Hybrid Dialog State Tracker

Miroslav Vodolán[†‡]
Charles University in Prague[‡]
Faculty of Mathematics and Physics
Malostranske nam. 25, 11800, Prague

Rudolf Kadlec[†] **and Jan Kleindienst**[†]
IBM Watson[†]
V Parku 4
Prague 4, Czech Republic

{mvodolan, rudolf_kadlec, jankle}@cz.ibm.com
vodolan@ufal.mff.cuni.cz

Abstract

This paper presents a hybrid dialog state tracker that combines a rule based and a machine learning based approach to belief state tracking. Therefore, we call it a hybrid tracker. The machine learning in our tracker is realized by a Long Short Term Memory (LSTM) network. To our knowledge, our hybrid tracker sets a new state-of-the-art result for the Dialog State Tracking Challenge (DSTC) 2 dataset when the system uses only live SLU as its input.

1 Introduction

Spoken Dialogue Systems (SDSs) consist of many modules, one of which is a Dialog State Tracker (DST). DST is responsible for accumulating evidence throughout the dialogue and estimating current true user's goal. The user goal estimate is subsequently used by other modules of the SDS, e.g., by a policy module that picks the next best action.

Recently proposed DSTCs [1, 2, 3] provide a shared testbed with datasets and tools for evaluating of dialog state tracking methods. It abstracts away the subsystems of end-to-end spoken dialog systems, focusing only on the dialog state tracking. It does so by providing datasets of ASR and SLU outputs on slot-filling tasks with reference transcriptions, together with annotation on the level of dialog acts and user goals.

The last three dialog state tracking challenges [1, 2, 3] were dominated by machine learning based trackers [4, 5, 6]. However, when we consider the case where all trackers have the same Spoken Language Understanding (SLU) input, some rule based trackers [7, 8, 9, 10] achieved performance comparable to the top trackers. In this work, we aim to unite the best of the both worlds — high accuracy of the machine learning trackers and better interpretability of the rule based trackers. A similar research direction was recently explored in [11]. The core of our proposed tracker consists of several update rules that use a few parameters that are computed by a recurrent neural network. We show that on the DSTC2 dataset our hybrid tracker achieves the state-of-the-art performance among the systems that use the original live SLU. Note that the DSTs that also use Automatic Speech Recognition (ASR) output as additional feature achieve even better tracking accuracy. We will add these features in a future work.

For evaluation of the tracker we chose the DSTC2 because it contains complex dialogs with changes of the user's goal and it also provides a lot of training data. Dialogs in the DSTC1 did not have frequent user goal changes and the DSTC3 had only a limited training dataset. The challenges also differ in their domains. The DSTC1 dataset is collected from system providing bus routes, the DSTC2 is focused on restaurant domain and the DSTC3 combines restaurant and hotel domains.

In the next section we describe the architecture of our Hybrid tracker. Then we evaluate the tracker on the DSTC2 dataset and conclude the paper with an outline of our future work.
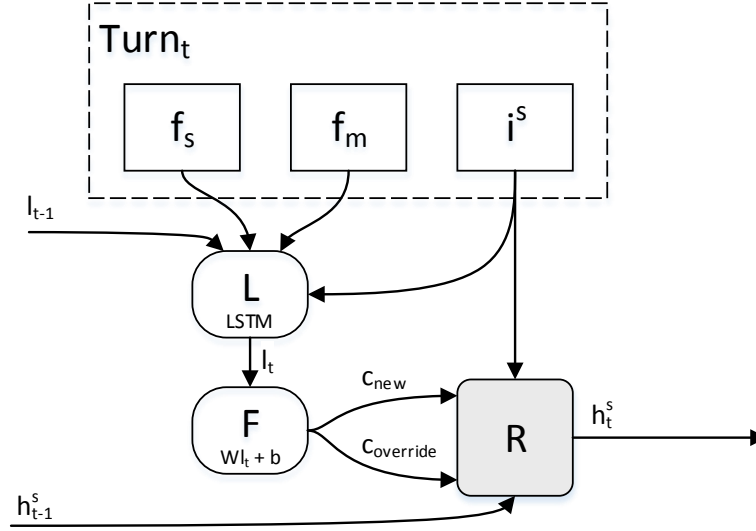
Figure 1: The structure of the Hybrid tracker for the turn t. It is a recurrent model which uses probability distribution h_{t-1}^s and l_{t-1} from the previous turn. Inputs of the machine learning part of the model (represented by functions L and F) are the features indicating the tracked slot f_s and the features f_m extracted from the machine actions. The features are used to produce values of parameters c_{new} and c_{override} for the R function.

2 Hybrid dialog state tracker model

In our previous work [10], we introduced a belief tracker based on a few simple rules which scored second in the joint slot accuracy in DSTC3 and its slightly modified version has the state-of-the-art accuracy on this dataset. Here we simplify the original rules for per slot tracking (in contrast with [10] where we tracked the slots jointly) and we add the machine learning component that provides parameters for these rules. We call the resulting architecture a hybrid tracker.

The tracker operates on a probability distribution over values for each slot separately. For each turn, the tracker generates these distributions reflecting the user's goals based on the last machine action, the observed user actions, the probability distributions in the previous turn and the hidden state l_{t-1} of a recurrent network L from the previous turn. The probability distribution h_t^s for a single slot s and turn t is represented by a vector indexed by possible values of the slot s. The joint belief state is represented by the probability distribution over Cartesian product for each slot.

In the following notation i^s denotes a user action pre-processed into a probability distribution of informed values for the slot s. During the pre-processing every *Affirm()* from SLU is transformed into *Inform(slot=value)* according to the machine action m. Further, we introduce a function corresponding to the simplified rules (fully described in Sec. 2.1) $h_t^s = R(h_{t-1}^s, i^s, c_{new}, c_{override})$, which is a function of a probability distribution in the previous turn, the pre-processed user action and two parameters which control how the new probability distribution h_t^s is computed. The next function $l_t = L(l_{t-1}, f_s, f_m, i^s)$ is recurrent and takes its own output l_{t-1} from the previous turn, the features f_s indicating the tracked slot, the features f_m representing machine actions and the pre-processed user action i^s of the turn t. The output of the recurrent network is then linearly transformed by $F(l_t)$ to parameters c_{new} and c_{override} for R. The structure of the tracker is shown in Figure 1.

In the next subsection, we will describe the rule based component of the Hybrid tracker. Afterwards, in Section 2.2, we will describe the machine learning part of the tracker.

2.1 Rule-based part

The rule-based part of the tracker represented by the function R consists of several simple update rules parametrized by parameters c_{new} and c_{override}[1]. Each of the parameters controls transition probability in a different way:

- c_{new} — controls how easy it would be to change the belief from hypothesis *None* to an instantiated slot value,

- c_{override} — models a goal change, that is, how easily it would be to override current belief with a new observation.

In this work we compute these parameters by a neural network. The rule based part of our tracker is specified by following equations. The first equation specifies belief update rule for probability assigned to slot's value v_1:

$$h_t^s[v_1] = h_{t-1}^s[v_1] - \tilde{h}_t^s[v_1] + i^s[v_1] \cdot \sum_{v_2 \neq v_1} h_{t-1}^s[v_2] \cdot a_{v_1 v_2} \tag{1}$$

Where $\tilde{h}_t^s[v_1]$ corresponds to amount of probability that will be transferred from $h_{t-1}^s[v_1]$ to other slot values in h_t^s:

$$\tilde{h}_t^s[v_1] = \sum_{v_2 \neq v_1} i^s[v_2] \cdot h_{t-1}^s[v_2] \cdot a_{v_2 v_1} \tag{2}$$

The $a_{v_1 v_2}$ is called transition coefficient between values v_1 and v_2. It controls amount of probability which is transferred from $h_{t-1}^s[v_2]$ to $h_t^s[v_1]$.

$$a_{v_1 v_2} = \begin{cases} v_1 = \textit{None} & c_{\text{new}} \\ v_1 \neq v_2 & c_{\text{override}} \end{cases} \tag{3}$$

As we can see, the R function is differentiable, therefore the machine learned part, described in the following subsection 2.2, can be trained by gradient descent methods together with the rule-based part.

We can find similar update equations in other rule-based trackers, e.g., [7, 12, 9, 10].

2.2 Machine learned part

The machine learning part of our tracker is realized by a LSTM [13] network. We use recurrent network for L since it can learn to output different values of c parameters for different parts of the dialog (e.g., it is more likely that new hypothesis will arise at the beginning of a dialog). This way, the recurrent network influences the rule-based component of the tracker. Since there are only two parameters that are used by the rule-based part, the tracker's decisions can be easily introspected.

The function L uses the feature f_s, which is one-hot representation of the tracked slot and the feature f_m which is a bag of words representation of machine actions. The last feature of the L function is pre-processed user action i^s representing marginal probabilities of informed values for slot s.

In our tracker we use one machine learned model that is shared for all slots. However, the model can distinguish between the slots according to f_s feature. The other systems use a different setup where a shared model is trained for all slots and then it is fine-tuned for each separate slot [14].

3 Evaluation

3.1 Method

The parameters of the hybrid tracker were trained by SGD with AdaGrad [15] and Adam [16] weight update rules. This is possible since of all parts of the model are differentiable (including the R function).

---

[1]These parameters were modelled by a so called *durability* function in our previous tracker [10].

We trained two groups of trackers with different settings. The first group was trained by the AdaGrad algorithm with the learning rate 0.5 and the gradient clipping with threshold 10. With this setting the training algorithm produced trackers heavily influenced by random initialization, which is good for later ensembling of the trackers. For the second group we used the Adam update rule with the learning rate 0.01, β_1 0.9 and β_2 0.999. These settings are much more invariant to random initialization therefore we randomly masked f_m features to get set of different trackers. Both groups used L function with 5 LSTM cells and $tanh$ as the activation function.

From each dialog in the *dstc2_train* data (1612 dialogs) we extracted training samples for the slots *food*, *pricerange* and *area* and used all of them to train each tracker. The training data was also used for selection of f_m features. We selected only those words from machine action[2] that appeared more than 5 times. This gives us the total number of 421 f_m features and 3 f_s features (one per *food*, *pricerange* and *area* slot).

The evaluated model was an ensemble of multiple trackers that were combined by averaging. Similar approach proved to be useful also in other RNN based trackers [14, 17]. For the ensemble, we used 100 trackers randomly selected from both tracker groups containing $115 + 143$ trackers.

We evaluated 10 different ensembles and selected the one with the best performance on validation *dstc2_dev* (506 dialogs) data, which is reported in subsection 3.2. Our tracker did not track the *name* slot because it hurts validation performance. Therefore, we always set value for the *name* slot *None*. The mean accuracy of the 10 ensembles on *dstc2_test* data (1117 dialogs) is 0.7448 with the standard deviation 0.0006.

The models were implemented using Theano [18] and Blocks [19].

	ASR	test2		post DSTC
		Acc.	L2	
Focus baseline		**.719**	**.464**	
HWU baseline		.711	.466	
DSTC2 stacking ensemble [2]	√	**.798**	**.308**	√
Williams [20]	√	**.784**	.735	
Henderson et al. [14]	√	.768	**.346**	
Yu et al. [21]	√	.762	.436	√
YARBUS [22]	√	.759	.358	√
Sun et al. [9]	√	.750	.416	
Hybrid Tracker – This work		**.745**	.433	√
Williams [20]		.739	.721	
Henderson et al. [14]		.737	**.406**	
Our previous tracker [10]		.737	.429	√
Sun et al. [9]		.735	.433	
Smith [23]		.729	.452	
Lee et al. [24]		.726	.427	
YARBUS [22]		.725	.440	√
Ren et al. [25]		.718	.437	

Table 1: Joint slot tracking results for various systems reported in the literature. The trackers that used ASR have √ in the corresponding column. The results of systems that did not participate in DSTC2 are marked by √ in the "post DSTC" column. The first group shows two baselines provided in the DSTC2. The second group shows results of an ensemble of all trackers submitted to the challenge. This system achieves the best result among the systems that use both the original SLU and ASR. The third group lists individual trackers that use ASR. The fourth group lists systems that use only the live SLU provided in the original dataset. Our hybrid tracker sets new state-of-the-art result in this category. The best results for a given metric and tracker group are in bold.

---

[2]The machine action is represented by dialog acts.

3.2 Results

Table 1 shows the results of our hybrid tracker and other top performing trackers known from the literature. In the category of trackers that use only the live SLU features our systems sets the new state-of-the-art with accuracy 0.745 on *dstc2_test*. The accuracy of the tracker on *dstc2_dev* is 0.657 and 0.767 on *dstc2_train*.

3.3 Discussion

Evaluation on the DSTC2 dataset shows that our hybrid system that extends rule based tracking core with the machine learning component outperforms the previous best tracker [20] that used the same SLU input. This result is also interesting since our ML component is relatively lightweight (it has only approx. 10k parameters, the hidden state consist of only 5 neurons) and it influences computation of the rules part by only 2 parameters.

4 Future Work and Conclusion

We have presented a belief tracker that combines our previous tracker with machine learning techniques. It performs better than our previous tracker while still being highly interpretable in comparison with pure neural network approaches.

However, trackers that use ASR as their input achieve even better accuracy. Therefore the next step will be to add ASR features to our machine learning component. This will hopefully allow us to further improve accuracy of our system.

Acknowledgments

References

[1] J. Williams, A. Raux, D. Ramachandran, and A. Black, "The Dialog State Tracking Challenge," in *Proceedings of the SIGDIAL 2013 Conference*, (Metz, France), pp. 404–413, Association for Computational Linguistics, August 2013.

[2] M. Henderson, B. Thomson, and J. D. Williams, "The second dialog state tracking challenge," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 263–272, Association for Computational Linguistics, June 2014.

[3] M. Henderson, , B. Thomson, and J. Williams, "The Third Dialog State Tracking Challenge," in *Spoken Language Technology Workshop, 2014. IEEE*, 2014.

[4] S. Lee and M. Eskenazi, "Recipe For Building Robust Spoken Dialog State Trackers : Dialog State Tracking Challenge System Description," *Sigdial*, no. August, pp. 414–422, 2013.

[5] J. D. Williams, "Web-style ranking and SLU combination for dialog state tracking," no. June, pp. 282–291, 2014.

[6] M. Henderson, B. Thomson, and S. Young, "Robust Dialog State Tracking Using Delexicalised Recurrent Neural Networks and Unsupervised Adaptation," *SLT*, 2014.

[7] Z. Wang and O. Lemon, "A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information," in *Proceedings of the SIGDIAL 2013 Conference*, (Metz, France), pp. 423–432, Association for Computational Linguistics, August 2013.

[8] R. Kadlec, J. Libovický, J. Macek, and J. Kleindienst, "IBM's Belief Tracker: Results On Dialog State Tracking Challenge Datasets," in *Dialog in Motion workshop on EACL 2014*, pp. 10–18, 2014.

[9] K. Sun, L. Chen, S. Zhu, and K. Yu, "The sjtu system for dialog state tracking challenge 2," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 318–326, Association for Computational Linguistics, June 2014.

[10] R. Kadlec, M. Vodolan, J. Libovicky, J. Macek, and J. Kleindienst, "Knowledge-based dialog state tracking," in *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pp. 348–353, IEEE, 2014.

[11] K. Sun, Q. Xie, and K. Yu, "Recurrent Polynomial Network for Dialogue State Tracking," *Dialog and Discourse*, pp. 1–22, 2015.

[12] L. Žilka, D. Marek, M. Korvas, and F. Jurčíček, "Comparison of bayesian discriminative and generative models for dialogue state tracking," in *Proceedings of the SIGDIAL 2013 Conference*, (Metz, France), pp. 452–456, Association for Computational Linguistics, August 2013.

[13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[14] M. Henderson, B. Thomson, and S. Young, "Word-based dialog state tracking with recurrent neural networks," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 292–299, Association for Computational Linguistics, June 2014.

[15] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[16] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] L. Zilka and F. Jurcicek, "Incremental LSTM-based Dialog State Tracker," 2015.

[18] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements." Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[19] B. van Merrienboer, D. Bahdanau, V. Dumoulin, D. Serdyuk, D. Warde-farley, J. Chorowski, and Y. Bengio, "Blocks and Fuel : Frameworks for deep learning," pp. 1–5, 2015.

[20] J. D. Williams, "Web-style ranking and slu combination for dialog state tracking," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 282–291, Association for Computational Linguistics, June 2014.

[21] K. Yu, K. Sun, L. Chen, and S. Zhu, "Efficient Dialogue State Tracking," vol. 23, no. 12, pp. 2177–2188, 2015.

[22] J. Fix and H. Frezza-buet, "YARBUS : Yet Another Rule Based belief Update System," 2015.

[23] R. Smith, "Comparative error analysis of dialog state tracking," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 300–309, Association for Computational Linguistics, June 2014.

[24] B.-J. Lee, W. Lim, D. Kim, and K.-E. Kim, "Optimizing generative dialog state tracker via cascading gradient descent," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 273–281, Association for Computational Linguistics, June 2014.

[25] H. Ren, W. Xu, and Y. Yan, "Markovian discriminative modeling for dialog state tracking," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, (Philadelphia, PA, U.S.A.), pp. 327–331, Association for Computational Linguistics, June 2014.