Comparison of Bayesian Discriminative and Generative Models for Dialogue State Tracking

Lukáš Žilka, David Marek, Matěj Korvas, Filip Jurčíček
Charles University in Prague
Faculty of Mathematics and Physics
Malostranské náměstí 25
118 00 Praha, Czech Republic
lukas@zilka.me, david@marek.me,
korvas@ufal.mff.cuni.cz, jurcicek@ufal.mff.cuni.cz

Abstract

In this paper, we describe two dialogue state tracking models competing in the 2012 Dialogue State Tracking Challenge (DSTC). First, we detail a novel discriminative dialogue state tracker which directly estimates slot-level beliefs using deterministic state transition probability distribution. Second, we present a generative model employing a simple dependency structure to achieve fast inference. The models are evaluated on the DSTC data, and both significantly outperform the baseline DSTC tracker.

1 Introduction

The core component of virtually any dialogue system is a dialogue state tracker. Its purpose is to monitor dialogue progress and provide compact representation of the past user input and system output in the form of a dialogue state. In previous works on this topics, Williams (2007) used particle filters to perform inference in a complex Bayesian network modelling the dialogue state, Williams (2008) presented a generative tracker and showed how to train an observation model from transcribed data, Young et al. (2010) grouped indistinguishable dialogue states into partitions and consequently performed dialogue state tracking on these partitions instead of the individual states, Thomson and Young (2010) used a dynamic Bayesian network to represent the dialogue model in an approximate form, and Mehta et al. (2010) used probabilistic ontology trees.

In this paper, we describe two probabilistic dialogue state trackers: (1) a *discriminative dialogue state tracker* (DT) – a model using a simple deterministic state transition probability, resulting in significant computational savings, and (2), a *generative dialogue state tracker* (GT) – a model using simple conditional dependency structure with tied and handcrafted model parameters. Both trackers were evaluated in the DSTC. The aim of the DSTC was to provide a common testbed for different dialogue state tracking methods and to evaluate these methods in a unified way. Because of limited space, the interested reader is referred to Williams et al. (2013) for information about the data and evaluation metrics used in the challenge.

This paper is structured as follows. The deterministic and generative trackers are detailed in Section 2 and the presented models are evaluated on the DSTC data in Section 3. Section 4 discusses the obtained results, and Section 5 concludes the paper.

2 Bayesian Dialogue State Tracking

The goal of dialogue state tracking is to monitor progress in the dialogue and provide a compact representation of the dialogue history in the form of a dialogue state. Because of the uncertainty in the user input, statistical dialogue systems maintain a probability distribution over all dialogue states called the *belief state* and every turn, as the dialogue progresses, updates this distribution in the light of the new observations in a process called *belief monitoring*.

Since the true observations are hidden, the belief state depends on the past and current observation probabilities, $p(\mathbf{o}_1), \ldots, p(\mathbf{o}_t)$, and system actions, $a_0, \ldots, a_{t-1}$, which are referred to as the observed history: $h_t = \{a_0, p(\mathbf{o}_1), \ldots, a_{t-1}, p(\mathbf{o}_t)\}$. If the system is Markovian, the belief state b_t depends only on the previous belief state b_{t-1}, the observation distribution $p(\mathbf{o}_t)$, and the last system action a_{t-1}. There are two ways to derive the belief state update using the Bayes theorem, resulting either in discriminative or generative probabilistic models.

The discriminative update can be represented as follows:

$$\begin{aligned} b_t &= b(\mathbf{s}_t|h_t) \\ &= \sum_{\mathbf{s}_{t-1},\mathbf{o}_t} p(\mathbf{s}_t|a_{t-1},\mathbf{s}_{t-1},\mathbf{o}_t) b(\mathbf{s}_{t-1}|h_{t-1}) p(\mathbf{o}_t) \end{aligned} \quad (1)$$

where the probability $p(\mathbf{s}_t|a_{t-1},\mathbf{s}_{t-1},\mathbf{o}_t)$ represents the discriminative dialogue model. By further factorisation of (1), we can derive the generative update formula:

$$\begin{aligned} b_t \propto \sum_{\mathbf{s}_{t-1},\mathbf{o}_t} & p(\mathbf{s}_t|a_{t-1},\mathbf{s}_{t-1}) p(\mathbf{o}_t|\mathbf{s}_t) \cdot \\ & \cdot b(\mathbf{s}_{t-1}|h_{t-1}) p(\mathbf{o}_t) \end{aligned} \quad (2)$$

where the transition probability $p(\mathbf{s}_t|a_{t-1},\mathbf{s}_{t-1})$ and the observation probability $p(\mathbf{o}_t|\mathbf{s}_t)$ represent the generative dialogue model.

In our approach, we define the dialogue state as a vector $\mathbf{s} = [s^1, \ldots, s^N]$ where s^i are values for slots in the dialogue domain, e.g. *to.desc* or *from.monument*. The observations are factored similarly to $\mathbf{o} = [o^1, \ldots, o^N]$, where o^i are individual slot-level observations, e. g. *inform(to.desc = downtown)* $\Leftrightarrow o^{to.desc} = downtown$. The probability of the slot-level observations $p(o^i)$ can be easily obtained by marginalising the observation probability $p(\mathbf{o})$. Because of limited space, only the processing of the *inform* dialogue acts is described in detail.

In the next two sections, we present the discriminative and generative models of belief update employed in the DSTC challenge by using the factorisation of the full belief state into independent factors to obtain computationally efficient updates.

2.1 Discriminative Belief Update

In this work, the belief state b_t is defined as a product of marginal probabilities of the individual slots, $b(\mathbf{s}_t) = \prod_i b(s^i_t)$, where s^i_t is the i-th slot at the turn t and the slot belief $b(s^i_t)$ is a probability distribution over all values for the slot i. To keep the notation uncluttered, the slot index, i, will be omitted in the following text. To further simplify the belief updates, similarly to the full belief monitoring represented by (1), the slot belief depends only on the previous slot belief b_{t-1}, the observation distribution $p(o_t)$, and the last system action a_{t-1}. This results in update rules for individual slots s as follows:

$$b(s_t) = \sum_{s_{t-1},o_t} p(s_t|a_{t-1},s_{t-1},o_t) b(s_{t-1}) p(o_t) \quad (3)$$

where the conditional probability distribution $p(s_t|a_{t-1},s_{t-1},o_t)$ represents the slot-level dialogue model.

There are two aspects which have to be taken into account when we consider the presented belief update: (1) the computational complexity and (2) the parameters of the dialogue model. First, the complexity of the belief update is given by the number of slot values and observations because the sum must be evaluated for all their combinations. This suggests that even this update may be computationally too expensive for slots where observations have a large number of values. Second, the slot-level dialogue model describes probabilistically how the value of a slot changes according to the context and the observations. Parameters of this conditional distribution would ideally be estimated from annotated data. Because of data sparsity, however, such estimates tend to be rather poor and either they must be smoothed or the parameters must be tied. To overcome this problem, we decided to set the parameters manually on the basis of two simple assumptions leading to very computationally efficient updates. First, we assume that our dialogue model should completely trust what the user says. Second, we assume that the user goal does not change when the user is silent. For example, if the user says: "I want to go downtown", $o^{to.desc}_t = downtown$, then the state should be $s^{to.desc}_t = downtown$; and when the user says nothing in the next turn, $o^{to.desc}_{t+1} = \odot$ (where the symbol $\odot$ is a special slot value representing that the user was silent), the state remains $s^{to.desc}_{t+1} = downtown$. This is captured by the following definition of the slot-level dialogue model:

$$p(s_t|a_{t-1},s_{t-1},o_t) = \begin{cases} 1 & (s_t = o_t \wedge o_t \neq \odot) \vee \\ & (s_t = s_{t-1} \wedge o_t = \odot) \\ 0 & otherwise \end{cases} \quad (4)$$

When (4) is substituted into (3), the belief update greatly simplifies and appears into the following form:

$$b(s_t) = \begin{cases} s_t = \odot : & p(s_{t-1} = \odot) p(o_t = \odot) \\ & \\ s_t \neq \odot : & p(o_t = s_t) \\ & + p(o_t = \odot) p(s_{t-1} = s_t) \end{cases} \quad (5)$$

Note that this model effectively accumulates probability from multiple hypotheses and from multiple turns. For example, its ability to "remember" the belief from the previous turn is proportional to the probability mass assigned to the SLU hypothesis that the user was silent about the slot in question. In the special case when the user is silent with probability 1.0, the current belief is equal to the previous belief.

This belief update is very computationally efficient. First, instead of summing over all combinations of the slot and observation values (3), the belief can be computed by means of a simple formula (5). Second, if the user does not mention a particular slot value during the dialogue, this value will always have a probability of zero. Therefore, only the probability for values suggested by the SLU component has to be maintained.

2.2 Generative model for belief update

Similarly to the discriminative belief update, the generative model relies on factorisation of the full belief state into a product of marginal slot beliefs and a simple dependency structure where a slot belief depends only on the previous slot belief, the slot observation distribution $p(o_t^i)$, and the last system action a_{t-1}. The dialogue model $p(s_t|a_{t-1}, s_{t-1}, o_t)$ is further factored, however, into the transition model $p(s_t|a_{t-1}, s_{t-1})$ and the observation model $p(o_t|s_t)$ as given in (2).

The transition model describes the probability that the user will change his/her goal, given the previous goal and the last system action. For example, if the system asks the user about a specific slot, then it is reasonable to have a larger probability of this slot changing its value. As noted for the discriminative model, estimation of the dialogue model parameters requires a large amount of data, which was not available in the challenge. Therefore, we used parameter tying as described by Thomson and Young (2010), and set the tied parameters manually:

$$p(s_t|a_{t-1}, s_{t-1}) = \begin{cases} \theta_t & \text{if } s_t = s_{t-1} \\ \frac{1-\theta_t}{|values|-1} & \text{otherwise} \end{cases} \qquad (6)$$

where θ_t describes the probability of a slot value staying the same and $|values|$ denotes the number of values for the slot. In other words, the probability θ_t sets a tradeoff between the system's ability to remember everything that was said in the past and accepting new information from the user. If θ_t is too high, the system will put a strong emphasis on the previous states and will largely ignore what the user is saying. When testing different values of θ_t on heldout data, we observed that if they are selected reasonably, the overall performance of the system does not change much. Therefore, the θ_t value was fixed at 0.8 for all slots and all datasets.

The observation model $p(o_t|s_t)$ describes the dependency between the observed values and the slot values. Similarly to the transition model, parameters of the observation probability distribution were tied and set manually:

$$p(o_t|s_t) = \begin{cases} \theta_o & \text{if } o_t = s_t \\ \frac{1-\theta_o}{|values|-1} & \text{otherwise.} \end{cases} \qquad (7)$$

where θ_o defines the probability of the agreement between the observation and the slot value. The probability of agreement describes how the model is robust to noise and systematic errors in SLU. When θ_o is set high, the model assumes that the SLU component makes perfect predictions, and therefore the SLU output must agree with the slot values. Based on manual tuning on held-out data, θ_o was set to 0.8.

Inference in the presented model is performed with Loopy Belief Propagation (LBP) (Pearl, 1988). LBP is an approximate message passing inference algorithm for Bayesian networks (BN). LBP can be computationally intensive if there are nodes with many parents in the network. Therefore, as previously described, our model uses a simple dependency structure where slots depend only on the same slot from the previous turn, and slot-level observations depend on the corresponding slot from the same turn. To make the inference even more efficient, one can take advantage of the tied observation and transition probabilities. We group all unobserved values in the nodes of BN together and maintain only a probability for the group as a whole, as suggested by Thomson and Young (2010).

3 Evaluation

The discriminative (DT) and generative dialogue (GT) trackers described in Sections 2.1 and 2.2 were evaluated on the DSTC data.

The input of DT and GT were the SLU n-best lists either with original probabilities or the scores mapped into the probability space. The trackers were evaluated on both live and batch data. The metrics were computed with Schedule 1 (see Williams et al. (2013)). In addition, we include into the evaluation the DSTC baseline tracker. The results on the live and batch data are shown in Table 1 in the Appendix. Please note that the results for GT differ from the results submitted for DSTC. Only after the submission deadline, did we find that some of the parameters in the transition model were set incorrectly. After the setting was fixed, the results improved.

The results show that the DT consistently outperforms the baseline tracker and the DT achieves comparable or better results than the GT. The DT clearly provides better estimates of the dialogue states because of the incorporation of the context and the processing of multiple hypotheses. To assess the statistical significance of the accuracy metric, 95% confidence scores for all measurements were computed. Overall, the confidence intervals were between 0.1% and 0.4% on the individual tests. On this basis, all differences larger than 1.0% can be considered statistically significant.

The GT outperforms the baseline tracker on all but the batch data. Manual inspection of the results revealed that the generative model is very sensitive to the probabilities assigned to the observations. For the batch data, presumably due to the score normalisation, the probabilities of hypotheses in the n-best lists were very similar to each other. As a result, the generative model had difficulties discriminating between the observed values.

In comparison with all trackers submitted for DSTC, the DT achieves second-best accuracy among the submitted trackers and the GT is among the average trackers. For more details see Table 2 in the Appendix, where the average scores were computed from the accuracy and the Brier score on test sets 1, 2, 3, and 4.

Regarding the Brier score, the results show that the DT outperforms the baseline tracker and estimates the belief state as well as the best tracker in the DSTC. This can prove especially important when the tracker is used within a complete dialogue system where the policy decisions do not depend on the best dialogue state but on the belief state.

4 Discussion

The presented discriminative and generative models differ in two main areas: (1) how they incorporate observations into the belief state and (2) computational efficiency.

(1) Both the DT and GT models can accumulate information from multiple hypotheses and from multiple turns. The GT, however, tends to "forget" the dialogue history because the generative model indiscriminately distributes some of the probability mass from a slot value that was not recently mentioned to all other slot values each turn. This behaviour (see Table 3 for an example) is not easy to control because "forgetting" is a consequence of the model being able to represent the dynamics of a user changing his/her goal. The DT does not have this problem because the change in the goal is directly conditioned on the observations. If the user is silent, then the DT "copies" the past belief state and no probability in the belief state is distributed as described in (5).

(2) The DT tracker is significantly faster compared with the GT tracker while offering comparable or better performance. The slot level belief update in the discriminative model has a complexity of $O(n)$ whereas in the generative model it has a complexity of $O(n^2)$, where n is the number of values in the slot. When tested on a regular personal computer, the DT processed all four DSTC test sets, 4254 dialogues in total, in 2.5 minutes whereas the GT tracker needed 51 minutes. Therefore, the DT tracker is about 20 times more computationally efficient on the DSTC data. Although GT achieved performance allowing real-time use (it needed 0.1 seconds per turn) in the Let's Go domain, for more complex applications the GT could simply be too slow. In this case, the proposed discriminative tracker offers a very interesting alternative.

5 Conclusion

This paper described two dialogue state tracking models submitted for the DSTC challenge: (1) the discriminative tracker and (2) the generative tracker. The discriminative tracker is based on a conceptually very simple dialogue model with deterministic transition probability. Interestingly, this discriminative model gives performance comparable to the more complex generative tracker; yet it is significantly more computationally efficient. An extended description of this work can be found in the technical report (Žilka et al., 2013).

Acknowledgements

This research was partly funded by the Ministry of Education, Youth and Sports of the Czech Republic under the grant agreement LK11221 and core research funding of Charles University in Prague. The authors would like to thank Ondřej Dušek and Ondřej Plátek for useful comments.

A Comparison of the BT, DT, and GT trackers

live data	metric	BT	DT	GT
test1	accuracy	0.77	**0.88**	**0.88**
	Brier score	0.29	**0.21**	**0.21**
test2	accuracy	0.79	**0.89**	0.85
	Brier score	0.27	**0.20**	0.23
test3	accuracy	0.92	**0.94**	0.93
	Brier score	0.14	**0.11**	0.16
test4	accuracy	0.82	0.86	**0.87**
	Brier score	0.24	0.21	**0.20**
ALL	accuracy	0.83	**0.89**	0.88
	Brier score	0.24	**0.18**	0.20
batch data	metric	BT	DT	GT
test1	accuracy	0.75	**0.88**	0.74
	Brier score	0.35	**0.27**	0.39
test2	accuracy	0.79	**0.88**	0.77
	Brier score	0.30	**0.26**	0.33
ALL	accuracy	0.77	**0.88**	0.76
	Brier score	0.32	**0.27**	0.36

Table 1: Accuracy of the trackers on the live and batch test sets, where BT stands for the DSTC baseline tracker, DT denotes the discriminative tracker, and GT denotes the generative tracker. ALL denotes the average scores over the live and batch test sets.

B Comparison with the DSTC trackers

team/system	accuracy	Brier score
BT - C	0.81	0.27
BT	0.83	0.24
DT	0.89	**0.18**
GT	0.88	0.20
team1	0.88	0.23
team2	0.88	0.21
team4	0.81	0.28
team5	0.88	0.21
team6	**0.91**	**0.18**
team7	0.85	0.23
team8	0.83	0.24
team9	0.89	0.20

Table 2: Accuracy of the trackers submitted for the DSTC, where BT - C denotes the DSTC baseline tracker without removing the systematically erroneous SLU hypotheses, BT denotes the DSTC baseline tracker, DT denotes the discriminative tracker, GT denotes the generative tracker, and team* denote the best trackers submitted by other teams. The scores are averaged scores obtained on the four DSTC test sets.

C The problem of “forgetting” of the observed values in the GT tracker

#	P	SLU hyp.	slot value	GS	DS
1	1.0	centre	centre	0.8	1.0
	0.0	null	null	0.2	0.0
2	1.0	null	centre	0.68	1.0
			null	0.32	0.0
3	1.0	null	centre	0.608	1.0
			null	0.392	0.0

Table 3: Example of three turns in which the generative system “forgets” the observed value. # denotes the turn number, P denotes the probability of the observation, SLU hyp. denotes the observed hypothesis, GS denotes the belief of the generative system, and DS denotes the belief of the discriminative system.

References

Neville Mehta, Rakesh Gupta, Antoine Raux, Deepak Ramachandran, and Stefan Krawczyk. 2010. Probabilistic ontology trees for belief tracking in dialog systems. In *Proceedings of SigDial*, pages 37–46. Association for Computational Linguistics.

Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562–588.

Jason D. Williams, Antoine Raux, Deepak Ramachandran, and Alan W. Black. 2013. The Dialog State Tracking Challenge. In *Proceedings of SigDial*, Metz, France.

Jason D. Williams. 2007. Using particle filters to track dialogue state. In *IEEE Workshop on Automatic Speech Recognition & Understanding, 2007. ASRU*, pages 502–507. IEEE.

Jason D. Williams. 2008. Exploiting the ASR N-best by tracking multiple dialog state hypotheses. *Proc ICSLP, Brisbane*.

Steve Young, Milica Gašić, Simon Keizer, Francois Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The Hidden Information State Model: a practical framework for POMDP-based spoken dialogue management. *Computer Speech and Language*, 24(2):150–174.

Lukáš Žilka, David Marek, Matěj Korvas, and Filip Jurčíček. 2013. Bayesian Discriminative and Generative Models used in the 2012 Dialogue State Tracking Challenge. Technical report, Faculty of Mathematics and Physics, Charles University in Prague, July.