Memory State Tracker: A Memory Network based Dialogue State Tracker

Di Wang and Simon O'Keefe
Department of Computer Science, University of York, Heslington, York, U.K.

Keywords: Dialogue State Tracker, Memory Network.

Abstract: Dialogue State Tracking (DST) is a core component towards task oriented dialogue system. It fills manually-set slots at each turn of an utterance, which indicate the current topics or user requirement. In this work we propose a memory based state tracker that includes a memory encoder which encodes the dialogue history into a memory vector, and then connects to a pointer network which makes predictions. Our model reached a joint goal accuracy of 49.16% on MultiWOZ 2.0 data set (Budzianowski et al., 2018) and 47.27% on MultiWOZ 2.1 data set (Eric et al., 2019), outperforming the benchmark result.

1 INTRODUCTION

Task oriented dialogue systems, such as Apple Siri or Amazon Alexa, address one of the major tasks in the NLP field. While the complete accomplishment of a dialogue system may still have a long way to go, a step-by-step approach that includes Dialogue Representation, State Tracking and Text Generation is been proposed. The main component of the middle step is State Tracking, which predicts at each turn of the dialogue what the topics or user requirements are. The state is represented as values in certain predefined slots. For example, given one of the sentences of a dialogue: *Is there any restaurant in the city center?* then the corresponding state values could be *[Task: Restaurant][Location: Center]*.

Various task oriented dialogue data sets have been released, including NegoChat (Rosenfeld et al., 2014) and Car assistance (Eric and Manning, 2017). Eric et al. introduced the MultiWOZ 2.1 data set (Eric et al., 2019), which is a multi-domain data set in which the conversation domain may switch over time. The user may start a conversation by asking to reserve a restaurant table, then go on to request a taxi ride to that restaurant. In this case, the state tracker has to determine the corresponding domain, slots and values at each turn of the dialogue, taking into account the history of the conversation if necessary.

In this paper, we introduce a memory based dialogue state tracker, which consists of two components: a memory encoder which encodes the dialogue history into a memory vector, and a pointer network which points to the set of possible values of states including words in the dialogue history and ontology of the data set. The memory vector will be updated at each turn of the dialogue, and it will then be passed to the pointer network, where prediction is made based on the current memory. For each dialogue there are multiple slots to be filled, each with a set of possible value. Both the slots and their possible values are predefined. The prediction is a two step procedure. Firstly, predict the domains the current utterance lies in, then for the specific domain, fill values into corresponding slots.

2 RELATED WORK

2.1 Dialogue State Tracking

Dialogue state tracking (DST), or Belief tracking, was introduced as an intermediate step in dialogue systems. In this step the task is to recognize user's goal as *state*, which will then be used to guide the system response correspondingly(Bohus and Rudnicky, 2006). Table 1 shows an example of a dialogue with the corresponding dialogue states at each turn of the dialogue, which will be updated at every user utterance.

DST systems can be classified into two types: ontology-based and ontology-free. Ontology-based DST (Ramadan et al., 2018; Zhong et al., 2018) requires a set of pre-defined possible values for the dia-

Wang, D. and O'Keefe, S.
Memory State Tracker: A Memory Network based Dialogue State Tracker.
DOI: 10.5220/0010385705330538
In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART 2021) - Volume 1*, pages 533-538
ISBN: 978-989-758-484-8

Table 1: An example of dialogue states at each turn of the dialogue.

utterance	dialogue state
User:I would like to find a cheap restaurant that serves tuscan food **System**: nothing is matching your request. I'm sorry.	(restaurant, food, tuscan) (restaurant, price range, cheap) (restaurant, name, not mentioned) (restaurant, area, not mentioned)
User:Could you help me find some cheap Italian food then? **System**: If you do not have a preference of area, I recommend La Margherita in the west.	(restaurant, food, italian) (restaurant, price range, cheap) (restaurant, name, not mentioned) (restaurant, area, not mentioned)

logue state, and the model selects the most likely one from the ontology set. The Ontology-free DST (Xu and Hu, 2018; Gao et al., 2019), on the other hand, does not require such an ontology set, and will choose the most likely phrase from the dialogue history and vocabulary set.

The ontology-based DST will require a manually defined set of values for all slots, which will be expensive for large scale dialogue systems and makes it difficult to generalise the model. On the other hand, the ontology-free DST is easy to generalise for large scale dialogue systems, but can not utilize expert knowledge in the model.

The two types of DST can be incorporated by using a gate function which determines if the current slot should be filled by choosing from an ontology or from the dialogue history (Qiu et al., 2019).

2.2 Multi Domain Dialogue State Tracking

The MultiWOZ 2.1 data set released by (Eric et al., 2019) is one of the largest task oriented dialogue system data sets so far. It includes 9 different domains, and at each turn of a dialogue there can be more than one domain active. As the dialogue continues the active domain and corresponding slot may be changed. Benchmark methods of dialogue state tracking on this data set include FJST, HJST (Eric et al., 2019), referring to Flat Joint State Tracker and Hierarchical Joint State Tracker respectively. FJST encodes the dialogue history into a vector and predicts the state. HJST is the same but uses a hierarchical encoder. TRADE (Wu et al., 2019) uses a slot generator to generate slot values from a dialogue history and vocabulary.

2.3 Memory Network

A memory network was firstly proposed by (Weston et al., 2014) for the Question Answering task. The core of a memory network is a memory vector which will be updated with each new input. MemN2N (Sukhbaatar et al., 2015) was the first end to end memory network. TheDynamic Memory Network (Kumar et al., 2016) proposes a method to update the memory vector repeatedly with different attention controlled by previous memory.

In this work, we adopt the idea of memory vectors, e_i, which will be updated by the dialogue history and the previously predicted dialogue states.

3 MODEL

In this section, we provide a detailed description of the proposed Memory State Tracking model, as shown in figure 1.

Each sentence in the input dialogue is first encoded to vector representation by sentence encoder. Then each sentence vector is fed into a RNN structured Memory network in turn, and the memory network will output a memory vector at each turn of the dialogue. The final step is to make prediction based on the memory vector, which is a three step procedure for each state slot to be filled: firstly, through a binary 'mention' gate to determine if the current state is mentioned or not in the dialogue. If the gate predicts it is not mentioned, then the state slot will be marked as "not mentioned". If it is mentioned, two predictors will make independent predictions of possible slot values in the ontology and dialogue history respectively, and another gate function will be used to decide which prediction will be used as the final predicted value of the slot.

3.1 Encoder

We used two types of sentence encoding model as our model's encoder, the first one used Glove word embedding(Pennington et al., 2014) and feed each word into a GRU and used the last hidden state as the sentence representation. The second one used BERT model for language understanding (Devlin et al., 2018) as the sentence encoder. For each turn of the di-

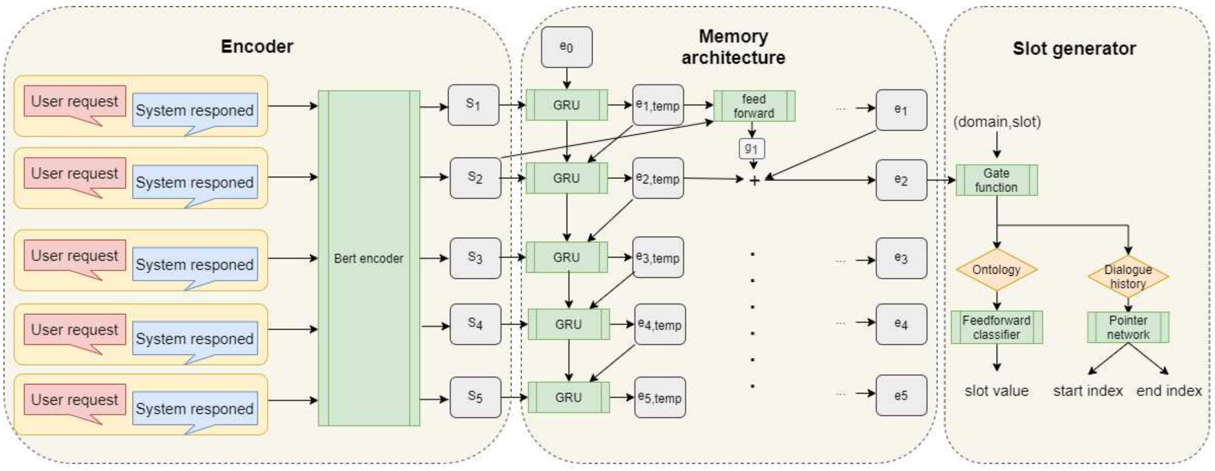

Figure 1: Memory state tracking system. In this graph the Memory is working on the second turn of the dialogue as example, and the Slot predictor is predicting for a specific (domain, slot) pair.

alogue, the input sequence is the current user request and system response separated by [SEP] token, and then padded to a fixed length and fed into the encoder, the output s_i is the vector representation of sentence i.

3.2 Memory Network

For each dialogue, a memory vector e_i is used to represent all information from the dialogue history. At each turn of the dialogue, the memory vector is updated using an RNN structured network, where the input is the sentence representation s_i of the utterance at current turn. At the beginning of the training process, a hidden parameter H_0 is initialized with all values set to zero, then for each turn of the dialogue, the memory is updated with

$$e_{temp},\ H_i = GRU(s_i, H_{i-1}) \tag{1}$$

$$e_i = g_i \cdot e_{temp} + (1 - g_i) \cdot e_{i-1} \tag{2}$$

where the e_i is the ith memory representation of the dialogue, representing the whole dialogue until the ith turn.

g_i is an attention score computed by a simple gate function with two layer feed-forward neural networks

$$g_i = sigmoid(W_2 \cdot tanh(W_1 \cdot tanh(s_i,\ e_{i-1}))) \tag{3}$$

The purpose of this gate function is to measure how important the current turn of the dialogue is. If the gate function gives a high score it means the current turn is informative, and by equation 2 it will update the memory vector greatly, and vice versa.

3.3 Pointer Predictor

We adopt the Pointer network architecture (Merity et al., 2016) as the predictor in our model. The prediction model makes predictions based only on the memory vector e_i at each turn. The prediction is a two step procedure. First we have a set of mention gate functions for each of the slot to be filled:

$$m_{i,j} = G_{j \in J}(e_i) \tag{4}$$

which decide if the jth (domain, slot) pair is mentioned or not at ith. J is the set of all (domain, slot) pairs to be filled, $m_{i,j}$ is the probability of the mention gate, and $G_{j \in J}$ is a fully connected layer with sigmoid activation.

$$G_{j \in J}(e_i) = sigmoid(W_j e_i + b_j) \tag{5}$$

Different from the three way gate in TRADE (Wu et al., 2019), we do not add “don’t care” in this gate, as the number of “don’t care” slots is relatively small compared with “not mentioned”, and it can be predicted later in the categorical predictor.

If the mention gate predicts the jth (domain, slot) pair is not mentioned, then the value of this slot will be filled with “not mentioned”. If the gate predicts that it is mentioned, the value will be filled by the pointer network.

For (domain, slot) pairs that are predicted to be “mentioned”, there are two independent predictors to predict its values. One is a pointer network point to words in the dialogue history

$$Index_{start,j} = argmax_k(sigmoid(W_{j,1} w_k + b_{j,1})) \tag{6}$$

$$Index_{end,j} = argmax_k(sigmoid(W_{j,2} w_k + b_{j,2})) \tag{7}$$

where w_k is the word embedding of the kth word in the dialogue history. $Index_{start,j}$ represents the start index of predicted slot values and $Index_{end,j}$ represents the end index, with words in between being the final prediction of pointer network.

Another predictor is a categorical classifier that chooses a word from possible values for current (domain, slot) pairs, with a feed-forward neural network.

A gate function is employed to determine which predictor is used, with a structure similar to the mention gate.

$$G_{pred}(e_i) = sigmoid(W_j e_i + b_j) \tag{8}$$

4 EXPERIMENT

This section discusses the experiment of our model trained on the MultiWOZ 2.0 (Budzianowski et al., 2018) and MultiWOZ2.1 (Eric et al., 2019) data sets, including experiment setups, parameters and results.

4.1 Data Set

MultiWOZ 2.0 and 2.1 are large goal oriented dialogue system data sets, which consist of around 10,000 dialogues, 113, 556 turns of dialogues with multiple domains including *restaurant*, *hotel*, *hospital*, *taxi*, *police train*, *attraction*, and *bus*. Table 1 is an example of a dialogue about restaurant booking.

MultiWOZ 2.1 is a refined version of MultiWOZ 2.0, modifing around 2% of the slots.

4.2 Experiment Setup

In our work the hospital and police domains are ignored as they contain only very few of the dialogues, following (Wu et al., 2019). We use the validation and test set supplied by the data set.

4.3 Model Details and Parameters

Our model is trained on a single GTX 2080 GPU with Pytorch environment.

Word embedding dimension is set to 300 using Glove embedding (Pennington et al., 2014). The maximum sentence length is set to 30, in order to train the model in batches. This means words in an input sentence after the 30th word will be discarded, if the sentence is longer than 30, and if shorter, a special token of **[PAD]** will be added until the length of input sentence is 30. Similarly, the max dialogue length is set to be 10.

The training batch size is 128, hidden size of encoder and all feed-forward neural networks is 256. We also used Gradient Clipping (Kanai et al., 2017) with clip parameter 50, to prevent gradient explosion.

We used different learning rates for encoder and pointer networks, which shows to have better performance. The encoder learning rate is 0.001 and predictor learning rate is 0.0001.

4.4 Experimental Results

We used joint goal accuracy to test our model. The slot accuracy is the accuracy of each single state value. For the joint accuracy, only if all the slots in one turn are correctly predicted will the prediction be marked as correct, otherwise it is incorrect.

Table 2 shows the joint goal accuracy of our model on MultiWOZ2.0 and MultiWOZ2.1 data set, compared with benchmark models. Our model beats these baseline models in both data sets.

Table 2: Joint goal accuracy on MultiWOZ 2.0 and MultiWOZ 2.1 data set.

	MWOZ 2.0	MWOZ 2.1
HJST(Eric et al., 2019)	38.4	35.55
FJST(Eric et al., 2019)	40.2	38.0
TRADE(Wu et al., 2019)	48.6	45.6
DST(Gao et al., 2019)	39.41	36.4
HyST(Goel et al., 2019)	42.33	38.10
MST(ours)	**49.16**	**47.27**

Table 3 shows the accuracy of each domain for MultiWOZ 2.1 data set.

Table 3: Domain-Specific Accuracy on MultiWOZ 2.1 data set.

Domain	Joint Accuracy	Slot Accuracy
Restaurant	66.41	98.22
hotel	48.13	97.14
Taxi	39.50	94.85
Attraction	66.47	98.43
Train	63.83	94.98

4.5 Ablation Test

Table 4 shows the ablation test on the MultiWOZ 2.1 data set. We tested the performance with different encoders, and with and without the memory mechanism. The test shows that the choice of the encoder does not make much difference, but the memory mechanism does improve the model performance significantly.

Table 4: Ablation test on MultiWOZ 2.1 data set.

Feature	Joint accuracy
GRU encoder(no memory)	46.55
Bert encoder(no memory)	46.62
Bert+Memory mechanism	47.27

5 CONCLUSION AND DISCUSSION

In this paper we introduced a novel memory mechanism for a dialogue state tracking system. The core contribution of our work is to incorporate the memory architecture into the dialogue state tracking system. We used a vector which will be updated at each turn of the dialogue, so it will preserve useful historical information in the model. This model outperforms a set of benchmark models with joint goal accuracy on both MultiWOZ 2.0 and MultiWOZ 2.1 data set.

In the domain-specific accuracy table, we can see that the Hotel and Taxi domains are shown to be more difficult compared with other domains. The Hotel domain has 11 slots to be filled which is the largest of all domains, so it is reasonable that the Hotel domain has a lower joint goal accuracy. For the Taxi domain, as shown in the Appendix, the number possible values for its state slot is the highest among all domains, which may lead to the low joint goal accuracy.

REFERENCES

Bohus, D. and Rudnicky, A. (2006). A “k hypotheses+ other” belief updating model.

Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., and Gašić, M. (2018). Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Eric, M., Goel, R., Paul, S., Sethi, A., Agarwal, S., Gao, S., and Hakkani-Tur, D. (2019). Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Eric, M. and Manning, C. D. (2017). Key-value retrieval networks for task-oriented dialogue. *arXiv preprint arXiv:1705.05414*.

Gao, S., Sethi, A., Agarwal, S., Chung, T., and Hakkani-Tur, D. (2019). Dialog state tracking: A neural reading comprehension approach. *arXiv preprint arXiv:1908.01946*.

Goel, R., Paul, S., and Hakkani-Tür, D. (2019). Hyst: A hybrid approach for flexible and accurate dialogue state tracking. *arXiv preprint arXiv:1907.00883*.

Kanai, S., Fujiwara, Y., and Iwamura, S. (2017). Preventing gradient explosions in gated recurrent units. In *Advances in neural information processing systems*, pages 435–444.

Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R., and Socher, R. (2016). Ask me anything: Dynamic memory networks for natural language processing. In *International conference on machine learning*, pages 1378–1387.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Qiu, L., Xiao, Y., Qu, Y., Zhou, H., Li, L., Zhang, W., and Yu, Y. (2019). Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150.

Ramadan, O., Budzianowski, P., and Gašić, M. (2018). Large-scale multi-domain belief tracking with knowledge sharing. *arXiv preprint arXiv:1807.06517*.

Rosenfeld, A., Zuckerman, I., Segal-Halevi, E., Drein, O., and Kraus, S. (2014). Negochat: a chat-based negotiation agent. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 525–532.

Sukhbaatar, S., Weston, J., Fergus, R., et al. (2015). End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Weston, J., Chopra, S., and Bordes, A. (2014). Memory networks. *arXiv preprint arXiv:1410.3916*.

Wu, C.-S., Madotto, A., Hosseini-Asl, E., Xiong, C., Socher, R., and Fung, P. (2019). Transferable multi-domain state generator for task-oriented dialogue systems. *arXiv preprint arXiv:1905.08743*.

Xu, P. and Hu, Q. (2018). An end-to-end approach for handling unknown slot values in dialogue state tracking. *arXiv preprint arXiv:1805.01555*.

Zhong, V., Xiong, C., and Socher, R. (2018). Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467.

APPENDIX

Table 5: Number of possible values for each (domain, slot) pairs, for MultiWOZ 2.0 and MultiWOZ 2.1 data set.

Slot Name	MultiWOZ2.0	MultiWOZ2.1
taxi-leaveAt	119	108
taxi-destination	277	252
taxi-departure	261	254
taxi-arriveBy	101	97
restaurant-people	9	9
restaurant-day	10	10
restaurant-time	61	72
restaurant-food	104	109
restaurant-pricerange	11	5
restaurant-name	183	190
restaurant-area	19	7
hotel-people	11	8
hotel-day	11	13
hotel-stay	10	10
hotel-name	89	89
hotel-area	24	7
hotel-parking	8	4
hotel-pricerange	9	8
hotel-stars	13	9
hotel-internet	8	4
hotel-type	18	5
attraction-type	37	33
attraction-name	137	164
attraction-area	16	7
train-people	14	12
train-leaveAt	134	203
train-destination	29	27
train-day	11	8
train-arriveBy	107	157
train-departure	35	31