BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 04/06 pp41–52
Volume 5, Number 1, March 2022
DOI: 10.26599/BDMA.2021.9020013

# Exploiting More Associations Between Slots for Multi-Domain Dialog State Tracking

Hui Bai, Yan Yang*, and Jie Wang

**Abstract:** Dialog State Tracking (DST) aims to extract the current state from the conversation and plays an important role in dialog systems. Existing methods usually predict the value of each slot independently and do not consider the correlations among slots, which will exacerbate the data sparsity problem because of the increased number of candidate values. In this paper, we propose a multi-domain DST model that integrates slot-relevant information. In particular, certain connections may exist among slots in different domains, and their corresponding values can be obtained through explicit or implicit reasoning. Therefore, we use the graph adjacency matrix to determine the correlation between slots, so that the slots can incorporate more slot-value transformer information. Experimental results show that our approach has performed well on the Multi-domain Wizard-Of-Oz (MultiWOZ) 2.0 and MultiWOZ2.1 datasets, demonstrating the effectiveness and necessity of incorporating slot-relevant information.

**Key words:** slot-relevant attention; multi-domain dialog state tracking; task-oriented dialog system

## 1 Introduction

Dialg State Tracker (DST) is an indispensable component in task-oriented dialog systems. Effective and accurate DST performance is essential for the update of the internal state of dialog systems and the formulation of dialog strategies. As a language understanding task of contextual knowledge[1], DST aims to extract users'goals and intentions in each turn of a dialog according to the dialogue context, and represent them as a compact dialog state, which is a set of slot-value pairs. The dialog state can provide a basis for selection of system actions and responses. Although the dialog system for specific tasks is constantly developing, with the diversification of user needs and the gradual increase in the complexity of user goals, ordinary single-domain dialog systems can no longer meet the needs of users. Accordingly, Budzianowski et al.[2] proposed a new challenge for the DST task, and called it multi-domain DST. This challenge allows users to conduct dialog in multiple domains with the system, and is no longer limited to dialogs around a single domain to complete certain requirements. Table 1 shows the process of a user booking a hotel first and then booking a taxi.

The commonly used DST method[3,4] assumes that all slot-value pair candidate values have been given in advance, and they rely on a predefined ontology. This method completes the DST task by scoring all possible slot-value pairs in the ontology and selecting the highest score as the predicted value of the slot. The advantage of the predefined ontology-based method is to simplify the DST task into a classification task to improve performance, but these methods may be unsuitable in practical situations. A complete ontology is difficult to be obtained in the real world, and even if there is a complete ontology, the number of domains-slots in the ontology may be huge and variable[4]. Its actual use range is relatively small, and is suitable for a small number of domains and can enumerate all slot-value dialog scenarios. To overcome these issues, recent studies have focused on open-vocabulary methods. This type of method has no predefined ontology, and most studies have used an encoder-decoder structure and copy

• Hui Bai, Yan Yang, and Jie Wang are with the School of Computing and Artifical Intelligence, Southwest Jiaotong University, Chengdu 611756, China. E-mail: baihui@my.swjtu.edu.cn; yyang@swjtu.edu.cn; JackWang@my.swjtu.edu.cn.

* To whom correspondence should be addressed.

Manuscript received: 2021-07-07; accepted: 2021-07-19

mechanism to directly generate or extract a value from the dialog context for each slot. The advantage of this method is that it breaks through the limitations of the ontology and can track slots in unknown areas. Therefore, the model proposed in this paper will also be based on the open-vocabulary method.

The goal of multi-domain DST is to predict the value of each domain-slot pair based on the dialog history and the state of the previous turn of dialog. However, with the increase in the number of dialog turns and domains, the number of domain-slot pairs and their candidate values are also increasing. If the DST method is still generating a slot value for each slot individually[5–7], it will exacerbate the data sparsity problem and is not conducive to model training. To alleviate this problem, we should further examine slot-relevant information to enhance the inferential capability of DST. If two slots are related, then they are likely to share slot values in a dialog. The relevant information is helpful for slots to learn implicit connections. Taking the dialog shown in Table 1 as an example, the user's goal is to book a taxi to the hotel. The semantics implied in this case is that the taxi destination should be the hotel-name. DST should learn the ability to infer implicit connections between dialogs. In this dialog, the relevant information of "taxi-destination" is the "hotel-name". Therefore, the slots "taxi-destination" and "hotel-name" should share the same value, i.e., "warkworth house".

Inspired by learning more slot-related information for each slot, in this paper, we propose a multi-domain DST model that uses graphs to mine relevant information between slots. To alleviate the data sparsity problem and enhance the ability of implicit connection in the model reasoning dialog, we use a stacked Integrate Relevant Information (IRI) module to incorporate the associated information between each domains-slots after feature extraction. In this way, each slot not only learns the information in the context of the dialog, but also learns from the relevant slots. Finally, the soft-gated copy mechanism is used to extract the value from the dialogue context or generate it from the vocabulary to improve the model's ability to generate unknown slot values. The contributions of this work are summarized as follows:

**Table 1 Example of multi-domain dialogs. The "hotel-name" and "taxi-destination" in red indicate that they have an implicit relationship and should share the same value "warkworth house".**

| |
|---|
| **User:** I need to book a hotel in the east that has 4 stars.<br>**State:** hotel-area=east, hotel-stars=4. |
| **System:** I would recommend the warkworth house.<br>**User:** Ok. Could you book it for 1 night , 1 person on Friday?<br>**State:** hotel-area=east, hotel-stars=4, hotel-name=warkworth house, hotel-book stay, hotel-book people=1, hotel-book day=1. |
| **System:**Booking is successful! Can I help you with anything else today?<br>**User:** I also need a taxi to get me to the hotel.<br>**State:** hotel-area=east, hotel-stars=4, hotel-name=warkworth house, hotel-book stay, hotel-book people=1, hotel-book day=1, taxi-destination=warkworth house. |

- We propose a multi-domain DST model that integrates slot-relevant information to fully exploit implicit reasoning relationships between dialogs.
- We design an IRI module that uses a graph adjacency matrix to model the information association between slots.
- Experimental results show that our method has achieved a good performance. Our ablation studies can confirm that incorporating slot relevant information is important.

The rest of this paper is organized as follows. In Section 2, we discuss the previous research about predefined ontology based DST and open vocabulary DST. In Section 3, we present the relevant definition of the multi-domain DST task and introduce our proposed model. In Section 4, we provide our experiments and the results on two benchmark datasets. Finally, in Section 5, we discuss the conclusions of our work.

## 2 Related Work

The early research on DST[8–10] relied on the semantics extracted by the Natural Language Understanding (NLU) module. However, because the errors generated by the NLU module can be accumulated and transferred to DST, it is unreasonable to train the two modules separately. To solve this issue, later studies focused on a joint training method[3, 11, 12]. With the development of deep learning, Mrkšić et al.[13] proposed a novel neural belief tracking framework which can learn the distributed representation of the dialog context using pretrained word vectors. In recent years, considerable research has mainly focused on predefined ontology based and open-vocabulary DST.

**Predefined ontology DST:** Eric et al.[14] proposed the FJST model that uses a bidirectional Long Short-Term Memory (LSTM) network[15] to encode dialog history and a feedforward neural network to select the value of each slot. They also put forward the HJST model. Its encoding stage is similar to FJST, but it uses a hierarchical network. The disadvantage of this type of method is that it cannot easily add a new slot, which makes the ontology setting lack flexibility. Accordingly, the SUMBT model[16] uses a contextual semantic encoder to learn the relationship between the domain-slot type and the slot value that appears in the dialog, and then uses nonparametric distance to predict the value of each slot. Zhou and Small[17] modeled multi-domain DST as a question-and-answer problem with a dynamically evolving knowledge graph and proposed the DSTQA model, which heavily relies on predefined ontology. The model abstracts DST as a process of answering one question to another. It generates a question and then asks the value of each (domain, slot) pair.

By modeling the relationship between slots, Chen et al.[18] expressed slots in different domains and the relationship between them as pattern graphs, and proposed the SST model. It uses a graph attention matching network to fuse utterances and schema graph and a recurrent graph attention network to control state updates. Shan et al.[19] designed the CHAN-DST model, which uses a context hierarchical attention network based on Bidirectional Encoder Representations from Transformers (BERT)[20], and introduced an adaptive target that dynamically adjusts the weights of different slots during the training process to relieve the slot imbalance problem.

**Open-vocabulary DST:** Similar to the DSTQA model, inspired by reading comprehension tasks, Gao et al.[21] modeled the dialog state tracking task as a reading comprehension task, and proposed the DST-Reader model. It expresses the DST problem as an extractive question-and-answer task, and uses BERT to contextualize the word vector and extract the slot value from the input using the predicted span. Zhang et al.[22] put forward the DST-SPAN model which uses BERT as the encoder and a problem-solving method similar to the DST-Reader model.

The TRADE[23] model uses a bidirectional Gated Recurrent Unit (GRU) to encode the dialog history and a GRU[24] decoder combined with the copy mechanism to decode the value of each state to obtain the slot value. Considering the serious computational complexity of multi-domain DST relying on predefined ontology, Ren et al.[25] proposed the COMER model. Given each turn of user utterance and system response, the model can generate a series of belief states using a hierarchical encoder-decoder structure. This design makes the computational complexity of the model constant. Le et al.[26] designed a new non-autoregressive DST framework, which uses a transformer-based non-autoregressive decoder to generate the current state. The non-autoregressive nature can not only significantly reduce the waiting time of DST for real-time dialog response generation, but also detect dependencies between slots other than the token level.

Kim et al.[27] regarded the dialog state as an explicit fixed-size memory and advanced a selective coverage mechanism to improve the efficiency of DST. This model divides DST into two subtasks: state operation prediction and slot value generation. State operation prediction classifies the encoder output representation into four categories, and only decodes the slots classified as UPDATE to generate values. This method of dividing into two subtasks has gradually become the mainstream framework of the open-vocabulary method. To increase the model's encoding capabilities, Zhu et al.[28] proposed the Context and Schema Fusion Network (CSFN)-DST model that employs a fusion network and graph-based internal and external attention mechanisms.

## 3 Method

In this section, we describe our proposed model. The whole model is an encoder-decoder architecture. The encoder is mainly composed of four parts: feature extraction layer, domain classifier module, IRI module, and operation classifier module. The decoder contains a value generator module. The overall architecture of the model is shown in Fig. 1.

### 3.1 Problem statement

In the multi-domain DST task, we assume that there are $M$ domains involved $D = \{d_1, d_2, \dots, d_M\}$. Each domain $d \in D$ has a set of slots $S^d = \{s_1^d, s_2^d, \dots, s_j^d\}$. The number of slots in each domain is not fixed, so $j$ is also variable. Each domain and its slots form a set of domain-slot pair $P = \{P_1, P_2, \dots, P_J\}$, where $J$ is the total number of domain-slot pairs that are not repeated. For example, in the Multi-domain Wizard-Of-Oz (MultiWOZ) dataset, there are 5 domains, 17 slots, and 30 domain-slot pairs.

A dialog with $T$ turns can be represented as $\{(D_1, B_1), (D_2, B_2), \dots, (D_T, B_T)\}$, where $D_t$ is the combination of the $t$-th turn of system response and user utterance, and $B_t$ represents the corresponding dialog state. $D_t$ is a token sequence, and $B_t$ is a domain-slot value tuple, e.g., train-day-Friday.

The goal of the multi-domain DST task is to provide

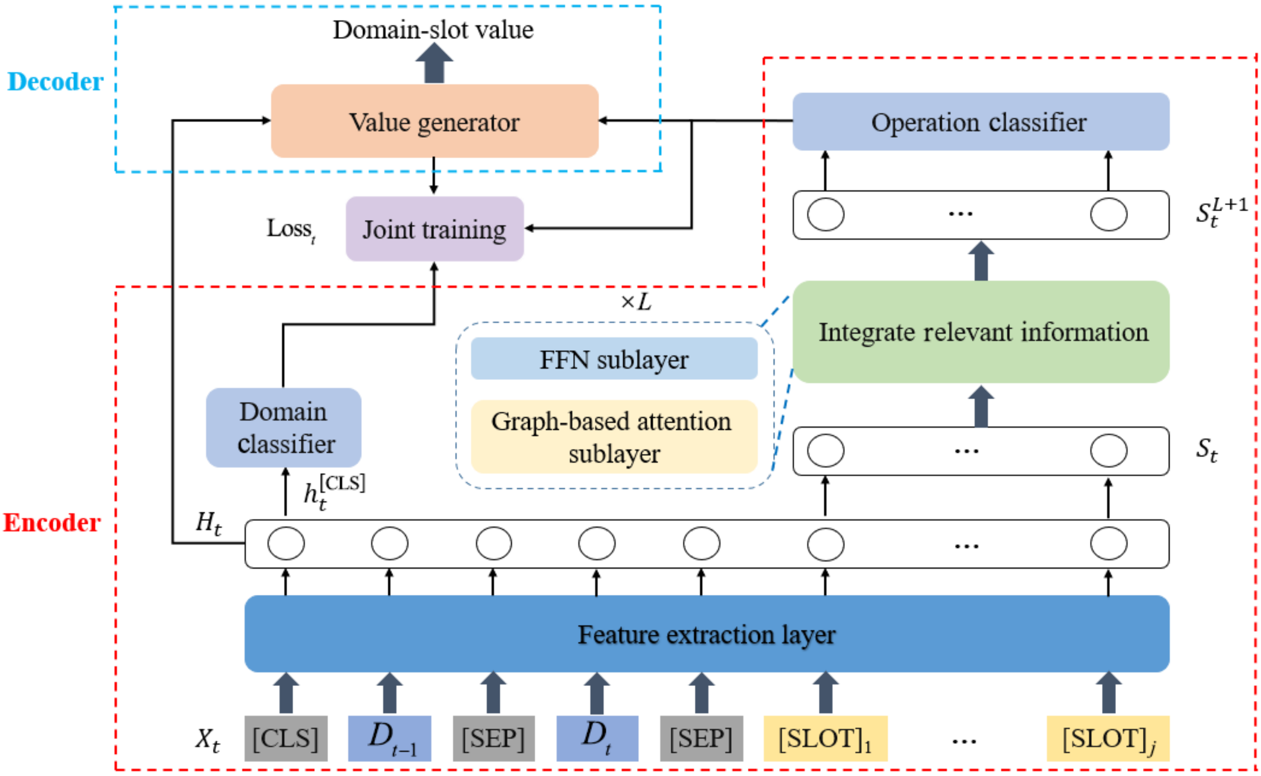

**Fig. 1 Architecture of the proposed model.**

the dialog history and the dialog state of the previous turn $\{(D_1, B_1), (D_2, B_2), \ldots, (D_{t-1}, B_{t-1}), (D_t)\}$, and then predict the current dialog state $B_t$.

## 3.2 Encoder

Kim et al.[27] demonstrated the influence of dialog history on DST performance, and most turns of a dialog state depend on the previous turn of the dialog history. Thus, in this study, we only use the previous turn of the dialog history $D_{t-1}$, $B_{t-1}$, and the current turn dialog $D_t$ to predict the current corresponding dialog state $B_t$. We concatenate the three elements as the model input.

### 3.2.1 Feature extraction layer

To obtain the context representation from the dialog history and dialog state, we embed the input sequences as vectors and use BERT to extract the latent features according to the embedding representation.

**Input representation:** We denote the $t$-th turn dialog $D_t$ as $D_t = R_t \oplus ; \oplus U_t$, where $R_t$ is the system response and $U_t$ is the user utterance. [SEP] is a special token used to separate different conversation turns. The previous dialog state $B_t$ contains all (domain, slot, value) tuples, which are denoted as

$$[\text{SLOT}] \oplus \text{domain} \oplus - \oplus \text{slot} \oplus - \oplus \text{value}.$$

The function of [SLOT] is similar to that of [CLS] in BERT, and it can be used as the information representation of aggregated (domain, slot, and value) tuples. "–" is used to a separate token, and "⊕" is used to concatenate two elements. For tuples with actual values, we fill in the corresponding value, otherwise, fill in [NULL], which means the value is none. We employ the pretrained BERT model to extract features, so its input is

$$X_t = [\text{CLS}] \oplus D_{t-1} \oplus [\text{SEP}] \oplus D_t \oplus [\text{SEP}] \oplus B_{t-1}.$$

Due to the input requirements of BERT, the input embeddings of $X_t$ are the sum of the token, segmentation, and position embeddings.

**Output representation:** The output of BERT after extracting features from $X_t$ is $H_t \in \mathbf{R}^{|X_t| \times d}$. The special tokens [CLS] and [SLOT] are expressed as $h_t^{[\text{CLS}]} \in \mathbf{R}^d$ and $h_t^{[\text{SLOT}]} \in \mathbf{R}^d$, respectively. $d$ is the hidden size of the model.

### 3.2.2 Domain classifier

Because [CLS] can be regarded as the information aggregation representation of the entire input sequence, it can be used as a basis for evaluating the domain involved in the current dialog turn. Domain classification can help the model learn the correlation information between slots and domains in dialog turns, and improve the accuracy of generating slot values for domain-related slots. It maps $h_t^{[\mathrm{CLS}]}$ to a linear space of domain numbers and uses softmax to calculate the most likely domain of the current turn,

$$P_t^{dom} = \mathrm{softmax}(W_{dom} h_t^{[\mathrm{CLS}]}) \tag{1}$$

where $W_{dom} \in \mathbf{R}^{d_{dom} \times d}$ is a learnable linear transformation parameter, and $d_{dom}$ is the total number of domains. $P_t^{dom} \in \mathbf{R}^d$ is the probability distribution of all domains in the $t$-th turn. Therefore, the predicted domain of the $t$-th turn of a dialog is

$$\hat{dom}_t = \arg\max_{dom} P_t^{dom} \tag{2}$$

### 3.2.3 Integrate relevant information

Because the location of [SLOT] incorporates the information of the domain-slot-value, we can regard $h_t^{[\mathrm{SLOT}]_j}$ as the feature representation of the $j$-th domain-slot pair (the following will be abbreviated as “slot”) in the $t$-th turn of dialog. To integrate the relevant information into the slot and improve its ability to learn implicit connections in dialog turns, we design the IRI module according to the idea of the transformer. This module stacks several identical layers, and each layer includes two sub-layers: the first is used to learn the relevant information between slots, and the second is used to provide non-linear learning capabilities. To facilitate training, we concatenate the representations of all [SLOT] positions into a new input vector,

$$S_t = [h_t^{[\mathrm{SLOT}]_1}, h_t^{[\mathrm{SLOT}]_2}, \ldots, h_t^{[\mathrm{SLOT}]_J}] \tag{3}$$

where $S_t \in \mathbf{R}^{d \times J}$.

**Graph-based attention sublayer:** Multi-head attention[29] can be described as mapping a query and a set of key-value pairs to an output. To incorporate the relevant information into the multi-head attention, we use the graph $G = (N, E)$ to represent the associations between slots. In the graph, a node means the slot, and the edge represents whether two nodes are related. An example is shown in Fig. 2. We employ the adjacency matrix $M \in \mathbf{R}^{|J| \times |J|}$ to represent the graph, and $M_{ij} \in \{0, 1\}$. In this matrix, if $M_{ij} = 1$, then there is relevant information between the $i$-th and $j$-th slots.

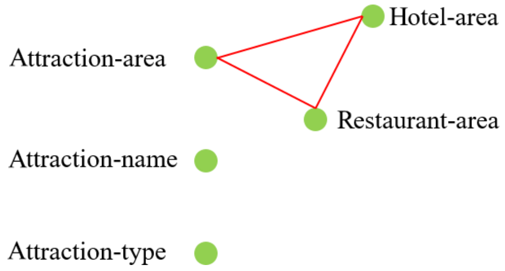

**Fig. 2 Example of the graph describing associations between nodes. Nodes are in green and edges are in red.**

Because we need to use multi-head attention to calculate the relevant information between slots, we assume that there are three vectors of query $Q = \{q_i\}_{i=1}^{|Q|}$, key $K = \{k_j\}_{j=1}^{|K|}$, and value $V = \{v_k\}_{k=1}^{|V|}$, where $q_i, k_j$, and $v_k \in \mathbf{R}^{1 \times d}$. Actually, the three vectors are the representations of all slots. The attention-weighted calculation of the query vector $q_i$ is as follows:

$$e_{ij}^{(h)} = \begin{cases} \dfrac{(q_i W_Q^{(h)})(k_j W_K^{(h)})^{\mathrm{T}}}{\sqrt{d_k/H}}, & \text{if } M_{ij} = 1; \\ -\infty, & \text{otherwise} \end{cases} \tag{4}$$

$$a_{ij}^{(h)} = \frac{\exp(e_{ij}^{(h)})}{\sum\limits_{n=1}^{|K|} \exp(e_{in}^{(h)})} \tag{5}$$

$$c_i^{(h)} = \sum_{k=1}^{|V|} a_{ik}^{(h)} \cdot (v_k W_V^{(h)}) \tag{6}$$

$$c_i^H = [c_i^1, c_i^2, \ldots, c_i^H] W_O \tag{7}$$

where $W_Q^{(h)}$, $W_K^{(h)}$, and $W_V^{(h)} \in \mathbf{R}^{d \times (d/H)}$ are different learnable matrices, $h$ is the attention head, and $H$ is the total number of $h$. $c_i^H$ brings together the representation of $H$ attention heads at the $i$-th position in the query vector, and $W_O \in \mathbf{R}^{d \times d}$ is the parameter matrix. The entire process is denoted as GraphAtt( ),

$$C = \mathrm{GraphAtt}(\mathrm{query}, \mathrm{key}, \mathrm{value}, M) \tag{8}$$

where $C$ is the result of the query vector that incorporates the graph-based attention weights. By learning on multiple attention heads, the query vector finally incorporates more important information from key and value vectors.

Therefore, to incorporate relevant information into $S_t$, its update in this sublayer is as follows:

$$C_t^l = \mathrm{GraphAtt}(S_t^l, S_t^l, S_t^l, M) + S_t^l \tag{9}$$

where $l$ is the update times and $S_t^l$ is the $l$-th update of $S_t$. Because the adjacency matrix $M$ contains all possible connections between slots, the vectors of the query, key, and value which calculated by GraphAtt() are all the same.

**FFN sublayer:** This sublayer can make the slots learn more nonlinear information and help parameter update,

$$S_t^l = FFN(C_t^l) + C_t^l,$$
$$FFN(y) = W_2 ReLu(W_1 y + b_1) + b_2 \tag{10}$$

where $W_1$ and $W_2 \in \mathbf{R}^{d \times d}$, $b_1$ and $b_2 \in \mathbf{R}^d$. The rectified linear unit[30] is an activation function.

In addition, each sublayer will add a LayerNorm[31] before being updated. After updating for $L$ layers,

$S_t^{L+1}=\{s_{t,1}^{L+1}, s_{t,2}^{L+1}, \ldots, s_{t,J}^{L+1}\}$ has learned slot relevant information for each slot as much as possible.

### 3.2.4 Operation classifier

In a dialog turn, only a few slots need to generate new values. Therefore, we use a four-way classifier to select the operation to be performed for each slot. The operation of the four-way classifier is {UPDATE, DELETE, DONTCARE, CARRYOVER}. UPDATE means that the slot needs to generate a new value, DELETE means that the slot's corresponding value is none, DONTCARE means that the slot's value could not be tracked, and CARRYOVER means that the slot's value is not changed. We add a linear layer to $S_t^{L+1}$ in the previous section to classify each slot operation,

$$P_{op,t}^{j}=\operatorname{softmax}(W_{op} s_{t,j}^{L+1}) \tag{11}$$

where $W_{op} \in \mathbf{R}^{4\times d}$. The prediction operation for the $j$-th slot is

$$\hat{op}_{j,t}=\arg\max_{op} P_{op,t}^{j} \tag{12}$$

## 3.3 Decoder

According to the content in the previous section, the model generates the value for the slot, whose operation is UPDATE. Through the analysis of the dialog turns, we know that some values are derived from the dialog inputs, and some are derived from the vocabulary. Therefore, we use a combination of the GRU decoder and soft-gated copy mechanism[32,33] to generate the corresponding value. The soft-gated copy mechanism combines the word distribution of the source input sequence and the vocabulary into an overall word probability distribution, and selects the word with the highest probability as the decoded word.

The initialization of the GRU is

$$\begin{aligned} x_t^{j,0} &= s_{t,j}^{L+1}, \\ g_t^{j,0} &= h_t^{[\mathrm{CLS}]} \end{aligned} \tag{13}$$

where $x_t^{j,0}$ is the word embedding of the 0-th decoding step of the $j$-th slot in the $t$-th dialog turn, and $g_t^{j,0}$ is the hidden state. Then the updated hidden state of the GRU is

$$g_t^{j,k}=\mathrm{GRU}(x_t^{j,k}, g_t^{j,k-1}) \tag{14}$$

The hidden state decoded by the GRU can be used to calculate the probability distribution of generating a word from the vocabulary $P_{vocab}^{j,k}$, and the probability distribution of copying a word from the input sequence $P_{input}^{j,k}$ is as follows:

$$\begin{aligned} P_{vocab,t}^{j,k} &= \operatorname{softmax}(E_{vocab} \cdot g_t^{j,k}), \\ P_{input,t}^{j,k} &= \operatorname{softmax}(H_t \cdot g_t^{j,k}) \end{aligned} \tag{15}$$

where $P_{vocab,t}^{j,k} \in \mathbf{R}^{d_{vocab}}$ and $P_{input,t}^{j,k} \in \mathbf{R}^{|X_t|}$. $E_{vocab} \in \mathbf{R}^{d_{vocab}\times d}$ is the vocabulary embedding matrix, and $d_{\mathrm{vocab}}$ is the vocabulary size.

Then, the soft-gated copy mechanism is used to calculate the final distribution through the two distributions,

$$P_{fianl,t}^{j,k}=p_{gen,t}^{j,k}\times P_{vocab,t}^{j,k}+(1-p_{gen,t}^{j,k})\times P_{input,t}^{j,k} \tag{16}$$

where $P_{fianl,t}^{j,k} \in \mathbf{R}^{d_{vocab}}$ is the final distribution in the vocabulary. The operation of mapping $P_{input}^{j,k}$ to a linear space of the vocabulary is omitted here. It aims to achieve the same dimension of two distributions. $p_{gen,t}^{j,k}$ is the gate threshold of generating a word and is calculated as follows:

$$p_{gen,t}^{j,k}=\operatorname{sigmoid}(W_1\times[g_t^{j,k}; x_t^{j,k}; c_t^{j,k}]) \tag{17}$$

where $W_1 \in \mathbf{R}^{1\times 3d}$ is a learnable linear transformation parameter, and $c_t^{j,k} \in \mathbf{R}^{d}$ is the context vector calculated by $c_t^{j,k}=P_{input}^{j,k}\cdot H_t$.

Finally, the model's $k$-th generated word in the $j$-th slot of the $t$-th dialog turn is the vocabulary word with maximum probability,

$$\hat{y}_{t,j}^{value,k}=\arg\max P_{fianl,t}^{j,k} \tag{18}$$

## 3.4 Optimization

During training, we optimize the domain classifier, operation classifier, and value generator,

$$\begin{aligned} \mathrm{loss}_{dom,t} &= -Y_{dom,t}\log(P_{dom,t}), \\ \mathrm{loss}_{op,t}^{j} &= -\sum_{j=1}^{J} Y_{op,t}^{j}\log(P_{op,t}^{j}), \\ \mathrm{loss}_{gen,t}^{j} &= -\sum_{j=1}^{J}\sum_{k=1}^{|N_j|} Y_{gen,t}^{j,k}\log(P_{final,t}^{j,k}) \end{aligned} \tag{19}$$

where $Y_{dom,t}$, $Y_{op,t}^{j}$, and $Y_{gen,t}$ are the ground-truths of the corresponding domain, operation, and value, and $N_j$ is the number of value tokens in the $j$-th slot. The final loss of the model at $t$-th dialog turn is

$$\mathrm{loss}_t=\mathrm{loss}_{dom,t}+\mathrm{loss}_{op,t}+\mathrm{loss}_{gen,t} \tag{20}$$

# 4 Experiment

## 4.1 Datasets and evaluation metric

The MultiWOZ dataset is a series of human-machine dialog sets that span multiple domains and topics. We use two public datasets, i.e., MultiWOZ2.0[2] and MultiWOZ2.1[14], to experiment with our model. The latter is a corrected version of the former. The dataset includes a total of seven domains, but since there is no dialogue from the hospital and police domains in the validation set and test set, we use five of the seven domains for training and testing. The preprocessed dataset contains 5 domains, 17 slots, 30 domain-slot pairs, and more than 4500 different values. The dataset statistics are shown in Table 2.

We use the joint goal accuracy[34] to evaluate the performance of the model. It is used to measure whether the user's goals are fully satisfied in each turn of the dialog. The joint goal has been achieved only when the predicted dialog states entirely match the ground truth. Thus, it is the most important metric to evaluate the DST performance.

### 4.2 Training details

We use BERT as the encoder which has 12 layers with 768 hidden units and 12 self-attention heads. The maximum length of the input is 256, and we set the batch size to 32 and the training epoch to 30. In the IRI module, we set the number of update layers to 4 and the attention heads to 8. We employ Adam as the optimizer. We set the learning rate to $1\times10^{-4}$ and the warmup proportion to 0.1. During training, we use the ground truth of the previous dialog state.

### 4.3 Baseline models

We compare our model with the following previous methods:

HJST[14] uses a bidirectional LSTM to encode the dialog history and a feedforward network to select the value of each slot.

FJST[14] encodes the dialog history using an LSTM, like FJST, but utilizes a hierarchical network.

**Table 2 Dataset statistics of MultiWOZ.**

| Domain | Slot | Train | Validation | Test |
|---|---|---|---|---|
| Attraction | Area, name, type | 2717 | 401 | 395 |
| Hotel | Area, day, people, stay, internet name, parking, price, stars, type | 3381 | 406 | 394 |
| Restaurant | Area, day, book people, time, food, name, price | 3813 | 438 | 437 |
| Taxi | Arriveby, departure, destination, leaveat | 1654 | 207 | 195 |
| Train | Arriveby, people, day, departure, destination, leaveat | 3103 | 484 | 494 |

SUMBT[16] uses BERT to initialize the encoder, and then nonparametric distance is used to predict the value of each slot.

DS-DST[22] uses two BERT encoders and designs a hybrid method for predefined ontology based DST and open-vocabulary DST.

DST-Picklist[22] uses a similar architecture to DS-DST, but it only performs predefined ontology based DST by treating all slots as picklist-based slots.

DSTQA[17] is abstracted as a question-and-answer problem. It generates a question asking for the value of each domain-slot pair.

SST[18] uses a graph attention matching network and recurrent graph attention network to fully encode dialog utterances and schema graphs.

CHAN-DST[19] employs a contextual hierarchical attention network to enhance the DST, and uses an adaptive objective to alleviate the slot imbalance problem.

DST-Reader[21] formulates the problem of DST as an extractive question-and-answer task, uses BERT contextualized word embeddings, and extracts slot values from the input by predicting spans.

DST-SPAN[22] uses BERT as the encoder, and uses the question-and-answer method which is similar to DST-Reader.

TRADE[23] employs a bidirectional GRU to encode the dialog history and uses a copy-based GRU decoder to decode the value for each slot.

COMER[25]uses BERT-large as an encoder and a hierarchical LSTM decoder.

NADST[26] applies a transformer-based non-autoregressive decoder to generate the current dialog state.

SAS[34] uses slot attention and slot information sharing to enhance the expressiveness of the information passed to each slot.

SOM-DST[27] applies a predictor generator framework. It uses BERT to encode the previous and current dialog utterances and the dialog state.

CSFN-DST[28] uses a schema graph to consider the relations among domains and slots. It uses the BERT encoder and a copy-based RNN decoder.

### 4.4 Experimental results

#### 4.4.1 Joint goal accuracy

We first show the joint goal accuracy of our model and the baselines on the MultiWOZ2.0 and MultiWOZ2.1 datasets in Table 3. Our model belongs to the

**Table 3 Joint goal accuracy on MultiWOZ.**

(%)

| Method | Model | MultiWOZ2.0 | MultiWOZ2.1 |
|---|---|---|---|
| Predefined ontology | HJST | 38.40 | 35.55 |
| | FJST | 40.02 | 38.00 |
| | SUMBT | 42.40 | – |
| | DS-DST | – | 51.21 |
| | DST-Plicklist | – | 53.30 |
| | DSTQA | 51.44 | 51.17 |
| | SST | 51.17 | 55.23 |
| | CHAN-DST | **52.68** | **58.55** |
| Open-vocabulary | DST-Reader | 39.41 | 36.40 |
| | DST-SPAN | 42.59 | 40.00 |
| | TRADE | 48.60 | 45.60 |
| | COMER | 48.79 | – |
| | NADST | 50.52 | 49.04 |
| | SAS | 51.03 | – |
| | SOM-DST | 51.38 | 52.57 |
| | CSFN-DST | 51.57 | 52.88 |
| | **Ours** | **52.26** | **53.59** |

open-vocabulary method. Table 3 illustrates that our model achieves the highest joint goal accuracy in the open-vocabulary DST: 52.26% in MultiWOZ2.0 and 53.59% in MultiWOZ2.1. The accuracy of this model in MultiWOZ2.0 is generally lower than that in MultiWOZ2.1 because the latter fixes some notation errors in the former. Therefore, these models can obtain more useful context information when encoding dialog inputs.

Compared with the predefined ontology based DST, the accuracy of our model is lower than that of SST and CHAN-DST. Because the predefined ontology based DST essentially functions to classify the correct slot value, normally, the accuracy of the open-vocabulary model is lower than that of the predefined method. Although the performance of the open-vocabulary method is slightly low, it does not need to know all the slot value information in advance. Thus, it can solve the problem of generating unknown words and is more suitable for real scenes.

### 4.4.2 Domain-specific accuracy

In Table 4, we show the domain-specific accuracy for each domain, which is the accuracy of using joint goal accuracy to measure the model performance on a single domain.

The results show that our model has achieved the highest score in hotel, restaurant, and train domains. Through the analysis, we found that the reason why the performance of our model is lower than SOM-DST in the attraction and taxi domains is that the amount of data in this two domains is less than that in the above mentioned three domains. This condition makes our model lack the ability to learn slot-relevant information in the two domains during training. From Table 2, we can also see that the hotel domain has the largest number of slots, hence making the accuracy of the hotel domain relatively low.

**Table 4 Domain-specific accuracy on MultiWOZ2.1.**

(%)

| Model | Domain | | | | |
|---|---|---|---|---|---|
| | Attraction | Hotel | Restaurant | Taxi | Train |
| NADST | 66.83 | 48.76 | 65.37 | 33.80 | 62.36 |
| SOM-DST | **69.83** | 49.53 | 65.72 | **59.96** | 70.36 |
| Ours | 68.63 | **50.92** | **66.03** | 59.03 | **71.82** |

## 4.5 Case analysis

Figure 3 shows an example of our model output in a test set dialog. This conversation has six turns. $U$ represents user utterance, and $S$ is the system response. The user's demand is to book a restaurant first and then a train ticket. In the whole interaction process, the user constantly puts forward the attribute conditions of demand, such as restaurant area and price range. Figure 3 shows that our model outputs the dialog state that needs to be updated in each turn of dialog.

However, in the sixth dialog turn, the model does not pay attention to the user's attitude toward the departure time (i.e., "dontcare"). In other words, it

| Turn | Dialogue | Model outputs $B_t$ | Ground truth $B_t$ |
|---|---|---|---|
| 1 | *U*: I am looking for an expensive restaurant in the north area. | restaurant-area-north<br>restaurant-pricerange-expensive | restaurant-pricerange-expensive<br>restaurant-area-north |
| 2 | *S*: I have moderate and expensive priced restaurants. City stop restaurant serves European and is on the expensive side. Or may I suggest Saigon city, they have Asian?<br>*U*: Do any of them specialize in Chinese? | restaurant-area-north<br>restaurant-pricerange-expensive<br>restaurant-food-Chinese | restaurant-food-Chinese<br>restaurant-pricerange-expensive<br>restaurant-area-north |
| 3 | *S*: Yes, both Hakka and Hotpot serve Chinese. Can I book either for you?<br>*U*: Let's try Hakka. It will be just myself at 16:45 on Tuesday. | restaurant-area-north<br>restaurant-pricerange-expensive<br>restaurant-food-Chinese<br>restaurant-book day-Tuesday<br>restaurant-book people-1<br>restaurant-book time-16:45<br>restaurant-name-Hakka | restaurant-book day-Tuesday<br>restaurant-book people-1<br>restaurant-book time-16:45<br>restaurant-food-Chinese<br>restaurant-pricerange-expensive<br>restaurant-name-Hakka<br>restaurant-area-north |
| ⋮ | ⋮ | ⋮ | ⋮ |
| 6 | *S*: There are 4 trains that will arrive before 12:45, do you have a preferred departure time?<br>*U*: No, any will be fine. I only need a ticket for myself. | restaurant-area-north<br>restaurant-pricerange-expensive<br>restaurant-food-chinese<br>restaurant-book day-Tuesday<br>restaurant-book people-1<br>restaurant-book time-16:45<br>restaurant-name-Hakka<br>train-departure-bishops stortford<br>train-destination-Cambridge<br>train-arriveby-12:45<br>train-day-Tuesday<br>train-book people-1 | restaurant-book day-Tuesday<br>restaurant-book people-1<br>restaurant-book time-16:45<br>restaurant-food-Chinese<br>restaurant-pricerange-expensive<br>restaurant-name-Hakka<br>restaurant-area-north<br>train-book people-1<br>train-leaveat-dontcare<br>train-destination-Cambridge<br>train-day-Tuesday<br>train-arriveby-12:45<br>train-departure-bishops stortford |

**Fig. 3 Example of our model output on a dialog (dialog ID: MUL1598.json) in the test set of MultiWOZ2.1. To save space, we have omitted some parts of the dialog. The green font indicates the state that needs to be tracked. The blue font represents the state that is correctly predicted in each round. The red font indicates the false predictions or unpredicted items.**

does not associate the semantics of the "departure time" and "train-leaveat" slots. The model does not learn such semantic association information. Hence, our future work should not only consider the association information between slots, but also learn from the global semantics.

### 4.6 Ablation analysis

Table 5 shows the results of our ablation experiments. When we remove the calculation of slot-related information from the model, the IRI module will become a full connection layer of slot information. It no longer has the ability to selectively learn slot correlation knowledge, but to learn the relevant information from all slots. This condition allows the non-associated slot connections to participate in the calculation. However, this error makes the model performance drop by 0.34% on MultiWOZ2.0 and 0.83% on MultiWOZ2.1.

**Table 5 Joint goal accuracy of the ablation study on MultiWOZ.**

(%)

| Model | MultiWOZ2.0 | MultiWOZ2.1 |
|---|---|---|
| Ours | **52.26** | **53.59** |
| Without graph | 51.92 | 52.76 |

After removing the graph adjacency matrix, our model can be regarded as the automatic learning of relevant information between slots. At this time, it does not require any prior knowledge and is entirely driven by data. However, even so, the accuracy of the model is basically higher than that of the model based on the open-vocabulary method. This finding illustrates the necessity of learning slot-relevant information.

In the training process of the model, we use the ground truth of the previous dialog state as the input to learn the training dataset. In the iterative process, each epoch is tested on the validation set. After the iteration, the model with the best performance will be used as the output model to be tested in the test set and finally evaluate the model performance. To verify the effect of the output of each module on the performance of the model, we use the ground truth of the previous dialog state t_pstate and slot operation classification t_op for testing.

As shown in Fig. 4, all the models are tested on the MultiWOZ2.1 test set. The evaluation metric is the joint goal accuracy. The purple bar chart (right) shows the model with IRI module and, the blue one (left) represents the model with the fully connected IRI module. "model" represents the original test model, "model+t_pstate" means that the model uses the correctly previous dialog state as the input, and "model+t_op" refers to the model directly using the real result in the operation classifier module. Through the histogram, we can find that the correct slot operation classification result has the greatest effect on the performance of the model. This finding also shows that multi-domain DST tasks need to fuse rich information for slots to enhance their ability to track the slot value in the context of the conversation.

## 5 Conclusion

In this paper, we propose a multi-domain DST model that uses a graph to mine the association information between slots. To improve the ability to learn implicit connections in the model, we design a module using a graph-based multi-head attention mechanism that can incorporate the information of associated slots into each slot. The fused slot representations can provide richer semantic features for the decoding stage to generate more accurate and effective slot values. Experimental results show that our approach has achieved good performance on the MultiWOZ2.0 and MultiWOZ2.1 datasets, proving the effectiveness and necessity of the IRI module.

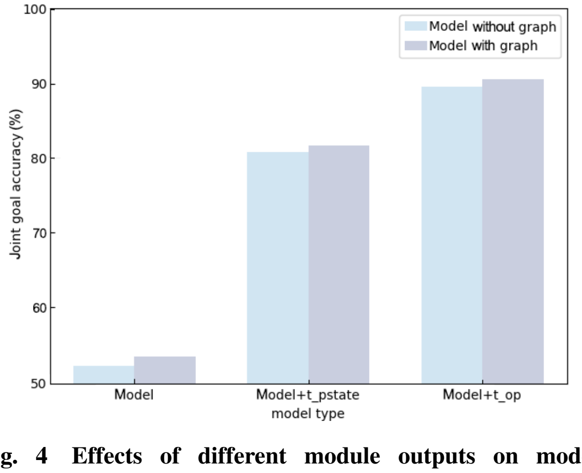

**Fig. 4 Effects of different module outputs on model performance.**

We use the same graph adjacency matrix in all dialogs, but actually, there is no slot connection in some conversations. Therefore, in future works, we will design a new model to establish unique slot-relevant information for each dialogue. Through the analysis of the previous experimental results, we will continue to study the methods of enhancing the semantic representation of slots and integrating global semantics.

## Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61976247).

## References

[1] F. Li, L. Li, J. Yin, L. Huang, Q. Zhou, N. An, and L. Yu, Machine knowledge and human cognition, *Big Data Mining and Analytics*, vol. 3, no. 4, pp. 292–299, 2020.

[2] P. Budzianowski, T. H. Wen, and B. H. Tseng, MultiWOZ–A large-scale multi-domain Wizard-Of-Oz dataset for task-oriented dialogue modelling, arXiv preprint arXiv: 1810.00278, 2018.

[3] M. Henderson, B. Thomson, and S. Young, Word-based dialog state tracking with recurrent neural networks, in *Proc. 5th Annu. Meeting of the Special Interest Group on Discourse and Dialogue*, Philadelphia, PA, USA, 2014, pp. 292–299.

[4] P. Xu and Q. Hu, An end-to-end approach for handling unknown slot values in dialogue state tracking, in *Proc. 56th Annu. Meeting of the Association for Computational Linguistics*, Melbourne, Australia, 2018, pp. 1448–1457.

[5] V. Zhong, C. Xiong, and R. Socher, Global-locally self-attentive dialogue state tracker, arXiv preprint arXiv: 1805.09655, 2018.

[6] L. Ren, K. Xie, L. Chen, and K. Yu, Towards universal dialogue state tracking, in *Proc. 2018 Conf. Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018, pp. 2780–2786.

[7] R. Goel, S. Paul, T. Chung, J. Lecomte, A. Mandal, and D. Hakkani-Tur, Flexible and scalable state tracking framework for goal-oriented dialogue systems, arXiv preprint arXiv: 1811.12891, 2018.

[8] 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 *Proc. 14th*

*Annu. Meeting of the Special Interest Group on Discourse and Dialogue*, Metz, France, 2013, pp. 423–432.

[9] J. D. Williams, Web-style ranking and SLU combination for dialog state tracking, in *Proc. 15th Annu. Meeting of the Special Interest Group on Discourse and Dialogue*, Philadelphia, PA, USA, 2014, pp. 282–291.

[10] J. Perez and F. Liu, Dialog state tracking, a machine reading approach using memory network, in *Proc. 15th Conf. European Chapter of the Association for Computational Linguistics*, Valencia, Spain, 2017, pp. 305–314.

[11] L. Zilka and F. Jurcicek, Incremental LSTM-based dialog state tracke, in *Proc. 2015 IEEE Workshop on Automatic Speech Recognition and Understanding*, Scottsdale, AR, USA, 2015, pp. 757–762.

[12] T. H. Wen, D. Vandyke, and N. Mrksic, A network-based end-to-end trainable task-oriented dialogue system, in *Proc. 15th Conf. European Chapter of the Association for Computational Linguistics*, Valencia, Spain, 2017, pp. 438–449.

[13] N. Mrkšić, D. Ó. Séaghdha, and T. H. Wen, Neural belief tracker: Data-driven dialogue state tracking, in *Proc. 55th Annu. Meeting of the Association for Computational Linguistics*, Vancouver, Canada, 2017, pp. 1777–1788.

[14] M. Eric, R. Goel, and S. Paul, Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines, arXiv preprint arXiv: 1907.01669, 2019.

[15] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] H. Lee, J. Lee, and T. Y. Kim, SUMBT: Slot-utterance matching for universal and scalable belief tracking, in *Proc. 57th Annu. Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019, pp. 5478–5483.

[17] L. Zhou and K. Small, Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering, arXiv preprint arXiv: 1911.06192, 2019.

[18] 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, *Proc. AAAI Conf. Artificial Intelligence*, vol. 34, no. 5, pp. 7521–7528, 2020.

[19] Y. Shan, Z. Li, J. Zhang, F. Meng, Y. Feng, C. Niu, and J. Zhou, A contextual hierarchical attention network with adaptive objective for dialogue state tracking, arXiv preprint arXiv: 2006.01554, 2020.

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

[21] S. Gao, A. Sethi, S. Agarwal, T. Chung, and D. Hakkani-Tur, Dialog state tracking: A neural reading comprehension approach, in *Proc. 20th Annu. SIGdial Meeting on Discourse and Dialogue*, Stockholm, Sweden, 2019, pp. 264–273.

[22] J. Zhang, K. Hashimoto, C. S. Wu, Y. Wang, S. Y. Philip, R. Socher, and C. Xiong, Find or classify? Dual strategy for slot-value predictions on multi-domain dialog state tracking, in *Proc. 9th Conf. Lexical and Computational Semantics*, Barcelona, Spain, 2019, pp. 154–167.

[23] 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 *Proc. 57th Annu. Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019, pp. 808–819.

[24] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in *Proc. 2014 Conf. Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. 1724–1734.

[25] L. Ren, J. Ni, and J. McAuley, Scalable and accurate dialogue state tracking via hierarchical sequence generation, in *Proc. 2019 Conf. Empirical Methods in Natural Language Processing and 9th Int. Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*), Hong Kong, China, 2019, pp. 1876–1885.

[26] H. Le, R. Socher, and S. C. Hoi, Non-autoregressive dialog state tracking, arXiv preprint arXiv: 2002.08024, 2020.

[27] S. Kim, S. Yang, G. Kim, and S. W. Lee, Efficient dialogue state tracking by selectively overwriting memory, in *Proc. 58th Annu. Meeting of the Association for Computational Linguistics*, doi: 10.18653/v1/2020.acl-main.53.

[28] S. Zhu, J. Li, L. Chen, and K. Yu, Efficient context and schema fusion networks for multi-domain dialogue state tracking, in *Proc. 2020 Conf. Empirical Methods in Natural Language Processing*, doi: 10.18653/v1/2020.findings-emnlp.68.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and I. Polosukhin, Attention is all you need, in *Proc. 31th Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 6000–6010.

[30] C. Banerjee, T. Mukherjee, and E. Pasiliao, Feature representations using the reflected rectified linear unit (RReLU) activation, *Big Data Mining and Analytics*, vol. 3, no. 2, pp. 102–120, 2020.

[31] J. L. Ba, J. R. Kiros, and G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450, 2016.

[32] A. See, P. J. Liu, and C. D. Manning, Get to the point: Summarization with pointer-generator networks, in *Proc. 55th Annu. Meeting of the Association for Computational Linguistics*, Vancouver, Canada, 2017, pp. 1073–1083.

[33] 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.

[34] J. Hu, Y. Yang, C. Chen, L. He, and Z.Yu, SAS: Dialogue state tracking via slot attention and slot information sharing, in *Proc. 58th Annu. Meeting of the Association for Computational Linguistics*, doi: 10.18653/v1/2020.acl-main.567.

**Hui Bai** received the BS degree from Southwest Jiaotong University, Chengdu, China in 2019, where she is currently a master student. Her current research interests include data mining, natural language processing, and machine learning.

**Yan Yang** received the BS and MS degrees from Huazhong University of Science and Technology, Wuhan, China in 1984 and 1987, respectively, and the PhD degree from Southwest Jiaotong University, Chengdu, China in 2007. She is currently a professor at the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, China. Her current research interests include multi-view learning, big data analysis and mining, ensemble learning, semi-supervised learning, and cloud computing.

**Jie Wang** received the MS degree from Southwest Petroleum University, Chengdu, China in 2020. He is currently a PhD candidate at the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, China. His current research interests include natural language processing and cross-modal analysis.