


# Multi-Domain Dialogue State Tracking with Hierarchical Task Graph

Tianhao Shen, Xiaojie Wang 

*School of Computer Science*

*Beijing University of Posts and Telecommunications*

Beijing, China

{shentianhao, xjwang}@bupt.edu.cn

**Abstract**—Multi-domain dialogue state tracking (DST), which tracks user goals and intentions across multiple domains, is a core task for multi-domain task-oriented dialogue system. Previous works in multi-domain DST focus on the open-vocabulary setting to alleviate the over-dependence on pre-defined ontology. However, they come up short of modeling the relationships among domains and slots in an explicit and efficient way. In this paper, we propose a multi-domain dialogue state tracker with hierarchical task graph (DST-HTG) to address the above issues. DST-HTG uses a copy mechanism to perform DST under the open-vocabulary setting, which makes our model eliminate the dependence on pre-defined full ontology. Moreover, we extend our DST model with a hierarchical task graph which has simple structure and rich semantic information to incorporate the relationships among domains and slots into DST process explicitly and efficiently. Empirical results show that DST-HTG achieves the state-of-the-art joint goal accuracy and slot accuracy in MultiWOZ 2.0, a recently proposed multi-domain task-oriented dialogue dataset, which indicates the effectiveness of our proposed model.

**Index Terms**—natural language processing, task-oriented dialogue systems, multi-domain dialogue state tracking, task graph


## I. INTRODUCTION

Multi-domain task-oriented dialogue systems handle user requests from multiple domains in a single dialogue, such as booking a hotel firstly and then finding a restaurant nearby. Multi-domain dialogue state tracking (DST) is a key task in multi-domain task-oriented dialogue systems. It aims at tracking user goals and intentions across multiple domains in each turn of the dialogue and output the dialogue state, i.e., a set of (domain, slot, value) triplets to summarize the entire dialogue until the current turn. Because the subsequent dialogue management and response generation rely on the dialogue state, maintaining an accurate dialogue state is important for the whole dialogue system.

Traditional Neural DST methods [1]–[4] assume that the ontology, i.e. all slots and their values, is defined and known in advance, which can simplify DST into a classification problem [5]. But in the multi-domain setting, there are some issues when applying these approaches. 1) The ontology of multi-domain task-oriented dialogue is usually quite large. For example, in the recent proposed multi-domain task-oriented dialogue dataset MultiWOZ [6], there are 30 (domain, slot)

pairs and more than 4500 values, which makes it very difficult for classification. 2) In practical perspective, some slots have a dynamically changing or even infinite numbers of possible values, which makes the ontology hard to be pre-defined or enumerated. For example, the value set of a hotel name slot may change over time, and a train departure time slot has an infinite number of values. To address these issues, there are some previous works that can directly copy value for every mentioned slot from dialogue context [7], [8], making these approaches support open-vocabulary setting, which assumes that the value set of (domain, slot) pair is unavailable, instead of the aforementioned pre-defined ontology setting.

However, there is another unique yet less studied issue in multi-domain DST, which is also an important difference between multi-domain DST and single-domain DST. In practice, a multi-domain dialogue is not the simple concatenation of multiple single-domain dialogues. Specifically, two slots from different domains may have relationship when they co-occur in the same dialogue. In this case, the user may provide slot value in an implicit approach. For example, as shown in Fig. 1, the (hotel, area, centre) triplet can be inferred from the aforementioned (restaurant, area, centre) triplet, even though the user did not mention the explicit value of area slot in the hotel domain. There are only few previous works focus on this issue. TRADE [5] follows an encoder-decoder architecture with soft-gated copy mechanism [9], [10], which takes dialogue history as the encoder input and generate slot value from the decoder output. But it models relationships among domains and slots implicitly, instead of using an explicit approach. Recently proposed DSTQA [11] treats multi-domain DST as a document-based question answering task and explicitly construct the connection among (domain, slot) pairs. But the number of (domain, slot) pairs will increase rapidly with the number of domains and slots, which makes it inefficient in representing relationships among domains and slots when the dialogue dataset has a large number of domains and slots. Moreover, in order to determine which domains and slots are mentioned in current user turn, these models choose to iterate over all (domain, slot) pairs in every turn, and then use a three-way classifier to determine if the value of each (domain, slot) pair is none, don't care or mentioned in the dialogue context. For example, in MultiWOZ dataset, there are 30 (domain, slot) pairs. So, TRADE will generate 30 different

 Corresponding Author

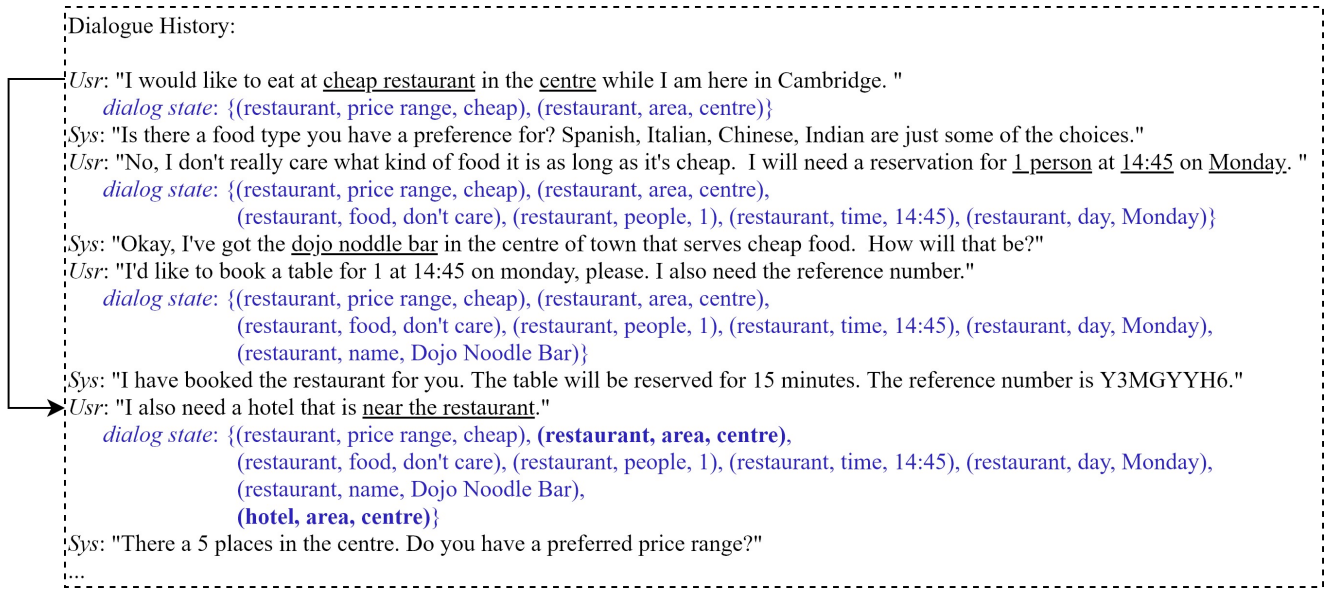


Fig. 1. An example of multi-domain dialogue state tracking (DST). The solid arrow and the bold text in dialogue state indicate the relationship between two different (domain, slot) pairs: (restaurant, area) and (hotel, area).

slot values for every single turn, and DSTQA will generate 30 different questions instead, which leads to inefficient and time-consuming DST process. We expect DST models can model the relationships among domains and slots in an explicit and efficient way.

In this paper, we propose a multi-domain dialogue state tracker with hierarchical task graph (DST-HTG) for multi-domain DST. Based on the open-vocabulary setting, our model applies a copy mechanism to directly extract slot value from dialogue context, so pre-defined ontology is not needed. Besides, our model does not need to iterate over all (domain, slot) pairs. Instead, it generates the distribution over domains and slots using the dialogue representation, which makes our model more efficient. Moreover, we propose a graph called hierarchical task graph, in which flat (domain, slot) pairs are separated into a hierarchical structure including domain nodes and slot nodes, and add edges between these nodes to represent the relationships among (domain, slot) pairs. This graph structure has two advantages. 1) The domain nodes and slot nodes can be effectively reused to reduce the complexity of graph. For example, the area slot in hotel, attraction, and restaurant domain can share the same "area" slot node, which also means this graph naturally contains the relationships among pairs like (hotel, area), (attraction, area) and (restaurant, area). 2) the graph is still capable of containing rich relationship information with less nodes and edges. To model the relationships among domains and slots as complete as possible, we construct three kinds of edges among these nodes, which will be described in detail in the following sections. As we will show later, incorporating hierarchical task graph helps improve the final DST accuracy.

In summary, our contributions are as follows: 1) we propose a novel multi-domain DST model that can improve DST accu-

racy in open-vocabulary scenarios. 2) we propose a mechanism that can efficiently determine the domain and slot mentioned in current user turn without iterating over all (domain, slot) pairs. 3) we extend our DST model with a hierarchical task graph which can not only reduce the graph complexity but also retain rich relationship information among domains and slots.

## II. PROPOSED MODEL

In this section, we describe the details of the proposed model, DST-HTG. The model structure is illustrated in Fig. 2, which comprises seven components: dialogue history encoder (DHE), current user utterance encoder (CUE), bi-directional multi-head attention (BMA) module, domain-slot generator, task graph, slot gate, and span prediction decoder (SPD). Note that in this paper, dialogue history refers to the dialogue utterances in previous turns, and current user utterance refers to the user utterance in current turn.

1) DHE is a multi-layer transformer encoder [12] which takes as input the dialogue history, and outputs a dialogue history representation  $H_{dhe}$  which consists of a sequence of fixed-length vectors. Similar to DHE, CUE is a single-layer transformer encoder, which encodes the current user utterance and outputs a user utterance representation  $H_{cue}$ .

2) BMA module computes and fuses context-to-query attention and query-to-context attention to provide complementary information to each other. In the proposed model, there are two BMA modules to fuse  $H_{dhe}$  and  $H_{cue}$  to obtain the dialogue representation  $H_c$ , and then fuse  $H_c$  with the domain-slot embeddings  $e_{ds}$  extracted from task graph.

3) The main task of domain-slot generator is using  $H_{cue}$  to generate the domain pointer  $p_d$  and slot pointer  $p_s$ , which indicates the (domain, slot) pairs mentioned in the current user

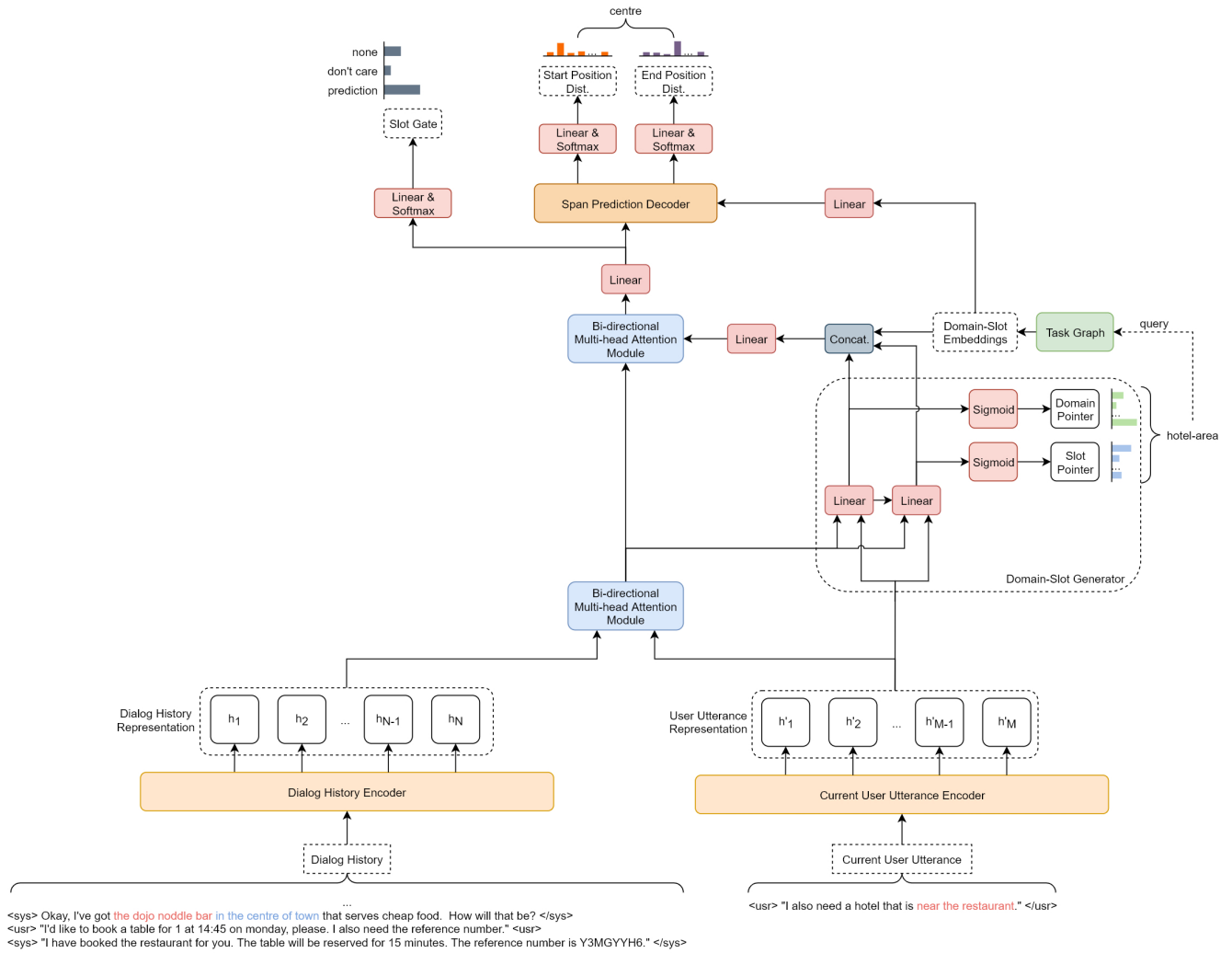


Fig. 2. The overall architecture of DST-HTG.

utterance. In order to better recognizing domains and slots, it also takes  $H_c$  as input to make the model refer to the dialogue context.

4) A task graph is utilized to model the relationships among domains and slots, and then computes  $e_{ds}$  according to query, i.e., predicted (domain, slot) pairs, with a graph attention network [13] which performs self-attention over nodes. It takes the word embeddings of generated (domain, slot) pairs from the domain-slot generator as input, and outputs their corresponding node embeddings, which will be fed into SPD.

5) SPD is a transformer decoder with linear and softmax layer that takes as input  $e_{ds}$  and  $H_{fuse}$ , the output of last BMA layer. It outputs begin position distribution  $p_b$  and end position distribution  $p_e$ , which indicate the position of slot value in the concatenation of dialogue history and current utterance. And finally, a slot gate is used to decide whether the values of predicted (domain, slot) pairs are none, don't care, or the text spans copied from dialogue.

The detail of each component will be described in the

following subsections.

#### A. Dialogue History Encoder (DHE) and Current User Utterance Encoder (CUE)

As a multi-domain dialogue is usually longer than a single-domain one, it is important for the model to capture long-distance dependencies and catch relevant information, which is a great advantage of transformer. So, as mentioned above, we use a multi-layer transformer encoder [12] to obtain the representation of the dialogue history. Unlike [5], we use the entire dialogue history rather than a fixed size of subset. This is because for a slot whose value is implicitly mentioned, the actual slot value may occur in any position in dialogue history. Specifically, at turn  $t$ , the input to the DHE is the concatenation of all words in utterances of dialogue history with additional special token, which can be denoted as:  $X_{dhe} = [\langle usr \rangle, U_1, \langle /usr \rangle, \langle sys \rangle, S_1, \langle /sys \rangle, \dots, \langle usr \rangle, U_{t-1}, \langle /usr \rangle, \langle sys \rangle, S_{t-1}, \langle /sys \rangle] \in R^{|X_{dhe}| \times d_{emb}}$ , where  $U_i$  and  $S_i$  denote the user and system utterance at turn  $i$  respectively.

$\langle usr \rangle$ ,  $\langle /usr \rangle$ ,  $\langle sys \rangle$ , and  $\langle /sys \rangle$  are four kinds of special tokens to help the model identify user and system utterances.  $d_{emb}$  is the size of word embeddings. The output of the DHE is dialogue history representation  $H_{dhe} = [h_{dhe}^1, \dots, h_{dhe}^{|X_{dhe}|}] \in R^{|X_{dhe}| \times d_{hid}}$ , where  $d_{hid}$  is the hidden size. Considering that the current user utterance is much shorter than the dialogue history in most cases, we use a single-layer transformer encoder as CUE. Similar to DHE, the CUE take as input the current user utterance  $X_{cue} = [\langle usr \rangle, U_t, \langle /usr \rangle, \langle sys \rangle, S_t, \langle /sys \rangle] \in R^{|X_{cue}| \times d_{emb}}$  and outputs the current user utterance representation  $H_{cue} = [h_{cue}^1, \dots, h_{cue}^{|X_{cue}|}] \in R^{|X_{cue}| \times d_{hid}}$ .

### B. Bi-directional Multi-head Attention (BMA) Module

Inspired by BiDAF [14], we propose BMA module to compute and fuse the context-to-query and query-to-context relevance information with multi-head attention function  $MultiHead(Q, K, V)$  proposed in [12], which makes the module extract rich relevant features and provide them to other components. The structure of BMA module is shown in Fig. 3.

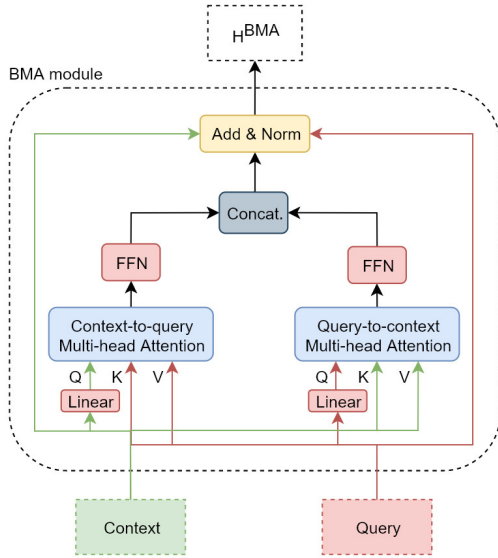


Fig. 3. The structure of BMA module.

There are two BMA modules in our proposed model. Here we introduce the first BMA module, which use  $H_{dhe}$  as context and  $H_{cue}$  as query to compute  $H_c = BMA(H_{dhe}, H_{cue}) \in R^{(|X_{dhe}|+|X_{cue}|) \times d_{hid}}$  as the representation of the whole dialogue. Another BMA module has exactly the same computation process and will be briefly introduced later. Firstly, the context-to-query multi-head attention is computed by

$$H_{c-q} = MultiHead(W_c H_{dhe}, H_{cue}, H_{cue}) \quad (1)$$

And the query-to-context multi-head attention is computed by

$$H_{q-c} = MultiHead(W_q H_{cue}, H_{dhe}, H_{dhe}) \quad (2)$$

Then we apply feed forward network (FFN) [12] on  $H_{c-q}$  and  $H_{q-c}$  respectively, and concatenate them to obtain a single matrix  $H_{FFN}$ :

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (3)$$

$$H_{FFN} = [FFN(H_{c-q}); FFN(H_{q-c})] \quad (4)$$

Finally, to enhance robustness of BMA module, we add a residual connection [15] and layer normalization [16] to the module, and obtain the output dialogue representation  $H_c$ . This process is described as follows:

$$H_c = LayerNorm(H_{FFN} + [H_{dhe}; H_{cue}]) \quad (5)$$

We denote the above equation (1)-(5) as  $H_{BMA} = BMA(context, query)$

### C. Domain-Slot Generator

Unlike previous works, we use a domain-slot generator to extract (domain, slot) pairs mentioned in the current user utterance, which makes the proposed model unnecessary to iterate over all (domain, slot) pairs. However, it is not enough to only take the current user utterance into consideration because the mentioned (domain, slot) pairs need to be determined with the extra information from dialogue history in some cases. So, we take both  $H_{cue}$  and  $H_c$  as input. Specifically, a linear map and a sigmoid layer are applied to the concatenation of  $H_{cue}$  and  $H_c$  to compute the domain pointer, i.e., the probability of mentioning each domain in the current user utterance:

$$H_d = W_d[H_{cue}; H_c] \quad (6)$$

$$P_d = \text{sigmoid}(H_d) \quad (7)$$

The slot pointer indicates slot names mentioned in the current user utterance. However, it should be computed under the constraint of domain pointer, because each domain contains different set of slots. So, we should also take  $H_d$  along with  $H_{cue}$  and  $H_c$  as input to put constraints of domains on the slot pointer generation process. Similar to the domain pointer, these three inputs are concatenated and fed into a linear map and a sigmoid layer to compute the slot pointer, which is described as follows:

$$H_s = W_s[H_d; H_{cue}; H_c] \quad (8)$$

$$P_s = \text{sigmoid}(H_s) \quad (9)$$

### D. Task Graph

The (domain, slot) pairs has natural relationships among each other, which appears in two aspects. On the one hand, in a specific dialogue, the values of (domain, slot) pairs may reuse or relate. For example, the user wants to book a hotel near the restaurant. In this case, (hotel, area) and (restaurant, area) share the same slot value, which makes user unnecessary to indicate the slot value of (hotel area) explicitly. On the other hand, among all the dialogues in a dataset, tracking strategies may share across (domain, slot) pairs. For example, the value set of (hotel, area) and (restaurant, area) are usually similar because hotels and restaurants usually co-exist in most areas,

Task Graph Structure:

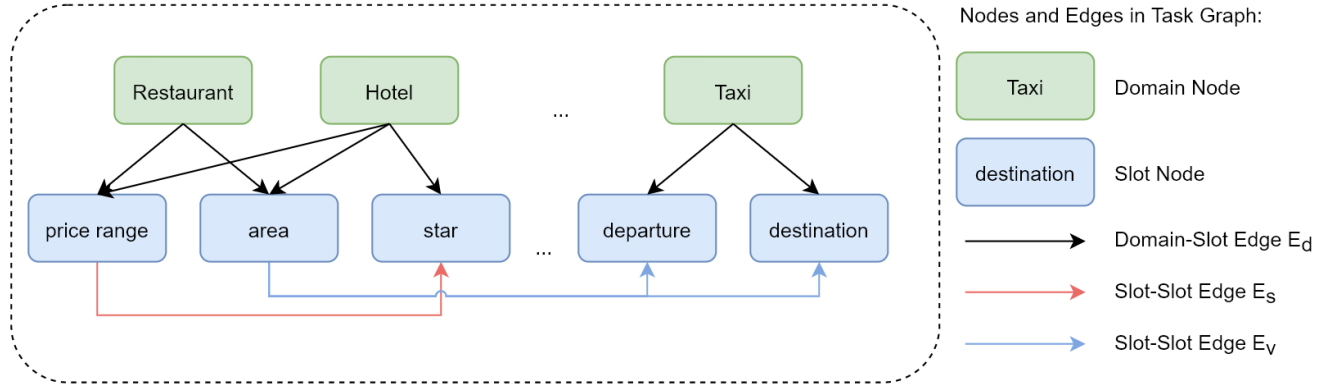


Fig. 4. The structure of hierarchical task graph.

which leads to similar state tracking strategies. So, if the DST model can utilize the relationships among (domain, slot) pairs, boosting in DST performance can be achieved.

In order to incorporate the relationships among domains and slots, we propose a task graph, which is also a core component of the proposed DST-HTG model. Fig. 4 gives an overview of task graph structure, which contains two kinds of nodes and three kinds of edges. Specifically, for a domain node  $d$  and two slot nodes  $s_1, s_2$ , the combinations of nodes and edges are listed and described as follows:

1)  $(d, E_d, s_1)$ : domain  $d$  has a slot  $s_1$ . This edge is used to construct (domain, slot) pairs using domain nodes and slot nodes.

2)  $(s_1, E_s, s_2)$ : the value set of slot  $s_1$  and slot  $s_2$  are related in semantics. For example, hotels with higher stars are usually more expensive, and vice versa. Moreover, if a user wants to book a low-star hotel and then book a restaurant, we may have a prior guess that he wants to book a cheap restaurant before he mentions the specific value of (restaurant, price range). So, we can describe the relationship between slot node “star” and “price range” using the  $E_s$  edge.

3)  $(s_1, E_v, s_2)$ : the value set of slot  $s_1$  is a subset of the value set of slot  $s_2$ . For example, the value set of (hotel, name) and (restaurant, name) are subsets of (taxi, departure) and (taxi, destination). So, we can use  $E_v$  edge to describe the relationship between slot node “name” and “departure/destination”.

This hierarchical structure also brings another two advantages. Firstly, the proposed task graph naturally contains the relationships among pairs like (hotel, area), (attraction, area) and (restaurant, area) without additional modeling because they reuse the same slot node “area”. Secondly, by refactoring the flat structure of (domain, slot) pairs into the hierarchical structure of domain and slot nodes, the complexity of graph, i.e. the number of nodes and edges, reduces by reusing these nodes and edges to represent (domain, slot) pairs. So, the task graph can incorporate the relationships among domains and slots into DST process explicitly and efficiently. To the

best of our knowledge, DST-HTG is the first DST model that incorporates a graph describing relationships among domains and slots with the hierarchical structure.

To model the task graph, we use a multi-layer graph attention network (GAT) [13] which applies attention mechanism over nodes to combine the local graph structure and node-level features. GAT can gather information from the one-hop neighborhood, which is suitable for capture the relationships among domain and slot nodes in the graph. Formally, let  $h_i^{(l)}$  represent the node embedding of node  $i$  in layer  $l$ , then the node embedding of node  $i$  in layer  $l + 1$  is computed by:

$$z_i^{(l)} = W^{(l)} h_i^{(l)} \quad (10)$$

$$e_{ij}^{(l)} = \text{LeakyReLU}(\vec{a}^{(l)T} (z_i^{(l)} || z_j^{(l)})) \quad (11)$$

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})} \quad (12)$$

$$h_i^{(l+1)} = \sigma \left( \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)} \right) \quad (13)$$

$\mathcal{N}(i)$  is the set of node  $i$ 's one-hop neighborhood, which comprises  $k$  neighbor nodes  $j_1, j_2, \dots, j_k$ . The node embeddings in the bottom layer are initialized by the word embeddings which are the same as the one used in the input of DHE and CUE. And now we can obtain the embedding of each node. As mentioned above, the domain-slot generator outputs the domains and slots mentioned in the current user utterance, which are used to construct (domain, slot) pairs. Then we can iterate over these (domain, slot) pairs and concatenate the corresponding domain embedding and slot embedding into the domain-slot embedding  $e_{ds_1}, \dots, e_{ds_n}$ , where  $n$  is the number of predicted (domain, slot) pairs. Note that we iterate over the predicted (domain, slot) pairs, which is a very small subset of the entire (domain, slot) pairs. For example, there are 30 (domain, slot) pairs in MultiWOZ 2.0 dataset, but most of the user utterances only contain less than 3 (domain, slot) pairs,



Dialogue History:	Extracted dialogue state each turn w/ task graph:	Extracted dialogue state each turn w/o task graph:
Usr: "Hello, I am looking for a Korean restaurant in the centre. Could you help me with that?"	(restaurant, food, korean), (restaurant, area, center)	(restaurant, food, korean), (restaurant, area, center)
Sys: "I have one restaurant name little seoul in the expensive price range. would you like me to book it?" Usr: "Are there any other choice? If not I would like to get a table on Wednesday at 12 for 1 please."	(restaurant, book day, wednesday), (restaurant, book people, 1), (restaurant, book time, 12:00), (restaurant, name, little seoul)	(restaurant, book day, wednesday), (restaurant, book people, 1), (restaurant, book time, 12:00), (restaurant, name, little seoul)
Sys: "I looked and this was the only restaurant serving korean food. I have booked you a table. The table will be reserved for 15 minutes. Ref# WH331DNY. Any thing else?" Usr: "Are there any places to go in the same area as the restaurant?"	(attraction, area, centre)	(attraction, area, <b>same</b> )
Sys: "There are dozens of places to go in city centre. What type of attraction are you interested in today?" Usr: "I don't really mind, as long as it is near the restaurant. What looks good to you?"		
Sys: "I would like to suggest all saints church located in the centre. Is that okay? " Usr: "Yes that is fine can you please provide the postcode?"	(attraction, name, all saints church)	(attraction, name, all saints church)
Sys: "The postcode is cb58bs. Is there anything else I can help with?" Usr: "I also need a taxi to go between them and need it to arrive at the restaurant by the booked time."	(taxi, departure, all saints church), (taxi, destination, <b>all saints church</b> ), (taxi, arrive by, 12)	(taxi, departure, city centre), (taxi, destination, <b>between them</b> ), (taxi, arrive by, 15)
Sys: "I have booked a cab to take you to the restaurant when you leave All Saint's Church. The booked car type is a yellow volkswagen. The contact number is 07396045902." Usr: "Thanks so much, that's all I need today! "		
Sys: "My pleasure. Have a great day." Usr: "Goodbye again, thanks for your help. "		
Sys: "It's no problem, contact us again any time."		

Fig. 5. A complex case in multi-domain DST, which is related to three domains: restaurant, attraction, and taxi.

which also leads to improvement in efficiency of the proposed model.

### E. Span Prediction Decoder (SPD) and Slot Gate

After obtaining the domain-slot embeddings from the task graph, SPD is applied to generate the begin and end position in dialogue for each predicted (domain, slot) pair, which indicates the text spans in dialogue. Then a slot gate is applied to determine whether the values of predicted (domain, slot) pairs are none, don't care, or the text spans copied from dialogue. To incorporate the domain and slot features into the dialogue representation  $H_c$ , we use another BMA module to fuse  $H_c$  and the information of domains and slots. Firstly, we concatenate and apply linear map on the domain-slot representations from both domain-slot generator and task graph:

$$H_{ds} = W_{ds}[H_d; H_s; W'_{ds} e] \quad (14)$$

Note that SPD is a transformer decoder, which needs two inputs to start decoding: the information from encoder and the decoder input. For the first input, we apply a BMA module to fuse  $H_c$  and  $H_{ds}$ , where  $H_c$  is served as context and  $H_{ds}$  is served as query, and outputs the representation  $H_{fuse}$  as the information from encoder, which is described as:

$$H_{fuse} = W_{fuse} BMA(H_c, H_{ds}) \quad (15)$$

For the second input, we apply a linear map on domain-slot embedding  $e_{ds}$ , and use the transformed embedding as the decoder input  $y_0$ :

$$y_0 = W_0 e_{ds} \quad (16)$$

Then, for each predicted (domain, slot) pair, SPD takes  $y_0$  and  $H_{fuse}$  as input, and decodes for a fixed number of two time steps and get two outputs  $O_1$  and  $O_2$ , which are fed into

linear map and softmax layer to obtain the distribution over the begin and end position in dialogue  $P_b$  and  $P_e$  respectively:

$$P_b = \text{softmax}(W_b O_1) \quad (17)$$

$$P_e = \text{softmax}(W_e O_2) \quad (18)$$

The slot gate is a three-way classifier to help rectify the prediction of domain-slot generator, which can improve robustness of the proposed model. The slot gate  $P_{sg}$  is computed by applying a linear map and softmax layer on  $H_{fuse}$ :

$$P_{sg} = \text{softmax}(W_{sg} H_{fuse}) \quad (19)$$

## III. EXPERIMENTS AND RESULTS

### A. Experimental Settings

Recently proposed MultiWOZ 2.0 dataset [6], which is used in this study, is the largest existing multi-domain task-oriented dialogue dataset. The basic statistics of MultiWOZ 2.0 dataset are shown in TABLE I. Following [5], we exclude the dialogues in police and hospital domain since these domains only occur in training set and take up a very small proportion in the dataset. The remaining five domains (restaurant, hotel, attraction, taxi, train) are used in our experiment.

For hyperparameters in the implementation, the hidden size of DHE, CUE, and SPD are all set to 512. We use pre-trained EIMo [17] embeddings to initialize the word embeddings, and the embedding size is set to 512, which is same as the hidden size. We set the number of layers in graph attention network to 2. The number of heads in BMA is set to 8, which is following the original setting in [12]. The learning rate is set to 0.005, and batch size is set to 64.

TABLE I  
THE BASIC STATISTICS OF MULTIWOZ 2.0 DATASET (TRAINING SET ONLY)

# Dialogues	Total # turns	Total # tokens	Avg. turns per dialogue	# Total unique tokens
8438	115,424	1,520,970	13.68	24071

TABLE II  
THE EXPERIMENTAL RESULTS OF DST-HTG ON MULTIWOZ 2.0 DATASET

Model	Joint Accuracy	Slot Accuracy
TRADE	48.62	96.92
DSTQA	51.44	97.24
DST-HTG	<b>51.68</b>	<b>98.02</b>

TABLE III  
THE EXPERIMENTAL RESULTS OF ABLATION STUDY

Model	Joint Accuracy	Slot Accuracy
DST-HTG	<b>51.68</b>	<b>98.02</b>
- context-to-query attention	50.91	97.15
- query-to-context attention	50.97	97.17
- task graph	49.25	96.44
- $E_d$	50.59	97.06
- $E_s$	51.05	97.18
- $E_v$	51.08	97.20

### B. DST-HTG Model Results

We use two evaluation metrics, joint accuracy and slot accuracy, to evaluate our proposed model. Joint accuracy is designed to check whether all predicted slot values in a dialogue state exactly match the ground truth dialogue state. And the slot accuracy checks whether the individual (domain, slot, value) triplet matches its ground truth label. Table II shows the performance of our DST-HTG model comparing with two previous works: TRADE [5] and DSTQA [11], the state-of-the-art model on MultiWOZ 2.0. As shown in the table, DST-HTG surpasses DSTQA with 0.78% absolute gains on slot accuracy and 0.24% on joint accuracy. Therefore, DST-HTG achieves both state-of-the-art joint accuracy and slot accuracy on MultiWOZ 2.0 dataset.

### C. Ablation Study

We perform three ablation experiments to prove the effectiveness of different components of DST-HTG. The results are shown in Table III. The first ablation experiment is to explore the effectiveness of the context-to-query and query-to-context multi-head attention, which are both the key parts of the BMA module. As shown in Table III, the removal of context-to-query or query-to-context attention all leads to a drop in DST performance, this is because the context-to-query and query-to-context attention are complementary information from two different directions, which helps our model to better capture the features of dialogues. This result demonstrates that the BMA module is important in DST-HTG model.

The second ablation experiment is to observe the performance of the proposed model without the task graph. We observe 2% absolute drop on both joint accuracy and slot

accuracy without the task graph. This is because the task graph contains rich relationship information among domains and slots which is important for a multi-domain DST model to track the implicitly mentioned slot values. Moreover, we do a case study to check the effect on dialogue state without the task graph. The Fig. 5 shows a complex case for DST model and the corresponding dialogue states in each turn generated by DST-HTG with and without task graph. We can observe from the generated dialogue states of two models that the DST-HTG with task graph is significantly better in capture the implicitly mentioned slot value such as (attraction, area, centre) and (taxi, departure, all saints church). This is because the task graph naturally models relationship between (attraction, area) and (taxi, departure), and the task graph has edges between node “name” and node “departure/destination”, which is utilized by model to track implicit slot values. Although it tracks a wrong value of (taxi, destination), it still tracks a name value rather than random text spans from dialogue. Moreover, as shown in Fig. 5, without the guidance of task graph, the copy mechanism cannot accurately copy meaningful text spans, which further proves the effectiveness of task graph.

To deeply explore the contribution of different edges, we conduct the third ablation experiment, in which we choose to remove some specific kinds of edges each time. Table III shows the experimental results. As shown in the table, the removal of edge  $E_d$  makes the biggest drop. This is because there are more  $E_d$  edges than another two kinds of edges. Besides, removing any one kind of edges leads to a drop in DST performance, which indicates that all three kinds of edges are necessary to construct the task graph.

## IV. RELATED WORK AND DISCUSSION

In traditional modular task-oriented dialogue systems [18], DST modules receive input from natural language understanding (NLU) modules to update the current dialogue state [19]–[21]. However, they heavily rely on the manually constructed features and semantic dictionaries to perform delexicalization, which rephrase the alternative mentions of ontology items to standard (slot, value) pairs in ontology. This drawback makes these models difficult to extend to the multi-domain setting.

In order to solve the issue of over-dependence on hand-crafted works, some neural DST models are proposed, which use neural network to learn semantic information directly from word embeddings instead of the pre-build semantic dictionaries. NBT [2] leverages semantically specialized pre-trained word embeddings to generate the distributional representation of user utterance, and then pass the representation to context modeling and semantic decoding module to generate a distribution over pre-defined ontology. However, it uses the

semantically specialized word embeddings rather than normal one trained from language model, which is more difficult to obtain and thus limits the generalization ability of the model. Besides, it is difficult to scale to the multi-domain setting because it generates the distribution over ontology from scratch for only one slot each time. GLAD [3] leverages a global module to share parameters among slots, and a local module to learn the feature representation inside each slot, which can use normal word embeddings and handle some rare slots due to the shared parameters. However, it still needs a slot-specific encoder for each slot, which makes it hard to scale to multi-domain settings.

Therefore, in order to address the issue of over-dependence on full ontology and make scalable multi-domain DST models available, recent works on multi-domain DST choose to apply the copy mechanism to directly extract slot values from dialogue context, such as TRADE which we introduce above. The other attempt to address this issue is considering multi-domain DST as other tasks. For example, recently proposed DSTQA considers multi-domain DST as a document-based question answering problem which treat dialogues as documents, and construct different questions for all (domain, slot) pairs to ask for the slot values according to dialogue contexts. However, as mentioned in the introduction, these models still fall short in modeling the relationships among domains and slots in an explicit and efficient approach, which comprises our major difference from previous works. In order to address this issue, we propose a task graph with hierarchical structure, which contains two levels of nodes and three kinds of relationships, to reduce the complexity of graph and enhance the efficiency in modeling relationships by reusing nodes in representing (domain, slot) pairs, and then utilize the hierarchical task graph as a guidance to help the model track the implicit slot values.

## V. CONCLUSION

In this paper, we propose DST-HTG, a novel open vocabulary based multi-domain dialogue state tracking model, which leverages a simple yet effective hierarchical task graph to incorporate relationships among domains and slots in an explicit and efficient way. Moreover, it is unnecessary for the model to iterate over all (domain, slot) pair in DST process, which leads to improvement in efficiency. Our model achieves state-of-the-art results on MultiWOZ 2.0 dataset, and the experimental results indicate the effectiveness of our model and the hierarchical task graph inside.

## REFERENCES

- [1] 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)*, 2014, pp. 292–299.
- [2] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1777–1788.
- [3] V. Zhong, C. Xiong, and R. Socher, "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)*, 2018, pp. 1458–1467.

- [4] E. Nouri and E. Hosseini-Asl, "Toward scalable neural dialogue state tracking model," *arXiv preprint arXiv:1812.00899*, 2018.
- [5] C.-S. Wu, A. Madotto, E. Hosseini-Asl, C. Xiong, R. Socher, and P. Fung, "Transferable multi-domain state generator for task-oriented dialogue systems," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 808–819.
- [6] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gasic, "Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 5016–5026.
- [7] S. Gao, A. Sethi, S. Agarwal, T. Chung, and D. Hakkani-Tur, "Dialog state tracking: A neural reading comprehension approach," in *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, 2019, pp. 264–273.
- [8] L. Ren, J. Ni, and J. McAuley, "Scalable and accurate dialogue state tracking via hierarchical sequence generation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1876–1885.
- [9] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1073–1083.
- [10] B. McCann, N. S. Keskar, C. Xiong, and R. Socher, "The natural language decathlon: Multitask learning as question answering," *arXiv preprint arXiv:1806.08730*, 2018.
- [11] L. Zhou and K. Small, "Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering," *arXiv preprint arXiv:1911.06192*, 2019.
- [12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, E. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," *International Conference on Learning Representations*, 2018, accepted as poster. [Online]. Available: <https://openreview.net/forum?id=rJXMpikCZ>
- [14] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," *arXiv preprint arXiv:1611.01603*, 2016.
- [15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [16] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
- [17] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of NAACL-HLT*, 2018, pp. 2227–2237.
- [18] B. Thomson and S. Young, "Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems," *Computer Speech & Language*, vol. 24, no. 4, pp. 562–588, 2010.
- [19] J. D. Williams and S. Young, "Partially observable markov decision processes for spoken dialog systems," *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.
- [20] 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*, 2013, pp. 423–432.
- [21] 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)*, 2014, pp. 282–291.