International Journal of Innovative
Computing, Information and Control
Volume **17**, Number **6**, December **2021**

ICIC International ©2021 ISSN 1349-4198
pp. 1955–1971

MULTI-STEP DIALOGUE STATE TRACKER WITH SCHEMA-GUIDED GRAPH CONVOLUTIONAL NETWORK

Xuejun Zhang[1,2], Pengyuan Zhang[1,*] and Yonghong Yan[1]

[1]The Key Lab of Speech Acoustics and Content Understanding, Chinese Academy of Sciences
Institute of Acoustics, Chinese Academy of Sciences
No. 21, North 4th Ring Road, Haidian District, Beijing 100190, P. R. China
{ zhangxuejun; yanyonghong }@hccl.ioa.ac.cn
*Corresponding author: zhangpengyuan@hccl.ioa.ac.cn

[2]Institute of Acoustics
University of Chinese Academy of Sciences
No. 19(A), Yuquan Road, Shijingshan District, Beijing 100049, P. R. China

Received June 2021; revised September 2021

Abstract. *Dialog state tracking, which is one of the most crucial modules in task-oriented dialog systems, predicts the dialog states based on a dialog history. For multi-domain DST, lacking dependencies reasoning across domains and slots is also a major obstacle due to increasingly complex interactive dialogues. Existing methods that extract the values from a pre-defined slot value list or generate values based on the dialog context have difficulty tracking slot values that are not explicitly mentioned in the pre-defined value list and the dialogue history. In this paper, we propose a* ***Mu****lti-****s****tep* ***D****ialogue* ***S****tate* ***T****racker with Schema-Guided Graph Convolutional Network (MusDST) that incorporates a multi-step prediction procedure for dialogue state tracking. In this procedure, a novel transfer mechanism is proposed to track the slot values that are not pre-defined in the list or explicitly mentioned in the dialogue history. Also, we model slot relations using graph convolutional network to reason slot values over schema graphs. Experiments demonstrate that our proposed MusDST achieves state-of-the-art performance on both MultiWOZ 2.0 and MultiWOZ 2.1 datasets. Also, we show the capability of our model in tracking not explicitly mentioned slot values.*

Keywords: Task-oriented dialog system, Graph convolutional network, Dialog state track, Schema-guided network

1. **Introduction.** As a typical human-computer interaction system, task-oriented dialogue systems have aroused widespread research interest. DST allows for natural and personalized interactions with users to assist them with achieving goals such as booking a restaurant or a flight. Dialogue State Tracking (DST) is a core component of the task-oriented dialog system which is important for proper dialogue management [10, 37]. Recently, motivated by commercial applications like Google Assistant, Apple Siri, or Amazon Alexa, there is significant interest in adding numerous domains to these dialogue systems. Therefore, multi-domain DST becomes crucial.

Multi-domain DST predicts the state of a dialogue based on the dialogue history. A state is usually represented as a triplet (domain, slot, value) which represents the value of the slot in the current domain. Turn states are the specified (domain, slot, value) triplets of the current turn, and joint goals are the set of accumulated turn states. The target of DST is to track joint goals. An example of a dialogue with DST label is shown in Appendix A, where the user books attraction and hotel first, and then calls a taxi.

DOI: 10.24507/ijicic.17.06.1955

Traditional methods replace the slot values in dialogue history that change lexically with general tags to obtain delexicalization characteristics. However, these methods usually depend on complex domain-specific lexicons and hand-crafted generic features [15, 24, 33, 42]. The recently proposed data-driven methods based on the deep learning model have shown hopeful performance in various fields [20, 23, 35].

[23, 40] proposed the picklist-based methods which regard (domain, slot) pairs as picklist slots. The slot values are predicted by employing classification on the list of candidate values. These approaches rely on an ontology in which the slots and values are pre-defined and cannot change dynamically. Regrettably, that is unrealistic in a realistic scenario. Taking the travel domain as an example, new hotels or attractions are likely to be added, leading to the changes of the ontology.

To address this issue, generative methods generate slot values directly instead of relying on the fixed ontologies or dictionaries [21, 35, 36]. Unfortunately, such methods may generate malformed strings such as repeated tokens when generating complex slot values. For instance, a small difference in train destination can make the dialogue state tracking result wrong.

Recently, Chao and Lane, Gao et al., and Le et al. introduce span-based methods, which view (domain, slot) pairs as span slots [4, 11, 19]. In these approaches, the values will be extracted from the dialogue history by span matching. While this method is more robust to track dialogue states, it is challenging to process slot values that do not appear in the dialogue history. As shown in Figure 4 in Appendix A, the user specifies the value of the "`internet`" slot in the "`hotel`" domain as "`yes`" by saying "`I would also prefer that they offer free parking and wifi`".

To address these challenges, Zhang et al. [38] and Zhou and Small [41] propose to combine span-based methods and picklist-based methods by treating (domain, slot) pairs as span slots and picklist slots. The models extract the values of the span slots by span matching with dialogue context and pick the values of the picklist slots from the corresponding pre-defined value list. It is a human choice to determine if a slot is a picklist slot or a span slot.

However, many of the previous works are not efficient in terms of computational complexity and memory consumption since they predict all the slot values from scratch at every dialogue turn. [6, 16] take in certain turns of utterances instead of all the dialogue history to achieve efficient DST by using the previous dialog state at each turn.

Existing approaches generally either directly generate the value based on the current dialogue history or pick the value from the pre-defined value list for each slot. However, some slot values cannot be pre-defined and do not appear explicitly in the current dialogue history. Instead, we need to consider the interaction between different slots and obtain their values through reasoning. Taking the last turn in Figure 4 in Appendix A as an example, the user said "`I need a taxi from the attraction to the hotel`" to book a taxi, but did not indicate the specific departure and destination. We need to get the real departure and destination, namely the names of the attraction and the hotel, through interactive analysis of slots and reasoning.

To address these issues, we propose **Mu**lti-**s**tep **D**ialogue **S**tate **T**racker with Schema-Guided Graph Convolutional Network (**MusDST**), modeling the values of (domain, slot) pairs as different types. We introduce multi-step prediction instead of directly predicting from scratch every time, which improves efficiency while improving the performance of DST. For the slot values that need inference, we propose a transfer mechanism for copying the values between slots, enabling switching between domains, and accepting the values offered by other slots during dialogue. Also, the capability of sharing the knowledge between the slots and domains might help a model to work across multiple domains, as well

as to handle the slot values that need reasoning. Thus, we introduce Graph Convolutional Network (GCN) to build a schema graph that models the relationship between different domains and slots. Transfer mechanism and schema-guided GCN are complementary to each other.

The main contributions are as follows:

- A novel multi-step dialogue state tracker with schema-guided graph convolutional network is proposed to track dialogue state efficiently.
- We propose a novel transfer mechanism for copying the values between slots, indicating the necessity for proper reasoning over domains and slots.
- The model employs schema encoder to model slot interactions and relations with graph convolutional network, enabling a model to work across multiple domains and benefiting the transfer mechanism for better reasoning.
- Experiments demonstrate that our proposed MusDST achieves state-of-the-art performance on both MultiWOZ 2.0 and MultiWOZ 2.1 datasets.

The remainder of this paper is organized as follows. Related work to the deep learning-based DST and GCN is reviewed in Section 2. In Section 3, we present our proposed multi-step dialogue state tracker with schema-guided graph convolutional network. Section 4 shows the experimental setting in detail. In Section 5, we conduct empirical analysis, discuss the experimental results and propose further analysis. Finally, the conclusion and future work are summarized in Section 6.

We will provide the source code after publication.

2. **Related Work.**

2.1. **DST.** To prevent gathering mistakes from Spoken Language Understanding (SLU), early dialogue state trackers usually learn DST and SLU jointly [13, 27, 34]. Traditional delexicalization trackers [14, 15, 31, 32, 33] use handcrafted dictionaries to list basic words, mentions, and rephrasing which achieve delexicalization and generalization. [23] uses a Convolutional Neural Network (CNN) to learn dialogue history representation, achieving better performance than handcrafted feature-based models in 2018. However, parameters of the model are not shared over slots. [25] employs a global module to share parameters across different slots.

[40] exploits a global-local model which learns slot-specific features with a local module and global features with a global module. Later, [29] presents a new model called StateNet, which gets the slot value by comparing the distances between the value vectors in the pre-defined value list and the dialog history representation. However, all these methods regularly need a pre-defined ontology which is hard to acquire in advance. The number of values and slots in the different domains may be very large and changeable even if the ontology exits.

To solve the problem above, some span-based approaches are introduced to extract the slot values from the dialogue history directly. Xu and Hu [36] try to copy the slot values from the dialog history with a pointer network. Chao and Lane, Gao et al., and Perez and Liu model DST as a reading comprehension task. They obtain the dialogue state by span matching with the dialogue history [4, 11, 27]. However, it is not sufficient to track the dialogue states from just the dialogue history, because many slot values cannot be found in the history owing to different descriptions of slot values or annotation errors.

Wu et al. [35] propose to generate the slot values of all slots directly using a hierarchical decoder without relying on spans or fixed vocabularies. However, such generative methods are easy to generate malformed strings especially when the slot values are relatively long strings. In contrast, both span-based and picklist-based approaches can depend on currently existing strings instead of generating them.

Zhou and Small [41] and Zhang et al. [38] propose to separate the slots into span slots and picklist slots relying on humans. For instance, the requests for "price" are usually "expensive" or "cheap" with limited choices which are treated as picklist slots. The slot such as "book time" that has unlimited values is treated as span slots.

However, many of the previous works are not efficient in terms of computational complexity and memory consumption since they predict all the slot values from scratch at each dialog turn. [6, 16] take in certain turns of utterances instead of all the dialogue history to achieve efficient DST by using the previous dialog state at every turn.

Existing approaches generally either directly generate the value based on the current dialogue history or pick the value from the pre-defined value list for each slot. However, some slot values cannot be pre-defined and do not appear explicitly in the current dialogue history. As a result, the performance of the model in tracking these slot values is very poor. Also, they predict the value for each slot independently, which does not incorporate slot relations and support information interactions among different slots.

2.2. **Graph convolutional network.** The knowledge graph is a novel type of data structure that represents objects and their relationships as nodes and edges respectively [30]. The neural network on the graph that is called graph neural network was first proposed by Bruna et al. in 2014 [2]. GCN is a novel graph network model for representation learning of graphs [17]. The various important operations and generalization frameworks of GCN have been successful in many NLP tasks like semantic role labeling and neural machine translation [1, 17, 22].

GCN can capture local context information by encoding edge or node features. Inspired by the capability of GCN in determining neighborhood-aware edge features, we propose the first to introduce GCN in DST.

3. Model.

3.1. **Definition.** For multi-domain DST, we assume that there are M domains $D = \{d_1, d_2, \ldots, d_M\}$. Each domain $d \in D$ has N^d slots, i.e., $S^d = \left\{s_1^d, s_2^d, s_3^d, \ldots, s_{N^d}^d\right\}$. For each slot $s \in S^d$, there are K^s possible values, i.e., $V^s = \{v_1^s, v_2^s, v_3^s, \ldots, v_{K^s}^s\}$. For example, the "attraction" domain has a slot named "area", and the possible values are "centre", "west", "east", "south", and "north". However, it is impossible to pre-define all values V^s for some slots. Taking the slot "name" in the "hotel" domain as an example, since the number of V^s could vary with time, it is impossible to enumerate all possible hotels.

The dialog history is denoted as $X = \{(U_1, R_1), (U_2, R_2), (U_t, R_t), \ldots, (U_T, R_T)\}$, where R_t is the response of agent and U_t is the user utterance at the t-th dialogue turn. We define $B = \{B_1, B_2, B_t, \ldots, B_T\}$ as the dialog states, where B_t is the dialogue state of the t-th dialogue turn. For multi-domain DST, the dialog state B_t consists of tuples like (domain, slot, value). Let us assume there are K (domain, slot) pairs, and Y_k is the value of the k-th (domain, slot).

If the user has not specified a slot value, we fill the slot value with "None". Similarly, the slot value will be set as "does not care" if the user claims that they do not care about the value. We define these values as "special" values. Different from the traditional method, we classify "yes" and "no" as "special" values. Given that the value of "yes" or "no" is usually specified by the user with a full utterance, rather than directly appearing in the users utterance.

Therefore, B is the prediction target of the model. However, lots of previous methods appear to have high memory consumption and computational complexity since they obtain the dialogue state based on the whole dialogue history from scratch. That is they do not explicitly utilize the previous dialogue state at each turn. In our proposed MusDST

model, we update dialogue state from B_{t-1} to B_t depending on the current dialogue history $X_t^h = \{(U_{t-1}, R_{t-1}), (U_t, R_t)\}$ instead of all the dialogue history X.

$$B_t = f_{MusDST}\left(B_{t-1}, X_t^h\right) \tag{1}$$

The input of our model is state of last turn B_{t-1}, dialog history X_t^h at turn t, and output is the dialog state of current turn B_t. The values of some slots in B_t are the same as that in B_{t-1} because the user has not changed their values. We define these slot values in B_t as "`carryover`" slot value, and we can get the values directly from B_{t-1}. For example, the value of (attraction, type) pair in the second turn of dialogue state in Figure 4 in Appendix A is inherited from the first turn. We define the slot values that are explicitly specified or modified by the user in the current turn as the "`generate`" slot value. We can directly generate its value through the dialogue history, because its value can generally be found in the dialogue history, such as the value of (attraction, name) pair in Figure 4 in Appendix A.

However, it is very challenging to extract the slot value that is not explicitly mentioned in the current dialog history X_t^h. Taking the values of (taxi, departure) and (taxi, destination) in the last turn in Figure 4 for example, we need to consider dependencies that occur on dialogues, and reason over slots. To solve this issue, we proposed a transfer mechanism, which believes that these slot values need to be transferred from other slots through reasoning. Specifically, we define these slot values as "`COPY`" slot value (Section 3.4), and design a "`Copy Value Decoder`" (Section 3.5.4) to get their values. This is one of our main contributions. To the best of our knowledge, we are the first to propose a simple but effective mechanism to deal with these very challenging slot values for stronger performance.

3.2. **Overall framework.** The proposed MusDST model is shown in Figure 1 which consists of four components: schema encoder, category predictor, value decoder, and state updater. The schema encoder is a graph convolutional network that encodes the ontology schema into schema embeddings. These schema embeddings help the category predictor and the value decoder to share or transfer information across different domains and slots. That is, the utterance and ontology schema embeddings are fed to the category predictor and value decoder. The category predictor is a BERT model which predicts the category of each (domain, slot) pair from the current dialog history X_t^h. There are four sub-modules in the value decoder for decoding different types of slot values, namely "`special value decoder`", "`generate value decoder`", "`carryover value decoder`" and "`copy value decoder`". We obtain the turn belief with the value decoder module. Finally, the state updater takes the previous dialogue state and the current turn belief as input and then predicts the joint belief of the dialog by summarizing and aggregating the knowledge between turns.

That is, our proposed novel MusDST model obtains the joint belief via multi-step prediction instead of predicting directly from all dialogue history, which lowers both the training and inference time. This is one of our main contributions. Details of our proposed model will be introduced in the following sections.

3.3. **Schema encoder.** In the conventional approach, ontology schema information is not fully used to track the dialogue state. Here we utilize a schema encoder to model slot interactions and relations via GCN.

We used the similar method introduced in [6] for modeling the ontology schema. The schema graph $G = (V; E)$ is defined based on the ontology scheme. We show an example on the right-hand side of Figure 1. The nodes in our graph comprise all slots like food, time, area, and all domains such as restaurant, hotel, train. We divide the slots or domains

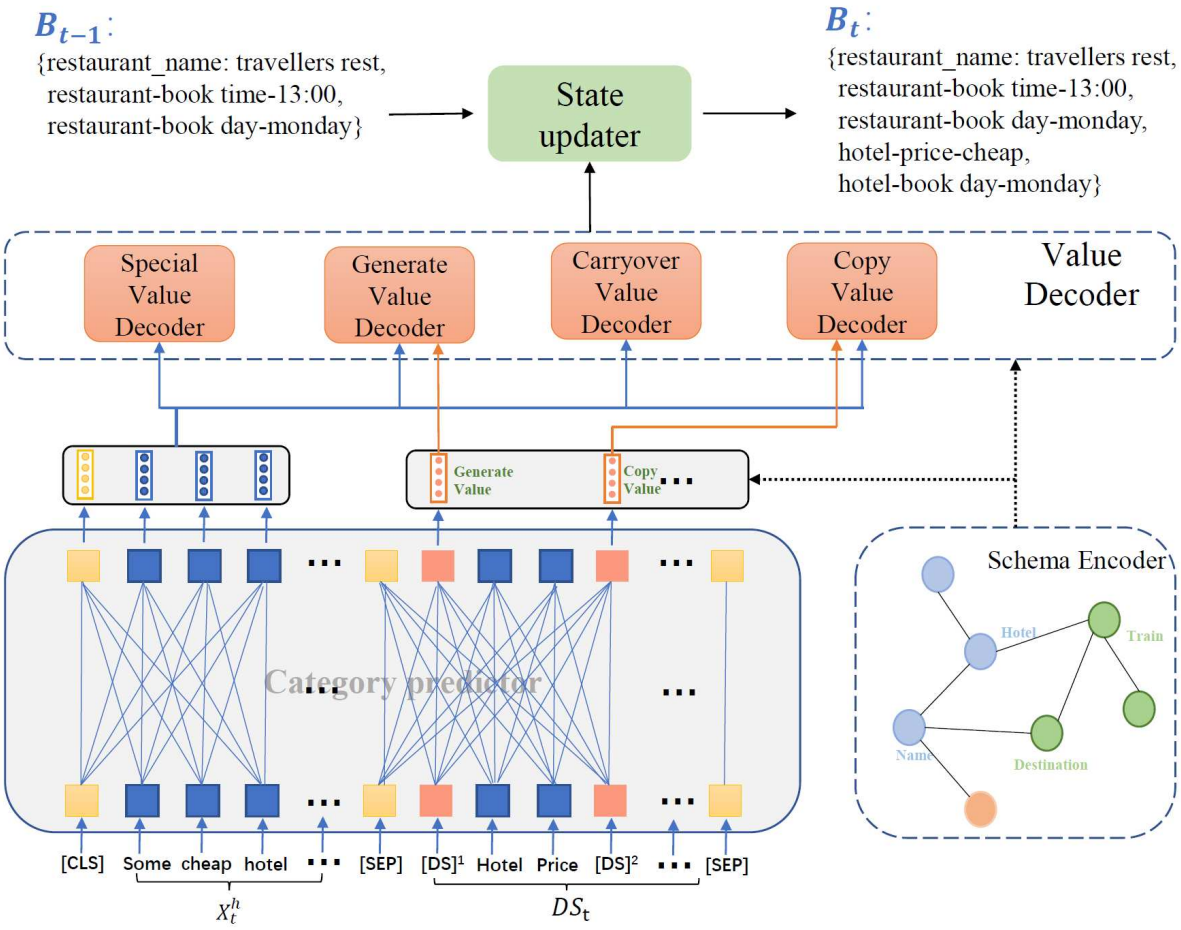

FIGURE 1. The architecture of our proposed MusDST model. MusDST takes the dialog history X_t^h at turn t and the previous dialogue state B_{t-1} as the input and outputs the current dialogue state B_t. This is performed by four sub-components: schema encoder, category predictor, value decoder, and state updater. Schema encoder encodes the ontology schema into schema embeddings. These schema embeddings help the category predictor and the value decoder to share or transfer information across different domains and slots. Category predictor takes X_t^h, and B_{t-1} as the input and predicts the categories of the slot values. There are four sub-modules in the value decoder for decoding different categories of slot values. Finally, the state updater takes the previous dialogue state and output of the value decoder as input and then outputs the joint belief of the dialogue by summarizing and aggregating the knowledge between turns.

which are described with more than one word into single nodes, such as price range, and there are edges to link them. If the slots belong to the domains, there are edges between them. For example, there is an edge between "`hotel`" domain and "`name`" slot. Also, we link the slots if they consist of the same slot values, such as "`destination`" and "`name`".

The schema encoder is a two-layer GCN [17, 28] with highway gates. We represent the node in the l-th GCN layer as N^l, and the output N^{l+1} can be calculated as:

$$\tilde{A} = A + I \tag{2}$$

$$N^{l+1} = \xi\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}N^l W^l\right) \tag{3}$$

where I is an identity matrix, $\tilde{A}$ is the adjacency matrix of the primal graph G. $\tilde{D}$ is the diagonal node degree matrix of $\tilde{A}$. $W^l \in \mathbb{R}^{d^l \times d^{l+1}}$ is the learnable weight matrix of the l-th layer, d^l and d^{l+1} are the hidden size of the l and $l+1$ layers respectively. Here, all the hidden size is the same, so we represent the hidden size as d for simplicity. ξ is the activation function.

Also, we employ layer-wise gates between GCN layers, which is similar to the method described in [28]:

$$T\left(N^l\right) = \sigma\left(X^l W_T^l + b_T^l\right) \tag{4}$$

$$N^{l+1} = T\left(N^l\right) \cdot N^{l+1} + \left(1 - T\left(N^l\right)\right) \cdot N^l \tag{5}$$

where $W_T^l \in \mathbb{R}^{d \times d}$ and $b_T^l \in \mathbb{R}^d$ are the learning parameters of the gate $T\left(N^l\right)$; σ is a sigmoid function and $\cdot$ is element-wise multiplication. $N^{l+1} \in \mathbb{R}^d$ is the token embedding in the schema graph.

We exploit GCN as a schema encoder to model domain relations and slot interactions, which help a model to work across multiple domains. Also, the capability of GCN to share the knowledge between the slots and domains might help a model to reason over slots. That is, the transfer mechanism and the schema encoder are complementary to each other. To the best of our knowledge, we are the first to propose such novel complementary modules for DST.

3.4. **Category predictor.** Category predictor is a BERT model which performs slot value category predictor as a classification task.

Input Representation. We represent the current dialogue history at turn t as the $X_t^h = \{[CLS] \oplus U_{t-1} \oplus R_{t-1} \oplus U_t \oplus R_t \oplus [SEP]\}$, where $[CLS]$ and $[SEP]$ are special tokens, which are used to mark the start of input and the end of the current dialogue history respectively.

We represent all the (domain, slot) pairs as $DS_t = DS_t^1 \oplus DS_t^2 \oplus \cdots \oplus DS_t^K$, where $DS_t^k = [DS]^k \oplus d^k \oplus s^k$ is the representation of the k-th (domain, slot). $[DS]^k$ is also a special token to generalize the message of the k-th (domain, slot) pair into a vector, like $[CLS]$ in BERT [8].

The whole input of the category predictor is the concatenation of the current dialogue history and the (domain, slot) pairs. We use a BERT model as the category predictor and the input to BERT is the input tokens embeddings, position embeddings, and segment id embeddings. The segment id of tokens that belong to X_t^h is 0 and tokens that belong to DS_t is 1. We use the same method of BERT to calculate the position embeddings.

The output of BERT is the encoded dialogue history $H_t \in \mathbb{R}^{L \times d}$, and the output corresponds to $[CLS]$ $h_t^{[CLS]} \in \mathbb{R}^d$ and the aggregated representations of the (domain, slot) pairs $h_t^{[DS]^k} \in \mathbb{R}^d$, where L is the length of the input tokens, k is the k-th (domain, slot) pair, and d is the hidden size which is the same as schema encoder.

Category Predictor. Essentially, the category predictor is a four-way classifier. At turn t, a category $c_t^k \in C = \{SPECIAL, CARRYOVER, GENERATE, COPY\}$ is predicted by the category predictor for the k-th (domain, slot) pair. Different types of (domain, slot) pairs will be sent to the corresponding type of decoder to extract their values.

The outputs of BERT along with the schema embeddings are used as inputs to the category predictor to predict the slot value category. The schema embedding of the k-th (domain, slot) pair $g_t^k \in \mathbb{R}^d$ is the summed token embedding of the domain and slot in the schema graph. Thus, the category predictor for the k-th (domain, slot) pair is defined as

$$h_1 = GELU\left(W_1 h_t^{[DS]^k} + b_1\right) \tag{6}$$

$$h_2 = GELU\left(W_2\left(h_1 \oplus g_t^k\right) + b_2\right) \tag{7}$$

$$P_t^k = softmax(W_3 h_2) \tag{8}$$

where $GELU$ is the activation function used in BERT, $W_1, W_2 \in \mathbb{R}^{d\times 2d}$, $W_3 \in \mathbb{R}^{|C|\times d}$, $b_1, b_2 \in \mathbb{R}^d$ are learnable parameters. $P_t^k \in \mathbb{R}^{|C|}$ is the probability distribution over categories for the k-th (domain, slot) at turn t. In our paper, $|C| = 4$, because $C =$ "`SPECIAL`", "`CARRYOVER`", "`GENERATE`", "`COPY`". Then we get the category:

$$c_t^k = \arg\max\left(P_t^k\right) \tag{9}$$

Next, the different types of decoders were performed on the corresponding type of slots.

3.5. **Value decoder.** Value decoder consists of four sub-modules, which are used to decode different types of slot values.

3.5.1. *Carryover value decoder.* The slots whose category is "`CARRYOVER`" are passed to the carryover value decoder. This decoder tries to get a value for the slot from the previous system state B_{t-1}. That is the "`CARRYOVER`" slots would not get updated in the current turn.

3.5.2. *Special value decoder.* Special value decoder is used to retrieve the slot values of the slots classified as "`SPECIAL`". The slot value $v_t^k \in S = \{None, does\ not\ care, yes, no\}$ is chosen by the decoder. Specifically, the decoder performs four-way classification on top of the representation of (domain, slot) pair:

$$P_{s,t}^k = softmax\left(W_s\left(h_t^{[DS]^k} \oplus g_t^k\right)\right) \tag{10}$$

$$v_{s,t}^k = \arg\max\left(P_t^k\right) \tag{11}$$

where $W_s \in \mathbb{R}^{|S|\times 2d}$ is the parameter of the network, $h_t^{[DS]^k}$ and g_t^k are the vector representation and schema embedding of (domain, slot) pair respectively. $\oplus$ indicates the concatenation of two vectors. $P_{s,t}^k \in \mathbb{R}^{|S|}$ is the probability distribution over special values for the k-th (domain, slot) pair whose category is "`SPECIAL`". Here, $|S| = 4$, because $S = \{None, does\ not\ care, yes, no\}$. $v_{s,t}^k \in S$ is the final value.

3.5.3. *Generate value decoder.* A generate value decoder generates slot values for the slots whose category is "`GENERATE`".

We use Gated Recurrent Unit (GRU) [7] introduced in [35] to extract the slot value directly from the current dialogue history at turn t.

For the k-th (domain, slot) pair in "`GENERATE`" category, we initialize GRU with $h_t^{k,0} = h_t^{[CLS]}$, $w_t^{k,0} = h_t^{[DS]^k} \oplus g_t^k$, and recurrently update the hidden state $h_t^{k,m}$ by taking a word embedding $w_t^{k,(m-1)}$ as the input until [EOS] token is generated:

$$h_t^{k,m} = GRU\left(h_t^{k,(m-1)}, w_t^{k,m}\right) \tag{12}$$

Similar to [35], we map the hidden state $h_t^{k,m}$ into the vocabulary space to get a probability distribution over the vocabulary $P_{vocab,t}^{k,m}$, where $E \in \mathbb{R}^{|V|\times d}$ is the vocabulary embedding and $|V|$ is the size of the vocabulary. In addition, we calculated a probability distribution over the dialog history $P_{history,t}^{k,m}$:

$$P_{vocab,t}^{k,m} = softmax\left(E \cdot \left(h_t^{k,m}\right)\top\right) \in \mathbb{R}^{|V|} \tag{13}$$

$$P_{history,t}^{k,m} = softmax\left(H_t \cdot \left(h_t^{k,m}\right)\top\right) \in \mathbb{R}^{\left|X_t^h\right|} \tag{14}$$

The final probability distribution over the vocabulary for GENERATE slot $P_{g,t}^{k,m}$ is the weighted-sum of two distributions:

$$P_{g,t}^{k,m} = P_{k,m}^{gen} \times P_{vocab,t}^{k,m} + \left(1 - P_{km}^{gen}\right) \times P_{history,t}^{k,m} \tag{15}$$

where $P_{k,m}^{gen}$ is a trainable parameter.

3.5.4. *Copy value decoder.* Existing approaches generally either directly generate a slot value based on the current dialogue history or extract a value from the pre-defined value list for each slot.

For the slots whose category is "COPY", their values cannot be pre-defined and do not appear explicitly in the current dialogue history. Instead, we need to consider the interaction between different slots and obtain their values from other slots in the previous dialog state.

Copy value decoder would also utilize a projection layer to predict which slot value we will copy. For the k-th (domain, slot) pair whose category is "COPY", its representation from BERT $h_t^{[DS]^k}$ and its schema embedding g_t^k are given as inputs to the copy value decoder:

$$h = GELU\left(W_{c1}\left(h_t^{[DS]^k} \oplus g_t^k\right) + b\right) \tag{16}$$

$$P_t^k = softmax(W_{c2}h) \tag{17}$$

where P_t^k is the probability distribution over all candidate (domain, slot) pairs. Then, we get the (domain, slot) copied from by $idx_{c,t}^k = \arg\max\left(P_{c,t}^k\right)$ and the new slot value is copy from it:

$$v_{c,t}^k = v_{t-1}^{idx} \tag{18}$$

3.6. **State updater.** State updater would use all the results from the predictor and value decoders along with the previous dialog state to produce the current dialogue state. The update process can refer to Figure 2 below.

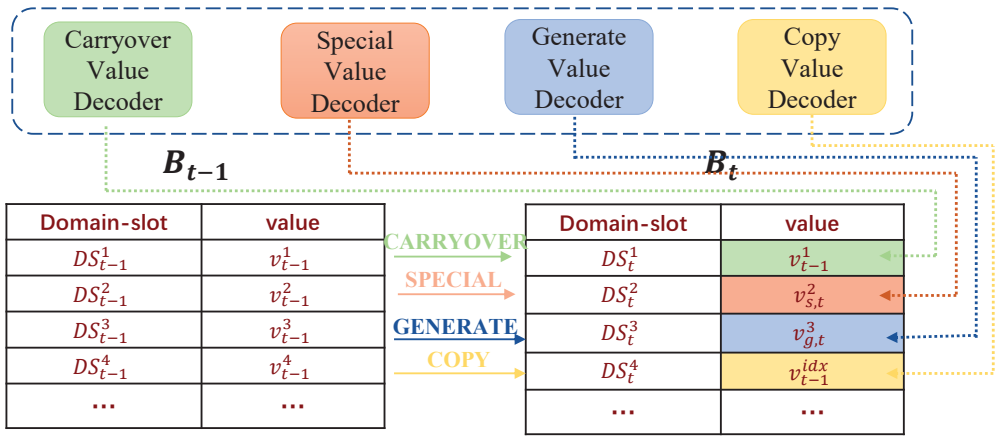

FIGURE 2. Update process

We start with the list of slots and values from the previous dialog state, and update it as the following:

- We keep the value of a slot unchanged when its category is "CARRYOVER".
- "SPECIAL" category sets the value of a slot to a special value obtained from a special value decoder.

- We set the value of a slot whose category is "`GENERATE`" to a new value generated by the generate value decoder.
- "`COPY`" category copies the value of the slot predicted by copy value decoder as its new value.

4. Experimental Setting.

4.1. **Data.** In our paper, the recently released MultiWOZ 2.0 dataset [3] and MultiWOZ 2.1 [9] dataset, which consist of dialogues across different domains, are used for experiments. There are seven domains (hospital, restaurant, attraction, train, hotel, police, and taxi) and 37 slots. MultiWOZ 2.1 corrects the annotation errors of MultiWOZ 2.0. We do experiments in the five domains (restaurant, attraction, train, hotel, and taxi) since the remaining two domains did not appear in the test set. The datasets contain 30 (domain, slot) pairs and 18529 dialogues. We use the original labels from the train, development and test set in the experiments. Table 1 shows the information of MultiWOZ datasets.

TABLE 1. The information of the MultiWOZ datasets. The number of dialogs of each domain is shown in the last three rows.

Domain	Restaurant	Hotel	Taxi	Train	Attraction	All Domains
Slots	book day book time price range name area food book people	Internet type book stay parking price range name book day stars book people area	leave at arrive by destination departure	arrive by leave at departure day destination book people	area type name	30 slots
Train	2717	3381	1654	3103	3813	14668
Dev	438	416	207	484	401	1946
Test	395	394	195	494	437	1915

4.2. **Metrics.** Similar to [23, 35, 39], we use two evaluation metrics below:

- Goals (joint goal accuracy): the proportion of dialogue turns where the dialogue states are correctly predicted. We compare the predicted dialogue state of every turn to the ground truth B_t, and the outputs are considered to be correct if and only if all the slot values exactly match the values in B_t.
- Slots (slot accuracy): the proportion of slots where the values are correctly extracted. Similarly, we compare every (domain, slot, value) to its ground truth label in B_t.

4.3. **Implementation details.** The pre-trained BERT-base-uncased is used for the category predictor. The max sequence length of inputs is 256. We use GRU [7] as generate value decoder, and employ a greedy search strategy.

The hidden size of schema encoder, predictor, and all decoders is the same, i.e., $d = 768$. Also, all the modules share the token embedding matrix. We set the dropout rate to 0.2. We train all modules jointly for 30 epochs with a batch size of 32. The category predictor makes use of a pre-trained model, whereas the value decoder needs to be trained from scratch. Therefore, we use different learning rate schemes for the predictor and the

decoder. The peak learning rate and warmup proportion for predictor are 1e-5 and 0.1; in contrast, we set them as 1e-4 and 0.1 for the decoder. We use the ground-truth label of the last turn for training and the predicted results for inference. We run five times to report the average results.

4.4. **Baselines.** The baselines are described below:

- **HYST [12]**: HYST is an open-vocabulary method which uses hierarchical RNNs to generate values.
- **TRADE [35]**: TRADE uses an encoder-decoder architecture with attention-based copying to extract the slot values. The input sentence of TRADE is the dialogue history, and the output sentence is the dialogue states.
- **NADST [19]**: NADST bases dialogue state tracking as a nonautoregressive procedure using a transformer-based decoder.
- **PIN [5]**: PIN introduces an interactive encoder to model the relationship between slots, and generates the slot values with a copy mechanism that copy tokens from system response or user utterances selectively.
- **DST-SC [26]**: DST-SC is also an open vocabulary method based on an encoder-decoder architecture. It employs slot connections module to consider slot correlations across different domains.
- **BDST [18]**: BDST uses a bi-level dialogue state tracker to track the dialogue state through learning information at both domain and slot level separately.
- **SOM [16]**: SOM realizes more effective DST by introducing an overwriting mechanism. It predicts dialogue state operation of each slot and overwrites the dialogue state of previous turn with new values based on the predicted dialogue state operations.

5. Results and Analysis.

5.1. **Joint state accuracy.** Table 2 reports the average results of MusDST. We can see that MusDST achieves the highest performance which indicates that MusDST can track the dialogue state effectively.

TABLE 2. Joint goal accuracy on MultiWOZ 2.0 and MultiWOZ 2.1

Model	MultiWOZ 2.0		MultiWOZ 2.1	
	Goals (%)	Slots (%)	Goals (%)	Slots (%)
HYST	42.33	95.44	38.10	95.46
TRADE	48.62	96.92	45.35	96.55
NADST	50.52	–	49.04	–
PIN	52.44	97.28	48.40	97.02
DST-SC	52.24	–	49.58	–
BDST	50.14	97.30	49.55	–
SOM	51.72	–	53.01	–
MusDST	**61.27**	**97.97**	**57.49**	**97.75**

For MultiWOZ 2.0, MusDST achieves 9.55% absolute improvement and 18.46% relative improvement on the "`Goals`". As for MultiWOZ 2.1, MusDST has a 4.48% relative improvement and an 8.45% absolute improvement on the "`Goals`". Our performance gain can be attributed to our multi-step prediction mechanism, transfer mechanism that reasons between different slots, and GCN that models the relationship between slots.

The performance difference between SOM and our model mainly comes from the transfer mechanism and the schema encoder. These two complementary modules can model domain relations and slot interactions, and reason between slots to track slot values that do not appear in the dialogue history. For example, the user said, "`I need a taxi from the attraction to the hotel`" to book a taxi, but did not indicate the specific departure and destination. We need to get the real departure and destination by copying the values from (attraction, name) and (hotel, name).

5.2. **Domain-specific accuracy.** We run domain-specific experiments on a subset of the test set, where the subset is composed of the slots that belong to a specific domain. Table 3 shows that MusDST outperforms other models in all domains, especially in "`Restaurant`" and "`Hotel`".

TABLE 3. Domain-specific accuracy on MultiWOZ 2.1

Domain	Model	Goals (%)	Slots (%)
Attraction	NADST	66.83	98.79
	SOM	69.83	98.86
	MusDST	**71.35**	**98.92**
Hotel	NADST	48.76	97.70
	SOM	49.53	97.35
	MusDST	**60.54**	**97.99**
Restaurant	NADST	33.80	96.69
	SOM	65.72	98.56
	MusDST	**68.53**	**98.72**
Taxi	NADST	50	50
	SOM	59.96	98.01
	MusDST	**60.74**	**98.36**
Train	NADST	62.36	98.36
	SOM	70.36	98.67
	MusDST	**72.16**	**98.79**

For multi-domain DST, the slots that appear in different domains are not independent. For instance, if a user booked an attraction in the east, then the hotel he requested may also be in the east. Also, if a user booked a hotel, then the departure of the taxi is probably that hotel. We define these slots as cross-domain slots. An obvious feature of the two domains of "`Restaurant`" and "`Hotel`" is that there are more cross-domain slots.

The high performance in these two domains reveals that our proposed MusDST model is more effective in tracking such cross-domain slots. This strength can be attributed to our transfer mechanism and schema graph that share or transfer information across different domains and slots, directly optimizing the tracking accuracy of cross-domain slots.

5.3. **Slot-specific accuracy.** The goal accuracy analysis of MusDST on different (domain, slot) pairs is shown in Figure 3. The MusDST model achieves much higher goal accuracy than TRADE on all (domain, slot) pair especially on the cross-domain slots, such as (train, destination). This result implies the validity of the transfer mechanism and schema encoder for sharing the knowledge between slots so that the values of cross-domain slots are exactly tracked.

As shown in Figure 3, correctly predicting the name slots, such as (hotel, name), (restaurant, name), is the most challenging. This is since these slots often have lots of possible values that are difficult to track. We also find that the (hotel, type) slot which is a simple

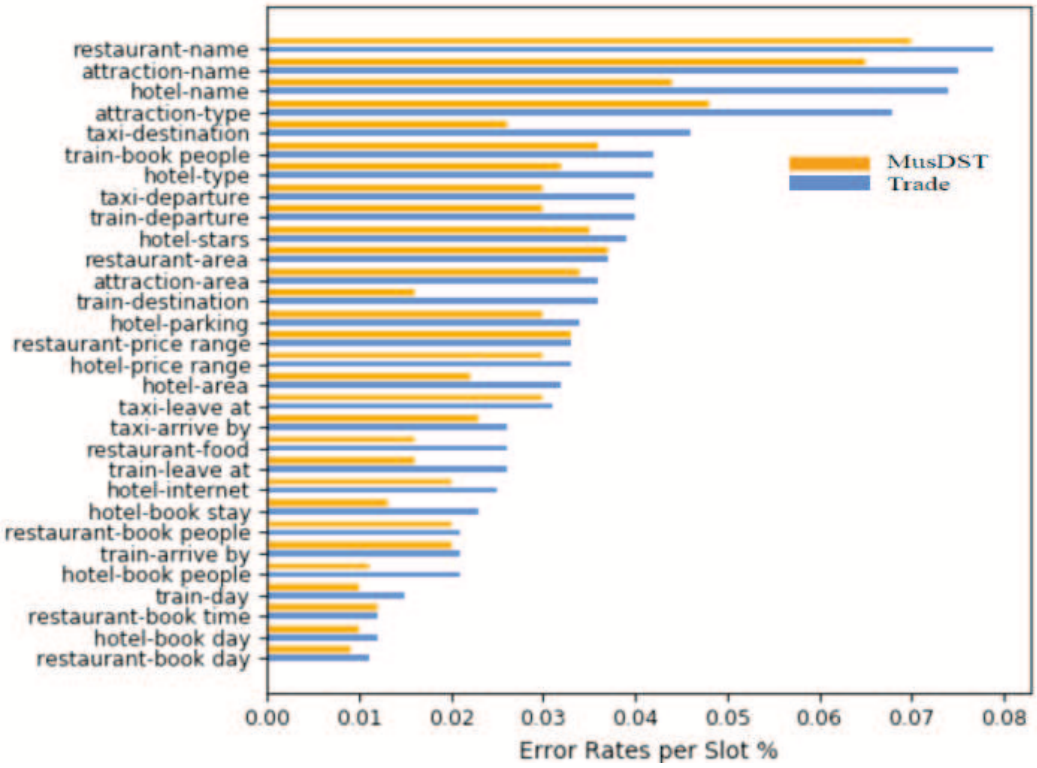

FIGURE 3. Slot-specific accuracy on test set of MultiWOZ 2.1

task with just two candidate values has a high relatively error rate. This is because the labels of (hotel, type) are sometimes absent in the dataset, making our prediction wrong even though it should be correct.

5.4. **Ablation study.** We study and evaluate the effectiveness of different aspects of our model in this section. Ablation study results of MusDST on the test set of MultiWOZ 2.1 are shown in Table 4. In each variant, one part of the model is disabled to show the effect of each part of the model on the final performance.

TABLE 4. Ablation study on MutiWOZ 2.1

Model	Goals (%)	Slots (%)
MusDST	57.49	97.75
-Schema Encoder	56.03	96.92
-Transfer Mechanism	53.52	97.12

Schema Encoder. We experimented with a variant of MusDST by removing the schema encoder module. The performance of the variant without schema encoder drops by 1.46%, demonstrating that the schema encoder module used to model domain relations and slot interactions is important for tracking multi-domain dialogue states.

Transfer Mechanism. We removed our transfer mechanism by removing the "`COPY`" category in the category predictor and the corresponding copy value decoder in the value decoder module. The performance drops by 3.97% which indicates that modeling the dependencies across different slots correctly and obtaining the slot value by copying from related slots can improve the performance of cross-turn inference and slot value extraction.

6. **Conclusions.** We propose a novel multi-step DST model with schema-guided graph convolutional network, which is a new state-of-the-art dialogue state tracker. We introduce a multi-step prediction mechanism to perform dialogue state tracking and extract dialogue states based on the previous states. For the slot values that are not pre-defined in the list or explicitly mentioned in the dialogue history, we propose a transfer mechanism to consider the interaction between different slots and obtain their values through reasoning. Also, the capability of sharing the knowledge between the slots and domains might help a model to work across multiple domains, as well as to handle the slot values that need reasoning. Thus, we introduce GCN to build a schema graph that models the relationship between different domains and slots. Experiments demonstrate that our proposed MusDST achieves state-of-the-art performance on both MultiWOZ 2.0 and MultiWOZ 2.1 datasets.

In this paper, our model tracks the dialogue state in some fixed domains. In future work, we plan to focus on the unseen domain performance. To overcome the data sparsity problem, we would like to try meta-learning techniques and augmentation methods.

REFERENCES

[1] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani and K. Sima'an, Graph convolutional encoders for syntax-aware neural machine translation, *EMNLP*, 2017.

[2] J. Bruna, W. Zaremba, A. D. Szlam and Y. LeCun, Spectral networks and locally connected networks on graphs, *arXiv.org*, arXiv: 1312.6203, 2014.

[3] 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, *EMNLP*, pp.5016-5026, 2018.

[4] G. L. Chao and I. Lane, BERT-DST: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer, *Interspeech*, pp.1468-1472, 2019.

[5] J. Chen, R. Zhang, Y. Mao and J. Xu, Parallel interactive networks for multi-domain dialogue state generation, *arXiv.org*, arXiv: 2009.07616, 2020.

[6] L. Chen, B. Lv, C. Wang, S. Zhu, B. Tan and K. Yu, Schema-guided multi-domain dialogue state tracking with graph attention neural networks, *AAAI*, pp.7521-7528, 2020.

[7] K. Cho, B. V. Merrienboer, D. Bahdanau and Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, *arXiv.org*, arXiv: 1409.1259, 2014.

[8] J. Devlin, M. W. Chang, K. Lee and K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, *arXiv.org*, arXiv: 1810.04805, 2018.

[9] M. Eric, R. Goel, S. Paul, A. Kumar, A. Sethi, A. K. Goyal, P. Ku, S. Agarwal and S. Gao, MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines, *LREC*, 2020.

[10] J. Gao, M. Galley, L. Li et al., Neural approaches to conversational AI, *Foundations and Trends® in Information Retrieval*, vol.13, nos.2-3, pp.127-298, 2019.

[11] S. Gao, A. Sethi, S. Agarwal, T. Chung and D. Hakkani-Tur, Dialog state tracking: A neural reading comprehension approach, *Proc. of the 20th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pp.264-273, 2019.

[12] R. Goel, S. Paul and D. Z. Hakkani-Tür, HyST: A hybrid approach for flexible and accurate dialogue state tracking, *Interspeech*, pp.1458-1462, 2019.

[13] M. Henderson, M. Gašić, B. Thomson, P. Tsiakoulis, K. Yu and S. Young, Discriminative spoken language understanding using word confusion networks, *2012 IEEE Spoken Language Technology Workshop (SLT)*, pp.176-181, 2012.

[14] M. Henderson, B. Thomson and S. Young, Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation, *2014 IEEE Spoken Language Technology Workshop (SLT)*, pp.360-365, 2014.

[15] M. Henderson, B. Thomson and S. Young, Word-based dialog state tracking with recurrent neural networks, *Proc. of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pp.292-299, 2014.

[16] S. Kim, S. Yang, G. Kim and S. W. Lee, Efficient dialogue state tracking by selectively overwriting memory, *ACL*, pp.567-582, 2020.

[17] T. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, *arXiv.org*, arXiv: 1609.02907, 2017.
[18] H. Le, D. Sahoo, C. Liu, N. F. Chen and S. Hoi, UniConv: A unified conversational neural architecture for multi-domain task-oriented dialogues, *arXiv.org*, arXiv: 2004.14307, 2020.
[19] H. Le, R. Socher and S. C. Hoi, Non-autoregressive dialog state tracking, *International Conference on Learning Representations (ICLR)*, pp.146-150, 2019.
[20] C. Lee and B.-D. Lee, Enhancement for automatic extraction of RoIs for bone age assessment based on deep neural networks, *ICIC Express Letters*, vol.14, no.2, pp.163-170, 2020.
[21] W. Lei, X. Jin, M. Y. Kan, Z. Ren, X. He and D. Yin, Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures, *ACL*, pp.1437-1447, 2018.
[22] D. Marcheggiani and I. Titov, Encoding sentences with graph convolutional networks for semantic role labeling, *EMNLP*, 2017.
[23] N. Mrkšić, D. Ó Séaghdha, T. H. Wen, B. Thomson and S. Young, Neural belief tracker: Data-driven dialogue state tracking, *ACL*, pp.1777-1788, 2017.
[24] N. Mrkšić, D. Ó Séaghdha, B. Thomson, M. Gašić, P. H. Su, D. Vandyke, T. H. Wen and S. Young, Multi-domain dialog state tracking using recurrent neural networks, *ACL*, pp.794-799, 2015.
[25] E. Nouri and E. Hosseini-Asl, Toward scalable neural dialogue state tracking model, *The 32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.
[26] Y. Ouyang, M. Chen, X. Dai, Y. Zhao, S. Huang and J. Chen, Dialogue state tracking with explicit slot connection modeling, *ACL*, 2020.
[27] J. Perez and F. Liu, Dialog state tracking, a machine reading approach using memory network, *The European Chapter of the Association for Computational Linguistics (EACL)*, pp.305-314, 2017.
[28] A. Rahimi, T. Cohn and T. Baldwin, Semi-supervised user geolocation via graph convolutional networks, *ACL*, 2018.
[29] L. Ren, K. Xie, L. Chen and K. Yu, Towards universal dialogue state tracking, *ACL*, pp.2780-2786, 2018.
[30] F. Scarselli, M. Gori, A. Tsoi, M. Hagenbuchner and G. Monfardini, The graph neural network model, *IEEE Transactions on Neural Networks*, vol.20, pp.61-80, 2009.
[31] 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.
[32] Z. Wang and O. Lemon, A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information, *Proc. of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pp.423-432, 2013.
[33] T. H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. Rojas-Barahona, P. H. Su, S. Ultes and S. Young, A network-based end-to-end trainable task-oriented dialogue system, *Proc. of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp.438-449, 2017.
[34] J. D. Williams, Web-style ranking and SLU combination for dialog state tracking, *Proc. of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pp.282-291, 2014.
[35] 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, *ACL*, pp.808-819, 2019.
[36] P. Xu and Q. Hu, An end-to-end approach for handling unknown slot values in dialogue state tracking, *ACL*, pp.1448-1457, 2018.
[37] S. Young, M. Gašić, B. Thomson and J. D. Williams, POMDP-based statistical spoken dialog systems: A review, *Proc. of the IEEE*, vol.101, no.5, pp.1160-1179, 2013.
[38] J. G. Zhang, K. Hashimoto, C. S. Wu, Y. Wan, P. S. Yu, R. Socher and C. Xiong, Find or classify? Dual strategy for slot-value predictions on multi-domain dialog state tracking, *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
[39] X. Zhang, X. Zhao and T. Tan, Robust dialog state tracker with contextual-feature augmentation, *Applied Intelligence*, vol.51, no.4, pp.2377-2392, 2021.
[40] V. Zhong, C. Xiong and R. Socher, Global-locally self-attentive encoder for dialogue state tracking, *ACL*, pp.1458-1467, 2018.
[41] L. Zhou and K. Small, Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering, *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
[42] L. Zilka and F. Jurcicek, Incremental LSTM-based dialog state tracker, *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp.757-762, 2015.

Appendix A. Dialogue Example with DST Label

FIGURE 4. (color online) A dialogue example with DST label. The turns are separated by dashed lines. Each turn consists of a user utterance (green), a system response (pink), a turn state (purple) and a joint goal (blue).

Author Biography

Xuejun Zhang received the B.S. degree in electronic communication engineering from North China Institute of Science and Technology, Hebei, China, in 2012 and the M.S. degree in communication engineering from Beijing Institute of Technology, Beijing, China, in 2015. She is currently pursuing the Ph.D. degree in natural language processing from Institute of Acoustics, Chinese Academy of Sciences (CAS) in 2012, Beijing, China. From 2015 to 2018, he was a Research Assistant with the Institute of Acoustics, Chinese Academy of Sciences (CAS), Beijing, China. She used to study cross-lingual semantic representation and conversational text classification. Her current research interest lies in spoken language understanding, dialog state tracking, and dialogue decision.

Pengyuan Zhang received the Ph.D. in speech signal processing from Institute of Acoustics, Chinese Academy of Sciences (CAS) in 2007, Beijing, China. From 2010 to 2016, he was an associate researcher in the Institute of Acoustics, Chinese Academy of Sciences and became a researcher since 2017. His research interest includes speech recognition and understanding, speech synthesis, emotion recognition and speech signal processing. He has been responsible for many national scientific research projects and published several papers and patents.

Yonghong Yan has been engaged in speech research for nearly 30 years, published more than 300 academic papers and held more than 100 invention patents (including 10 American invention patents). He ranked first and won the title of "excellent" in the final assessment conducted by Chinese Academy of Sciences in 2006. In 2007, he was selected as a national candidate for the new century hundred million talent project, and won the excellent teacher award of the Chinese Academy of Sciences in 2008. He won the national outstanding youth fund of the National Natural Science Foundation of China (the first person in the speech industry) in 2009. In 2010, he won the Ma Dayou award of the Chinese acoustic society, and won two first prizes of scientific and technological progress of Xinjiang Autonomous Region in 2013 (ranking first and second). In 2014, he won the outstanding scientific and technological achievement award of the Chinese Academy of Sciences. He won the second prize of Beijing scientific and technological progress award in 2019.