}{100})^P \qquad (1)$$

where ν is the confidence of the highest scoring ASR hypothesis that corresponds to α-value a and value c for concept C^i. In order to avoid zero probabilities for values $C^i = c$ that do not appear in any hypothesis, their likelihood is set to ϵ^P, where ϵ is a constant, if at least one other value for C^i appears in the N-best list. If no value for C^i appears in any ASR hypothesis, we note that $P(C_t^i|C^i = c, \alpha_t = a) = P(C_t^i|\alpha_t = a)$ (since the likelihood is independent of the value of C^i). We approximate this likelihood by that of the best hypothesis matching $\alpha_t = a$. The intuition behind Equation 1 is that by taking the power P of the scaled confidence score, we spread the scores more effectively: low scores get reduced more than high scores. This leaves us with two parameters to tune: P and ϵ. To perform this tuning, we first ran our belief tracker on the whole set using many parameter settings ($1 \leq P \leq 13$ and $0 \leq \epsilon \leq 0.3$). We then perform grid search using 10-fold cross-validation as follows. For each iteration, we use 90% of the data to select the settings with minimal error, and compute the accuracy on the remaining 10%. The parameters are quite stable: the best settings are the same for every fold ($P = 7$, $\epsilon = 10^{-3}$). We use these parameter values in the remainder of this paper.

[2]Nuance confidence scores are integers between 1 and 100.

4.3. Baseline and Evaluation Metric

The baseline belief tracker maintains a single value for each concept of the user goal. At the start of each dialog, all concepts are set to "`N/A`" (unspecified). Then at each turn, concept values that appear in the top ASR hypothesis overwrite the corresponding concepts in the user goal.

Ground truth for evaluation is established in a similar manner, except that each turn's manual semantic labels are used to update the user goal. Therefore, ground truth represents the user goal *as specified so far* by the user. Evaluation results are given in terms of Concept Belief Error Rate (CBER). CBER is computed at each turn by matching for each concept the system's top belief with the ground truth. If the ground truth is "`N/A`" but the belief has a value, we count one insertion error. Conversely, if the ground truth has a value but not the belief, we count one deletion error. If both have values but they do not match, we count one substitution error. We divide the number of errors per turn by the number of concepts in the domain and obtain for each type of error a rate between 0 and 1. Finally, we average the rate over all turns of the test set.

4.4. Results

As shown in Table 1, D-POT(1-best), i.e. the system's top belief, improves over the baseline by a relative 9.3% in overall error rate ($p = 0.05$ using Wilcoxon's Sign Rank Sign Test on

Table 1: *Concept belief error rate of D-POT at various depth of belief m-best list. The best results for each error rate are in bold.*

Version	Deletions	Insertions	Substitutions	Overall
Baseline	**3.2%**	4.6%	8.3%	16.1%
D-POT (1-best)	4.2%	3.4%	7.0%	14.6%
D-POT (2-best)	3.6%	2.8%	6.4%	12.9%
D-POT (3-best)	3.4%	**2.7%**	**6.2%**	**12.2%**

CBER averaged per dialog). To better assess the quality of the belief distribution (and not only the top-ranked hypothesis), we compute the performance when using an oracle to select the best hypothesis from the top 2 and 3 (D-POT(2-best) and D-POT(3-best)). As shown in Table 1, just considering the top 2 hypotheses doubles the relative reduction in error to 19.9%, indicating that D-POT is maintaining robust beliefs which can be used by a probabilistic dialog manager. Table 1 also shows that D-POT has a higher deletion rate than baseline but that the insertion errors and substitution errors have been reduced by a significant amount, leading to the overall performance improvement.

To further understand how D-POT exploits the ASR/NLU N-best list, we conducted evaluations using different length of N-best list. Figure 3 shows the relative change in performance when using increasingly longer N-best lists as input. Overall error rate keeps decreasing even when including as many as twenty different semantic hypotheses at each turn, including many erroneous ones. This indicates that, by using prior information about the domain (the Goal Network CPTs) and user behavior (the α node priors), D-POT is able to extract useful information from noisy input. Deletions increase significantly when the first few hypotheses are considered, whereas insertions and substitutions keep decreasing even for long lists.

The relatively large increase in deletions can be explained by the fact that, based on our training data, the prior probabilities for α-values with only one dialog act are very high, whereas the dynamic range of confidence scores returned by Nuance is fairly small. Our mapping function (elevating to the power of P) addresses some of this problem but does not completely solve it. By obtaining more appropriate probabilities from the ASR/NLU, many of these deletions could be avoided, and the overall performance significantly improved.

Finally, since we intend D-POT to be used in interactive systems, we verified the efficiency of the inference algorithm. Using only one core of an Intel i7 2.8 GHz processor, for 96.2% of the turns, inference took less than 500 ms and in 98.2%, less than 1 s. While the algorithm could potentially be improved by parallelizing the sampling process, these results confirm the practicality of our approach.

5. Conclusion

We presented Dynamic Probabilistic Ontology Trees, a new probabilistic model of dialog for the purpose of probabilistic belief tracking. In addition to representing the (static) user goal, the model explicitly captures the user's dialog acts at each turn. This structure, along with a specialized sampling algorithm that we designed, allows it to efficiently compute accurate beliefs from potentially large N-best lists from the ASR/NLU modules. In addition to the user goal, these beliefs can bear upon many aspects of dialog history, which could prove very useful to statistical dialog managers.

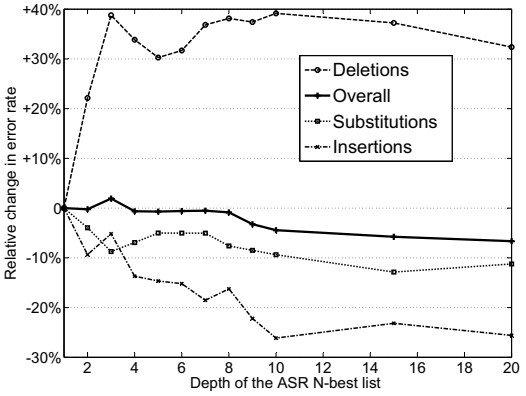


Figure 3: Variation in concept belief error rate as a function of the maximum number of semantically distinct ASR hypotheses considered. Negative changes mean better performance.

6. References

[1] E. Horvitz and T. Paek, "Harnessing Models of Users' Goals to Mediate Clarification Dialog in Spoken Language Systems," in *International Conference on User Modeling*, 2001.

[2] J. D. Williams, "Using Particle Filters to Track Dialogue State," in *IEEE Workshop on Automatic Speech Recognition & Understanding*, 2007.

[3] B. Thomson and S. Young, "Bayesian Update of Dialogue State: A POMDP Framework for Spoken Dialogue Systems," *Computer Speech and Language*, vol. 24, no. 4, pp. 562–588, 2009.

[4] N. Mehta, R. Gupta, A. Raux, D. Ramachandran, and S. Krawcyk, "Probabilistic ontology trees for belief tracking in dialog systems," in *SIGdial 2010*, 2010.

[5] S. Young, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The Hidden Information State Model: A Practical Framework for POMDP-based Spoken Dialog Management," *Computer Speech and Language*, vol. 24, no. 2, pp. 150–174, 2010.

[6] J. D. Williams, "Incremental partition recombination for efficient tracking of multiple dialog states," in *ICASSP*, 2010.

[7] C. S. Jensen, A. Kong, and U. Kjærulff, "Blocking-Gibbs sampling in very large probabilistic expert systems," *International Journal of Human-Computer Studies*, vol. 42, pp. 647–666, 1995.

[8] A. Raux, N. Mehta, R. Gupta, and D. Ramachandran, "Dynamic language modeling using bayesian networks for spoken dialog systems," in *Proc. Interspeech 2010*, 2010.