INTERSPEECH 2009 BRIGHTON

Using Graphical Models for Mixed-Initiative Dialog Management Systems with Realtime Policies

Stefan Schwärzler [†] *, Stefan Maier* [†] *, Joachim Schenk, Frank Wallhoff, Gerhard Rigoll*

Institute of Human Machine Communication
Technische Universität München, Germany

{sts,mai,joa,waf,ri}@mmk.ei.tum.de

Abstract

In this paper, we present a novel approach for dialog modeling, which extends the idea underlying the partially observable Markov Decision Processes (POMDPs), i. e. it allows for calculating the dialog policy in real-time and thereby increases the system flexibility. The use of statistical dialog models is particularly advantageous to react adequately to common errors of speech recognition systems. Comparing our results to the reference system (POMDP), we achieve a relative reduction of 31.6% of the average dialog length. Furthermore, the proposed system shows a relative enhancement of 64.4% of the sensitivity rate in the error recognition capabilities using the same specifity rate in both systems. The achieved results are based on the Air Travelling Information System with 21 650 user utterances in 1 585 natural spoken dialogs.
Index Terms: dialog modeling, dialog strategy, spoken language understanding

1. Introduction

In modern dialog management systems Markov Models, e. g. partially observable Markov Decision Processes (POMDPs), have proven their power for modeling sequences of naturally spoken dialogs [1, 2]. This is, because speech recognition technology remains imperfect: recognition errors are common and influence dialog management systems. Motivated by the advantages of POMDPs in comparison to conventional systems [3], we design a novel approach which is similar to the POMDP model, and additionally, allows for calculating the dialog policy in real-time. Thus, the flexibility of the system is increased: dialog strategies can be adjusted during the runtime, e. g. to the user behavior. In the proposed model, we use semantic slots, as introduced in [4]. These represent a cluster of sequences of words with a specific meaning, which are determined during the runtime.

Several semantic slots at a time define a dialog state whose transitions are called semantic pairs. Temporally consecutive semantic pairs form the trellis through a dialog. The user is led to a defined (or learned) user goal by well-directed questions asked by the system. The parameters of the dialog model are learned automatically from a annotated training set, which is introduced in Sec. 2. In Sec. 3, we describe the concept of semantic slots. In Sec. 4, the dialog model based on a trellis is presented. In Sec. 5, we introduce the POMDP reference model. Finally in Sec. 7, we compare the performance of our model to the one of the reference model, before we conclude in Sec. 8.

2. Corpus Description and Preprocessing

The data of this work are extracted from the air traveling information system (ATIS) [5]. The information about the real user goal in each dialog are encoded in the description of the scenarios of the ATIS log files. For the experimental results in the dialog process, we extract all 21 650 natural spoken questions and answers with the aid of keyword searching in ATIS log files. The transitions in the dialog model are learned from 1 585 spoken dialogs in the training phase.

In order to use the ATIS corpus for evaluating a dialog system, preprocessing as described in the following is needed. First, the system's answers are extracted with keyword spotting in the ATIS log files and their SQL-queries. In sum, $N(S) = 10$ concepts are hereby identified. As our novel dialog system is based on a trellis, this yields $N(T) = 2^{10} = 1\,024$ possible transitions or states. This number of states is computationally infeasible when POMPDs are used as a reference system. Hence, the system is reduced to eight semantic slots represented in a trellis with $N_\mathrm{r}(T) = 2^8 = 256$ states.

In order to provide an adequate training (see Sec. 6), for all semantic pairs $s_i|s_{i-1}$ a probability $p(s_i|s_{i-1})$ is estimated from the corpus. Due to its sparseness, however, not all semantic pairs occur in the corpus. Hence, we perform absolute discounting in conjunction with backing-off [6] in a second step. Therefore, the absolute count $N(s_i|s_{i-1})$ of each semantic pair is reduced by a fixed value β, if it is bigger than a threshold c, reducing the marginal probability to $\sum_{s_i \in \mathbf{T}} p(s_i|s_{i-1}) < 1$. The remaining probability "mass" $1 - \sum_{s_i \in \mathbf{T}} p(s_i|s_{i-1})$ is distributed among the semantic pairs, which are *not* found in the corpus. In addition, the backing-off factor c defines the minimum number of counts $N(s_i|s_{i-i})$ for using the semantic pair $s_i|s_{i-i}$ in the probability calculation

$$p(s_i|s_{i-1}) = \begin{cases} {}^{N(s_i|s_{i-1})-\beta}/_{N(s_{i-1})} & \text{if } N(s_i|s_{i-1}) > c \\ b(s_i, s_{i-1}) \cdot p(s_i) & \text{otherwise,} \end{cases} \quad (1)$$

with $b(s_i, s_{i-1}) = \frac{1 - \sum_{s_i \in \mathcal{B}(s_{i-1})} p(s_i|s_{i-1})}{1 - \sum_{s_i \in \mathcal{B}(s_{i-1})}}$ and $\mathcal{B}(s_{i-1}) = \{s_i | s_i, s_{i-1} \in \mathcal{S} \wedge N(s_i|s_{i-1}) > c\}$.

In the rare case that no information on the state s_i is available (i. e. the semantic knowledge represented by this state never occurs) in the corpus, the probability $p(s_i)$ is derived from a manually designed priorization matrix. It shall be noted that during each transition exactly one or no additional information slot is filled.

3. Semantic slots

User utterances are decoded as described in [4] and mapped to so-called "semantic slots". Each concept is represented by one semantic slot in a binary manner, and the assignment `Originating City` ↦ slot 1, `Destination City` ↦ slot 2, `Price Information` ↦ slot 3, `Date` ↦ slot 4, `Time` ↦ slot 5, `Class` ↦ slot 6, `Trip Type` ↦ slot 7, and `Stopover` ↦ slot 8 between the concepts and the semantic slots holds. Hence, each semantic slot denotes a binary digit.

3.1. Dialog states $\mathcal{S}$

Each combination of semantic knowledge, represented by the settings of the corresponding semantic slots, is denoted as *dialog state* s_i, where i is the decimal value of the semantic slots an yields $i = v(\text{slot } 1) \cdot 2^0 + v(\text{slot } 2) \cdot 2^2 + \cdots + v(\text{slot } 8) \cdot 2^8$, where $v(\text{slot } j \in \{0; 1\}$ is the binary value of the semantic slot j. Hence, with N semantic slots, $N(s) = 2^N$ possible dialog states is the result.

3.2. Oberservations $\mathcal{O}$

Given observations o from the user are encoded as semantic slots in the same way as dialog states s. Observations o_t are added cumulatively in every time step t to a observation state $o_T = \sum_t^T o(\text{Slot x}) \forall x \in \mathcal{O}$.

3.3. User goals $\mathcal{G}$

A user goal $g_t \in \mathcal{G} \subseteq S$ is defined as a subset of a possible database request, which is either learned from a training corpus or manually designed, e. g.:

Origin ∧ Destination ∧ Price ∧ Round trip

[†]Note: Both authors contributed equally to this paper.

In our approach the user goals g were deduced manually from the corpus (see Fig. 1).

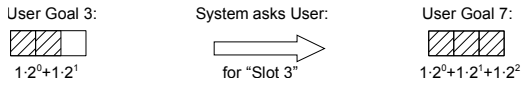

Figure 1: User goal g and transitions deduced from the ATIS corpus: encoded user goal 3 aims to reach user goal 7 by asking the user for slot 3.

4. Dialog Control Modeling

Mixed-initiative dialogs allow the estimation and the achievement of the user goals g to create a SQL statement. The dialog strategy is derived from a Graphical Model (GM), which combines the theory of probabilities and graphs. Edges model the statistical dependencies between the variables (nodes). Fig. 2 shows the GM of a mixed-initiative dialog system with observable prolog state s_0 and epilog state s_T. The hidden dialog states $s_1, \ldots, s_{T-1}$ in between are parents of semantic observations, like in an ordinary HMM. The future flag node determines the kind of connection between the hidden states and the observations. Therefore, the observations are split into a past $o_{0:t-1}$, present o_t and future part $o_{t+1:T}$. The past part grows with every answer given. The present part is always of the size of one time step and the future part is of the size of the slots missing when comparing the present state with the user goal g. The basic idea is, that time steps in the future use a different observation matrix than past time steps. In this implementation, future observations are virtually disconnected from their hidden states, so the Viterbi search is based only on the transition matrix in the future steps.

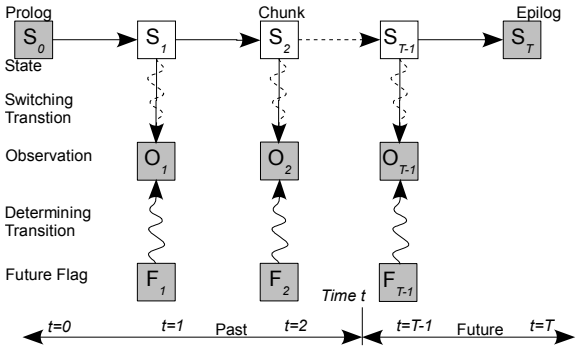

Figure 2: GM of the mixed-initiative dialog control system.

4.1. User goal estimation

In Fig. 2, we divide the dialog into three parts: First, the past dialog states $s_0, \ldots, s_t$ with their observable semantic slots use a heuristic distribution (Gaussian) for the estimated user goals. Second, if the present part of the dialog sate s_t is equal to an user goal g, the system returns either the results of the database request to the user or continues the dialog by filling additional semantic slots, i. e. the user's answers give more detailed information. Third, in the future part, we assume that g is the user goal, which can be reached with the fewest additional semantic slots being filled. By applying the Viterbi algorithm, we find the minimum number of future dialog states $s_{t+1}, \ldots, s_T$ necessary to reach the user goal g. If the current state s_t is identical to the user goal g the system lists the result of the database request to the user. The dialog may be continued either user- or system-initiated with a further user goal g^*.

4.2. Finding best dialog path

Beginning at the desired user goal, i. e. at the expected end of the dialog, the best series of observations $o_1, \ldots, o_T$ is found by the well-known Viterbi algorithm [7]. Because the user should be asked only for one semantic slot at a time, the system is limited to transitions that add one slot in each time step t. The Viterbi algorithm herein is realized as follows:
Initialization:

$$\begin{matrix}\delta_1(i) = p(s_i|o_1) \\ \psi_1(i) = 0\end{matrix} \quad , 1 \leq i \leq N. \tag{2}$$

Recursion:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) \cdot t_{ij}] \cdot p(s_j|o_t) \quad , \quad \begin{matrix}2 \leq t \leq T \\ 1 \leq j \leq N\end{matrix} \tag{3}$$

Termination:

$$\hat{q}_T = \operatorname*{argmax}_{1 \leq i \leq N} \delta_T(i) \tag{4}$$

The best path, i. e. the sequence of observations that are mapped to the semantic slots, is then found by backtracking:

$$\hat{q}_t = \psi_{t+1}(\hat{q}_{t+1}). \tag{5}$$

Taking the best path into account, the transition $s_t \rightarrow s_{t+1}$ can be obtained, and the semantic slot that is to be filled in time step $t+1$ is known (see Fig. 3). Hence, the system selects the action $a_{t+1} \in \mathcal{A}$ (which can e. g. be a question) to gain the desired information from the user.

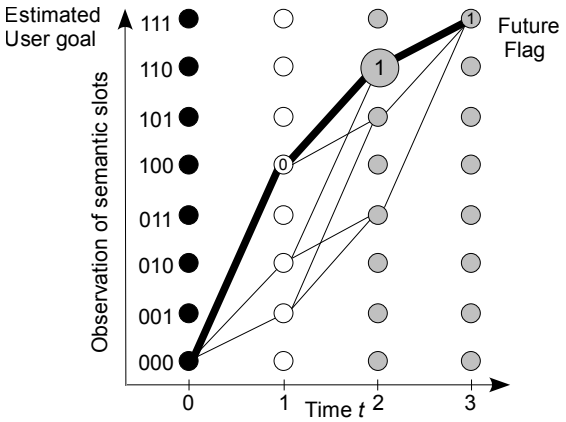

Figure 3: Trellis diagram of the modeled GM: The estimated user goal g_3 is 7 (all slots filled: $(1\,1\,1)$), the small lines are all possible paths through the trellis, stressed is the line that is found by the Viterbi algorithm. Therewith, the best path from the actual state $s_1 = (1\,0\,0)$ to the most likely next state $s_2 = (1\,1\,0)$ is estimated. Assuming a larger model with more slots, after reaching the first goal $g_1 = (1\,1\,1)$ the system either requests a SQL statement or estimates the next user goal g_2 from the best path.

4.3. Semantic slot re-estimation

The Viterbi algorithm finds the best past path through all time steps. Therewith, semantic information in past $o_1, \ldots, o_{t-1}$ is re-considered when the new information o_t is observed, by applying the transition rules $\mathbf{T}_p$ for past and $\mathbf{T}_f$ for future observations. While also self-transitions are allowed for the past frames, the transitions in $\mathbf{T}_f$ are restricted to deliver exactly one piece of semantic information per time step. Furthermore, recurrences in the dialog can be added through insertion of "ones" on the main diagonal of the matrix $\mathbf{T}_p$.

4.4. Influence and recognition of errors

Applying the Viterbi algorithm, the model has the ability to detect and correct erroneous inputs from a speech recognizer or a semantic interpretation. With the semantic slot re-estimation the system will probably choose other semantic slots in past time $t_{0:t-1}$ with $\mathbf{T}_p$ as recommended from the semantic interpretation unit, which is presented in [4].

5. Reference Model

In [2], a model-based approach to dialog management is described. The POMDP is a Markov Decision Process (MDP), which will be extended

in such a way that the states are not observable. This is done in a similar way as for the Hidden Markov Model (HMM), where the states of the Markov chain are hidden and an observation is linked to the states with certain probabilities. In contrast to the MDP, the POMDP does not know the current state with certainty. Hence, for each state its probability is inferred based on the current observation. In contrast to the HMM, these are Markov decision-making processes, where actions can be executed and change their environment accordingly. The problem of a dialog management system means that the system itself can perform certain actions, usually asking questions or producing results. POMDPs are suitable to model this behavior. The POMDP consists of a set of States $\mathcal{S}$, a set of observations $\mathcal{O}$, and a set of actions $\mathcal{A}$ (see graphical model in Fig. 4).

During a training phase (see Sec. 6), these parameters are estimated and form a transition matrix $\mathbf{T} = P(s'|a, s)$, which gives the probability of the current state s' subject to the condition of the previous state s and the executed action a. During the training phase, one determines the reward as a function of the state s and the executed action a in a reward matrix $\mathbf{R}(s, a, s')$. The observation matrix $\mathbf{Z} = P(o'|s', a_m)$ reflects the observation conditioned on the current state and the previous action. From all three matrices generated during the training phase, $\mathbf{T}, \mathbf{R}, \mathbf{Z}$, a policy is created. Later during operation, a suitable vector $\mathbf{b}$ can be constructed from it. In addition to the policy, a geometric discount factor λ is considered, which enhances the weight of the rewards that are closer to future times. This factor is important in agent-based systems, as a possible change of topic and corrections could lead to infinitely long dialogs. For initialization an additional belief state $\mathbf{b}_0$ is defined (see Fig. 4, right).

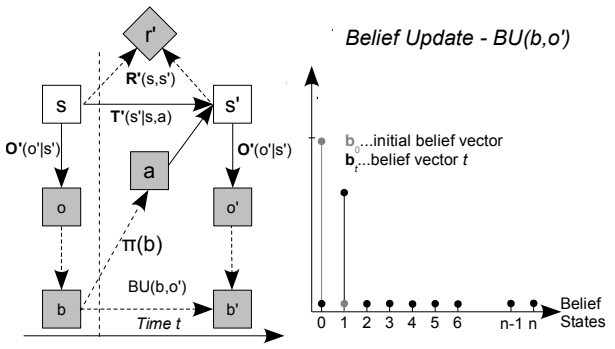

Figure 4: Left: BU: Belief update, $\pi(b)$: Policy Right: Graphical model of a POMDP, dashed lines: deterministic links, continuous lines: probabilistic links.

5.1. Belief update

As shown in Fig. 4, during the operation phase the appropriate action a_m is chosen according to the current state belief $\mathbf{b}_t$, and the state will be changed to the next state s'. There, an observation of semantic slots o takes place, and the belief state $\mathbf{b}_t$ is updated following Eq. 6, where k is a normalization factor.

$$b'(s') = k\, P(o'|s', a_m) \sum_{s \in S} P(s'|a_m, s)\, b(s). \tag{6}$$

6. Training

The parameters of the transition matrix $\mathbf{T}$ are derived from the described ATIS corpus. To compare both approaches, we preferred a (Gaussian) distribution of the observation $\mathbf{o}_t$. Hence, the probabilities of the observation matrix $\mathbf{O}$ are learned from the ATIS corpus and mixed with the described distribution. The parameters of matrix $\mathbf{R}$ used in the POMDP model are manually designed. To avoid repetitions $s_x \rightarrow s_x$, we use a high penalty ($p = 10$) in $\mathbf{R}$. The matrices are online available [1].

[1] http://www.mmk.ei.tum.de/∼sts/corpus

7. Experiments and Results

Both, the GM- and the POMPD-based dialog system, use the same transition $\mathbf{T}$ and observation $\mathbf{O}$ matrices for the experiments. Furthermore, a user model was set up with a user goal estimator and a user reaction model. First, the user goal estimator determines which slots have to be filled at the end of the dialog. Second, the user reaction model describes with manually designed rules, how the user reacts to system questions. The results of the experiments were achieved using the POMDP toolkit [8]. The dialog policy π was calculated with the aid of SARSOP [9] at the LRZ[2] supercomputer.

7.1. Average dialog length

The evaluated 1 990 dialogs contain 8 717 user utterances (UA) which means that the average dialog length is 4.35 UAs. The POMDP model with 2 000 dialogs resulted in 12 901 UA, thereby the average dialog length is 6.4 UAs. Compared to POMDP we require 31.6% less UAs to reach the same number of semantic slots. Therefore, the user goal g is achieved with significantly fewer questions and significantly faster.

7.2. Error recognition of GM-based vs. POMDP dialog control systems

In this paper, the error correctness based on the annotated user goals g in the corpus is evaluated. For that end, 317 dialogs ($\widehat{=}$ 20%) in ATIS are used for a statistical user model. The number of correctly decoded dialogs increases, if every decoded dialog step $d(s_{\text{dec}})$ is the same as the input from the simulation $d(s_{\text{in}})$ (see eq. 7).

$$d(s_{\text{in}}, s_{\text{dec}}) = \begin{cases} 1 & \text{if } s_{\text{in}} == s_{\text{dec}}, \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Our approach also corrects errors, e. g. from the speech recognizer, with Viterbi decoding. The number of correct decoded dialogs increases, if the input signal contains errors and the dialog management system revises the semantic slots rightfully:

$$d_{\text{corr}}(s_{\text{noisy}}, s_{\text{dec}}) = \begin{cases} 1 & \text{if } s_{\text{noisy}} == s_{\text{dec}}, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

330 of 8 717 UA were distorted by an error, which means the slot received at the dialog manager was different than that sent out by the user. This led to a total of 1 696 dialogs that were error-free and 294 dialogs that contained at least one error. According to the definition of error recognition $d(corr)$ the system decided that the dialog has an error in 229 cases and that the dialog is error-free in $d(corr) = 1\,761$ cases. The error recognition capabilities are the input of the well-known F1 measurement method (see eq. 9) and facilitate a comparison with the POMDP model in tab. 1: Out of the 229 cases detected as defective(P), 115 in fact contained an error (TP), the remaining 179 were classified as negative (FN). This leads to a sensitivity of 0.39. The relevance, as defined as the number of TP to number of all P is 0.5. Out of the 1 761 cases classified as negative (N), 179 were defective(FN) and 1 582 were in fact error-free (TN). The corresponding specifity is 0.93. The segregance, which expresses the probability that a negative test result is correct, is at a level of 0.90. Altogether, this leads to a correct classification rate of 0.85.

$$\begin{aligned} TP &= \{d(s_{\text{in}}, s_{\text{dec}}) == 0\} \wedge \{d_{\text{corr}}(s_{\text{noisy}}, s_{\text{dec}}) == 0\} \\ FP &= \{d(s_{\text{in}}, s_{\text{dec}}) == 1\} \wedge \{d_{\text{corr}}(s_{\text{noisy}}, s_{\text{dec}}) == 0\} \\ FN &= \{d(s_{\text{in}}, s_{\text{dec}}) == 0\} \wedge \{d_{\text{corr}}(s_{\text{noisy}}, s_{\text{dec}}) == 1\} \\ TN &= \{d(s_{\text{in}}, s_{\text{dec}}) == 1\} \wedge \{d_{\text{corr}}(s_{\text{noisy}}, s_{\text{dec}}) == 1\} \end{aligned} \tag{9}$$

In the 2 000 tested dialogs from the POMDP model, 1 688 are error-free and 312 have an error. For a comparison with the GM-based dialog management, we calculate the average of the probabilistic belief states $\mathbf{b}(s)$ in every time step t. Hence, we compute the receiver operation characteristic (ROC) curve [10] by matching this value to a variable threshold. The advantage is that a variable threshold can be used to optimize the system (see fig. 5).

In order to compare it with the GM-based dialog management system, the specifity is adjusted to the same value 0.93, which results into a threshold of 0.64. Hence, in the 177 cases that were detected as defective (P), 56 in fact contained an error (TP), the remaining 121 were

[2] Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities.

		GM	POMDP
Number of dialogs $N(d)$		1 990	2 000
User utterances (UA)		8 717	12 901
Average dialog length $\varnothing l(d)$		4.38	6.45
Correctly classificated dialogs d_{corr}		1 761	1 688
F1 measure	True Positive (TP)	115	56
	False Positive (FP)	114	121
	False Negative (FN)	179	256
	True Negative (TN)	1 582	1 567
Sensitity(SES)		0.39	0.18
Specifity (SPC)		0.93	0.93
False positive rate (FPR) [%]		0.07	0.07
False negative rate (FNR) [%]		0.61	0.82
Relevance (PPV) [%]		0.50	0.32
Segregance (NPV) [%]		0.90	0.86
False discovery rate (FDR) [%]		0.15	0.19
True discovery rate (1 − FDR) [%]		0.85	0.81

Table 1: GM-based vs. POMDP [2] dialog management system compared on their error recognition capabilities.

classified as negative (FN). This leads to a sensitivity of 0.18 and a relevance of 0.32. On the other side, we have 1 823 negatives (N), 1 567 were really error free(TN), 256 FN. This leads to the same specifity as in GM-based dialog management systems (0.93). The segregance is at a level of 0.86. Altogether, the correct classification rate is 0.81 (see tab. 1).

We estimate our approach in comparison to the POMDP-model by using the receiver operating characteristic (ROC) curve [10]. To compute the specifity and sensitivity of the 2 000 tested dialogs in the POMDP-based dialog model we increase the threshold by 0.1 from 0 to 1 stepwise. The POMDP-based dialog model is significant better than a random guess of the semantic slots (straight line in fig. 5). Our system (GM-based) outperforms the POMDP-based model by 64.4% relatively in the sensitivity rate at the same SPC (0.93).

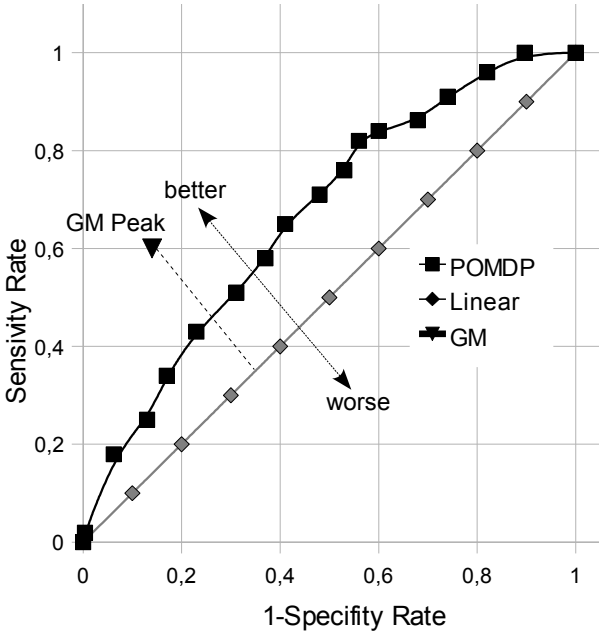

Figure 5: Receiver Operating Characteristic Curve for GM-based vs. POMDP model: As expected TPR (or sensitivity) of the POMDP-based dialog model performs better than random decisions in any case. In a comparison of both approaches in SPC=0.93, the GM-based dialog model outperforms the POMDP approach by relatively 64.4% in the sensitivity rate. The single point for GM instead of a curve is due to the classificator that is not based on a variable threshold. Enhanced classificators would also allow a threshold decision and therefore a real ROC curve.

8. Conclusions and Outlook

In this paper, a novel GM-based dialog management system was presented and compared to the state-of-the-art system POMDP . In both setups, the probabilities of the dialog model were estimated using naturally spoken sentences in the ATIS corpus. The systems were compared by their number of dialog steps taken on average and by semantic slot error recognition capabilities. To this end, a user model, a user goal estimator, and a user reaction model were set up for comparable and objective system tests. The GM-based dialog system showed a significantly higher sensitivity rate in the error recognition capabilities, by a relative enhancement of 64.4%, in comparison to the POMDP system. In addition, the average dialog length (an estimation criterion for short targeted dialogs) was reduced, i. e. the presented system requires relatively 31.6% less dialog steps than the POMDP system. In the future, we plan to correct speech recognition errors based on the proposed recognition capabilities of the GM-based dialog approach, and thus we will superiorly utilize the potential of graphical models. To this end, we plan to integrate the proposed approach into a multi-agent framework [11].

9. References

[1] E. Levin, R. Pieraccini, and W. Eckert, "Using markov decision processes for learning dialogue strategies," in *In Proc Int Conf Acoustics,Speech and Signal Processing*, Seattle,USA, 1998.

[2] S. Young, "Using POMDPs for dialog management," in *IEEE/ACL Workshop*, Palm Beach, Aruba, 2006.

[3] J. D. Williams, "The best of both worlds: Unifying conventional dialog systems and POMDPs," in *Proceedings of the 9th International Speech Communication Association (Interspeech 2008)*, 09 2008, pp. 1173–1176.

[4] S. Schwärzler, J. Geiger, J. Schenk, M. Al-Hames, B. Hörnler, G. Ruske, and G. Rigoll, "Combining Statistical And Syntactical Systems For Spoken Language Understanding With Graphical Models," in *Proceedings of the 9th International Speech Communication Association (Interspeech 2008)*, Brisbane, Australia, 09 2008, pp. 1590–1593.

[5] C. Hemphill, J. Godfrey, and G. Doddington, "The ATIS Spoken Language Systems Pilot Corpus," in *in Proceedings of the DARPA Speech and Natural Language Workshop*. Morgan Kaufmann, 1990, pp. 96–101.

[6] S. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 35, no. 3, pp. 400 – 401, 1987.

[7] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimumerror bounds for convolutional codes and an asymptotically optimum decoding algorithm," in *IEEE Transactions on Information Theory*, ser. 13, 1967, pp. 260–267.

[8] T. Taha, "POMDP Toolbox," Clemson University, Clemson, USA, Tech. Rep., 2007. [Online]. Available: www.tarektaha.com

[9] H. Kurniawati, D. Hsu, and W. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proceedings Robotics: Science and Systems*, 2008.

[10] D. Green and J. Swets, *Signal detection theory and psychophysics*. John Wiley and Sons Inc., 1966.

[11] S. Schwärzler, J. Schenk, G. Ruske, and F. Wallhoff, "A Multi-Agent Framework for a Hybrid Dialog Management System," in *Int. Conf. on Multimedia & Expo ICME 2009*. New York, USA: IEEE, 2009.