GCDST: A Graph-based and Copy-augmented Multi-domain Dialogue State Tracking

Peng Wu, Bowei Zou, Ridong Jiang, Ai Ti Aw
Aural & Language Intelligence Department,
Institute for Infocomm Research (I2R), A*STAR, Singapore
wudapeng268@gmail.com,
{zou_bowei,rjiang,aaiti}@i2r.a-star.edu.sg

Abstract

As an essential component of task-oriented dialogue systems, Dialogue State Tracking (DST) takes charge of estimating user intentions and requests in dialogue contexts and extracting substantial goals (states) from user utterances to help the downstream modules to determine the next actions of dialogue systems. For practical usages, a major challenge to constructing a robust DST model is to process a conversation with multi-domain states. However, most existing approaches trained DST on a single domain independently, ignoring the information across domains. To tackle the multi-domain DST task, we first construct a dialogue state graph to transfer structured features among related domain-slot pairs across domains. Then, we encode the graph information of dialogue states by graph convolutional networks and utilize a hard copy mechanism to directly copy historical states from the previous conversation. Experimental results show that our model improves the performances of the multi-domain DST baseline (TRADE) with the absolute joint accuracy of 2.0% and 1.0% on the MultiWOZ 2.0 and 2.1 dialogue datasets, respectively.

1 Introduction

A task-oriented dialogue system provides fundamental technologies for continuous interactions with a human to accomplish predefined specific goals, such as taxi reservation or hotel booking. Dialogue State Tracking (DST) is a crucial component in the task-oriented dialogue system. Users' intentions and goals are extracted from the current utterances and the conversation history. Then, the DST model encodes the information as a set of states to help dialogue systems to determine which actions should be taken in next steps (Young and Thomson, 2013).

A dialogue state generally comprises an entity attribute (*slot*) and its corresponding *value* of a specific *domain*. For example, there might be a slot-value pair (*book-day*, *Friday*) in the domain of *restaurant*. In general, the dialogue states in DST are predefined by a single domain ontology. However, as a real conversation is inherently complex and across multiple domains, modeling multi-domain DST is of great practical application value in real-life situations. As shown in Figure 1, the conversation includes three domains (*restaurant*, *hotel*, and *taxi*), in which some dialogue states and their expressions, such as the states connected with lines, are similar. This paper focuses on multi-domain DST.

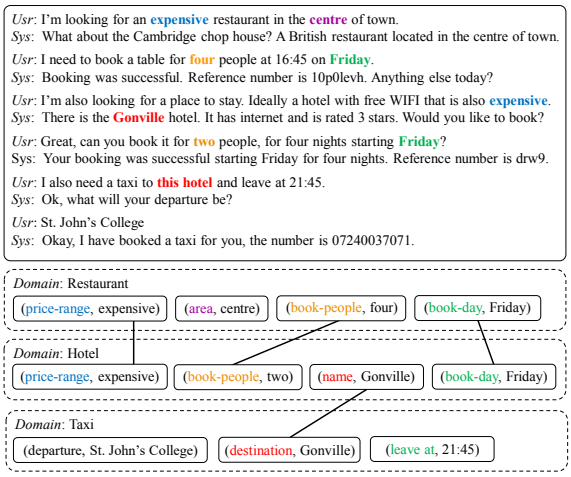

Figure 1: A conversation with dialogue states (solid box) of three domains (dashed boxes) of MultiWOZ 2.0. The colored slots in states are corresponding to their values with the same color in the conversation. Each tuple denotes a slot-value pair, and the lines between them represent that they have the same slot or the same value.

To extract dialogue states from a conversation, there are generally two kinds of approaches. One is utilizing delexicalization to get rephrasings of states by a semantic dictionary (Zilka and Jurci-

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1063–1073

cek, 2015; Rastogi et al., 2017). The other kind of DST models is based on neural networks, which uses word embeddings instead of delexicalization (Mrkšić et al., 2017). However, these approaches lack the capability of sharing and transferring information across domains, which causes low scalability in multi-domain settings.

Recently, Wu et al. (2019) proposed a generative multi-domain DST model (TRADE) based on a copy mechanism, which transfers state representations by sharing the parameters across domains. Beyond that, one challenge is that, is there a more straightforward and explicit approach to encode the states between domains and to further improve the performance of multi-domain DST? Besides, a conversation often piles up long contexts[1]. Previous multi-domain DST systems often behave defectively in predicting dialogue states with such long contexts at the current turn, which shows another challenge of the multi-domain DST task.

To address the above issues, we come up with a more scalable multi-domain DST model. In particular, to better represent the relationships between dialogue states, we first construct a state graph for each conversation. Then, we introduce Graph Convolutional Networks (GCN) (Kipf and Welling, 2017) to better encode the structured information into the representations of history state nodes. For each node, GCN recursively aggregates neighbour information over the dialogue state graph via efficient graph convolution operations, then extracts state-centric representations to benefit the feature transferred across domains. In addition, to avoid too much noise when generating states from long-term contexts, we utilize the previous states from dialogue history and propose a hard copy mechanism for the decoder to pick up the history states directly. To verify the proposed approach, we combine it into an effective multi-domain DST framework (Wu et al., 2019).

The experiments are carried out on the MultiWOZ 2.0 / 2.1 dialogue corpus (Budzianowski et al., 2018; Eric et al., 2019). The results show that the proposed multi-domain DST approach improves 2.0% / 1.0% of joint accuracy over the baseline. We also analyze our model from different perspectives to show the effectiveness of our approach.

The paper proceeds as follows. First, we introduce the state graph-based multi-domain DST model (§2). Next, we describe the experimental results and analyze the effects of different settings and the case study (§3). Finally, we discuss the related work (§4) and conclude the study (§5).

[1] 65% of conversations in MultiWOZ 2.0 are over 5 turns.

2 Method

Figure 2 illustrates the encoder-decoder framework of our **G**raph-based and **C**opy-augmented multi-domain **D**ialogue **S**tate **T**racker (**GCDST**). Different from the previous work (Wu et al., 2019), we introduce state graph representations into both the encoder and the decoder to model the associated information between dialogue states across domains. In addition, we propose a hard copy mechanism in dialogue decoder to get the history states from the last prediction. The framework consists of four main components.

- State graph representation extracts the graph-structured information of dialogue states in a conversation and provides the node representations using graph embeddings.
- Dialogue encoder models history utterances and states of previous turns into a sequence of fixed-length vectors.
- Dialogue decoder with copy mechanism predicts the current slot value by the historical states of the last turn. Such a mechanism helps to decode a sequence of tokens from all possible domain-slot candidates effectively.
- Slot gate, similar to the previous work (Wu et al., 2019), predicts *ptr*, *none*, and *dontcare* to filter some unrelated states.

2.1 State Graph Representation

A practical conversation usually contains dialogue states in more than one domain. Different domains often have lots of same slots that might share the same values or have similar expressions and linguistic features. As shown in Figure 1, when a user books a restaurant, a hotel, and a taxi simultaneously in the conversation, the slot *price-range* exists in the *restaurant* domain and the *hotel* domain respectively. Moreover, for the state expression, the value of *hotel-name* might be as same as that of *taxi-destination*, which means that after booking a hotel, the user will book a taxi to the hotel. Thus, representing and transferring features between the same slots across domains or different domain-slot pairs that have the same values are imperative.

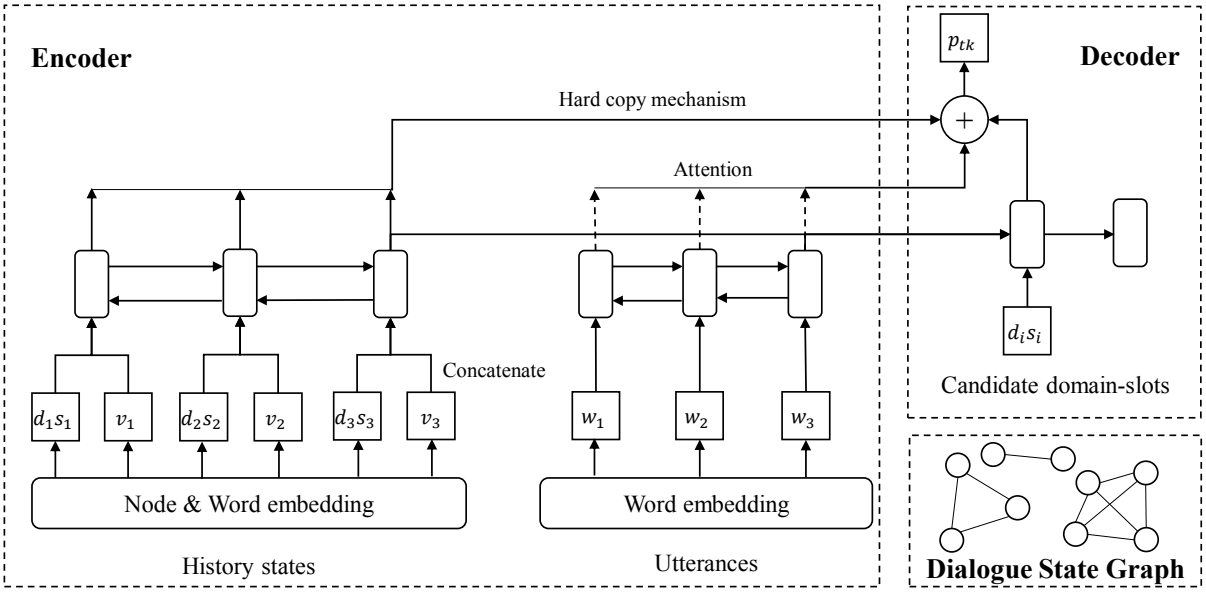

Figure 2: Framework of graph-based and copy-augmented multi-domain dialogue state tracker.

Graph Construction The prior work mainly tracks slot information across domains by sharing parameters (Zhong et al., 2018; Wu et al., 2019). However, it is difficult to transfer the information between slots explicitly and directly. Therefore, we come up with a graph structure to represent the relationship between dialogue states in a conversation. Based on the graph, features learned from the states in one domain are able to directly transfer to other domains.

Formally, a dialogue state graph is denoted as $G = \{N, E\}$, where $N = \{(d, s)\}$ stands for domain-slot tuple nodes and E represents undirected edges between nodes. Considering two nodes $N_i = (d_i, s_i)$ and $N_j = (d_j, s_j)$, we explore four ways of constructing of the edge adjacency matrix $\boldsymbol{A}$:

- Domain-connection: if $d_i = d_j$, $A_{ij} = 1$;
- Slot-connection: if $s_i = s_j$, $A_{ij} = 1$;
- Value-connection[2]: $\exists v_i, v_j$: if $v_i = v_j$, $A_{ij} = 1$ where v_i is one value of N_i;
- Slot/value-connection: union of slot and value connection.

Graph Encoding To propagate information among dialogue state nodes over the graph, we introduce the Graph Convolutional Networks (GCN) (Kipf and Welling, 2017) to update structure-aware node representations by pooling features of their adjacent nodes. In general, the input of GCN includes 1) the node embedding matrix $\boldsymbol{H} \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the number of nodes and d denotes the dimension of node embedding, and 2) the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{|V| \times |V|}$, where $A_{ij} = 1$ if there is an edge between the node N_i and the node N_j, which represents the dialogue state graph structure. In the dialogue state graph, the information propagation among nodes takes up at most two hops away. Thus, we consider a two-layers GCN, in which every layer can be written as a non-linear function and a symmetric adjacency matrix:

$$\begin{aligned} \boldsymbol{H}^0 &= \boldsymbol{I}, \\ \boldsymbol{H}^{l+1} &= \sigma(\hat{\boldsymbol{A}}\boldsymbol{H}^l\boldsymbol{W}^l + \boldsymbol{b}^l), \end{aligned} \tag{1}$$

where $\boldsymbol{H}^l$ is the input node embedding matrix, $\boldsymbol{H}^{l+1}$ is the output node embedding matrix, and $\boldsymbol{W}^l$ and $\boldsymbol{b}^l$ are a parameter matrix and a bias vector for the l-th GCN layer, respectively. $\sigma(\cdot)$ is a non-linear activation function (we use the $ReLU(\cdot)$ in this paper). Finally, we can obtain a $|V| \times d$ node-level feature matrix $\boldsymbol{E}_{node} = \boldsymbol{H}^{l+1}$.

In addition, the adjacency matrix $\boldsymbol{A}$ often adds self-loops to each node in the graph.

$$\hat{\boldsymbol{A}} = \boldsymbol{A} + \lambda \boldsymbol{I}, \tag{2}$$

where $\boldsymbol{I}$ is a $|V| \times |V|$ identity matrix. As suggested in Kipf and Welling (2017), we introduce the trade-off parameter λ, as the importance of self

[2]If the types of slots are times or numbers, e.g. *taxi-leaveat* or *hotel-book people*, there is no connection between these nodes, as the semantic correlation among them is uncertain.

and neighboring node connections might be not equal. Through the self-loop, the representation of each node can be affected by itself.

2.2 Dialogue Encoder

Previous works (Zhong et al., 2018; Wu et al., 2019) only exploited utterances to encode the dialogue history. However, the foregoing dialogue states are informative and related to the current state. For instance, when a user inquiries the area or the number of people for a hotel, she is quite likely to have similar inquiries for other domains such as the restaurant in the following conversation. Thus we propose an utterance encoder and a state encoder to encode history utterances and states respectively, by utilizing bi-directional gated recurrent units (GRU) (Chung et al., 2014).

Specifically, the input of utterance encoder is the word sequence of history utterance $\{w_1, w_2, ..., w_U\}$, where w_i is the ith token of the sequence of the user utterances and the system responses of previous turns. On the other hand, the input of state encoder is denoted as $\{(d_1 s_1, v_1), ..., (d_M s_M, v_M)\}$, where M is the max number of history state, and $d_j s_j$ is the jth domain-slot pair. For each domain slot pair, we use graph embedding to encode it. In a few cases, the value of a domain-slot pair is a phrase that contains more than one word[3]. For simplicity, we encode the value v_j only according to its first word by a shared word embedding of the utterance encoder[4]. Finally, we concatenate $d_j s_j$ and v_j as the input representation and feed it into the state encoder.

During testing, we only use predicted state as input of state encoder, although there might be some errors in the predicted states. In order to simulate this situation, we randomly replace, add, and delete some history states in the training step. Specifically, for replacing or adding operation, only the states that have the same domain, slot, or value are selected as the candidates.

2.3 Dialogue decoder with copy mechanism

To predict the current state of a conversation, both the historical utterances and states can be taken into account. Previous work (Wu et al., 2019) applies a copy mechanism to copy the words from historical utterances, but as the dialogue goes on, the context will become longer. In this case, RNN might lose much information of the states extracted from the first few turns. To address the issue, we first propose a hard copy mechanism to copy the value from the previous state directly, because the history state as a summary of context is important for the current prediction. Then we use a soft-gate to combine the probability based on vocabulary, utterances, and states.

In particular, we use a GRU to decode the value of each domain-slot pair and apply the node embedding $E_{node}(d_k s_k)$ to represent each dialogue state candidate. When decoding the t-th word of $d_k s_k$, the GRU takes a word embedding from the previous step $w_{t-1,k}$ as input. The hidden state of GRU is denoted as $h_{t,k}$. For the first word we use $h_{0,k} = h_u^{enc} + h_s^{enc}$ and $w_{0,k} = E_{node}(d_k s_k)$ to initialize its previous hidden state and word embedding, where h_u^{enc} and h_s^{enc} are the last hidden states of the utterance encoder and the state encoder, respectively. The distributions over vocabulary and historical utterance are calculated by

$$\begin{aligned} p_{t,k}^{vocab} &= \text{Softmax}(W_1 \cdot (h_{t,k})^T) \\ p_{t,k}^{utter} &= \text{Softmax}(H^{utter} \cdot (h_{t,k})^T) \end{aligned} \tag{3}$$

where W_1 is a mapping matrix from hidden state size to vocabulary size and H^{utter} is the history state from the dialogue utterance encoder.

As there might be many unchanged states in each dialogue turn, we try to refer to the history states predicted previously. Thus, we explore two kinds of methods, a hard copy mechanism (Eq.(4)) and an attention-based method (Eq.(5)), to get the distribution over the dialogue history state. While the hard copy mechanism will generate a one-hot vector, the output of attention-based method is a distribution over the vocabulary, as below

$$p_{t,k}^{state} = \text{One-hot}(state_{t,k}), \tag{4}$$

$$p_{t,k}^{state} = \text{Softmax}(W_2 \cdot [h_u^{enc}; h_{t,k}^{dec}; h_s^{enc}]), \tag{5}$$

where $state_{t,k}$ is the t-th word of the domain-slot pair $d_k s_k$ at the last turn. If $state_{t,k}$ is not exist, we fill it by padding. W_2 is a mapping matrix for training.

The final output distribution is a weighted sum of the mentioned three distributions.

$$\begin{aligned} p_{t,k} = (1-\gamma) \times [\beta \times p_{t,k}^{utter} \\ + (1-\beta) \times p_{t,k}^{vocab}] + \gamma \times p_{t,k}^{state} \end{aligned} \tag{6}$$

[3] According to the statistics on the training set of MultiWOZ 2.1, there are 85% of domain-slot pairs containing only one word in their corresponding value.

[4] We also try the ways of averaging the representations by word or using RNNs to encode the words, which get similar or worse results.

Metric	Train	Dev	Test
# Dialogues	8,420	1,000	999
Avg. turns per dialogue	6.73	7.37	7.37
# States (domain-slot pairs)	30	30	30

Table 1: Statistics on MultiWOZ 2.0 and 2.1. Note that one turn consists of one user utterance and its corresponding system response, which is different from the previous works (Budzianowski et al., 2018; Wu et al., 2019).

The parameters β and γ are trainable gates, computed by

$$\begin{aligned} \beta &= \text{Sigmoid}(W_3 \cdot [h_{t,k}; w_{t,k}; c^{uttr}_{t,k}; c^{state}_{t,k}]), \\ \gamma &= \text{Sigmoid}(W_4 \cdot [h_{t,k}; w_{t,k}; c^{uttr}_{t,k}; c^{state}_{t,k}]), \\ c^{uttr}_{t,k} &= p^{uttr}_{t,k} \cdot H^{uttr}, \\ c^{state}_{t,k} &= \text{Softmax}(H^{state} \cdot h_{t,k}) \cdot H^{state}, \end{aligned} \tag{7}$$

where W_3 and W_4 are trainable matrices, and $c^{uttr}_{t,k}$ and $c^{state}_{t,k}$ are context vectors of utterances and states, respectively. By Eq.(6), we are able to copy the states from $p^{state}_{t,k}$ directly.

2.4 Slot Gate

Similar with (Wu et al., 2019), we use a slot gate to predict the probabilities over ptr, $none$, and $dontcare$. If the context does not mention this slot, our gate predicts $none$. The gate predicts $dontcare$ if user think this slot does not matter. If the gate predicts a slot as ptr, we accept the output of the decoder. With the input of context vectors of the utterance and the state, the slot gate for $d_k s_k$ is denoted as

$$G_k = Softmax(W_5 \cdot [c^{uttr}_{1,k}; c^{state}_{1,k}]) \tag{8}$$

where W_5 is a trainable matrix, $c^{uttr}_{1,k}$ and $c^{state}_{1,k}$ are the context vectors computed by Eq.(7).

2.5 Optimization

During training, we optimize the sum of cross-entropy loss L_v of the decoder and L_g of the slot gate,

$$L = L_g + L_v. \tag{9}$$

3 Experimentation

3.1 Settings

Datasets The Multi-domain Wizard-of-Oz dialogue corpus (MultiWOZ 2.0) (Budzianowski et al., 2018)[5] is a human-human written conversational corpus spanning over seven domains. MultiWOZ 2.1 (Eric et al., 2019) is released after fixing 32% annotated noise. For an easy and fair comparison with previous works, we follow the experimental setup in Wu et al. (2019), which only uses five domains, since the other two domains have very few instances and only exist in the training set. Table 1 summarizes the statistics of MultiWOZ 2.0 and 2.1. It shows that the training set of the MultiWOZ contains 8,420 multi-turn dialogues, with an average of 6.73 turns per dialogue, and 30 states with over 4,500 possible values, which makes it significantly more diverse and complex than other datasets such as DSTC2 (Henderson et al., 2014a) and WOZ (Wen et al., 2017). We choose the best model on the development sets and evaluate the performances on the test sets of both MultiWOZ 2.0 and MultiWOZ 2.1. Figure 3 demonstrates the distributions of numbers of dialogues and turns of the five domains on the training set, the development set, and the test set, respectively. Note that the total amount of dialogues in all five domains is larger than that in Table 1 because a dialogue often spans over multiple domains in practice.

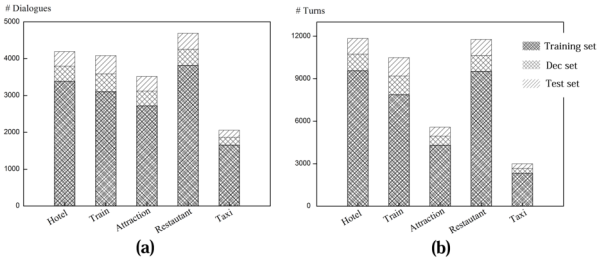

Figure 3: Statistics of the numbers of (a) dialogues and (b) turns of the five used domains on the MultiWOZ 2.0 / 2.1 dialogue corpus.

Metric To evaluate the multi-domain DST models, we employ joint goal accuracy as the evaluation metric. Joint accuracy assesses the predictive capability of the DST model on turn-level. A result is correct only if all of the predicted values exactly match the ground truth in a dialogue turn. This evaluation metric measures the capability of identifying the completed user goals on multiple domains in a turn, which is of paramount importance for multi-domain DST assessment.

Hyper-parameters The word embeddings are initialized with 400-dimensional pre-trained em-

[5] http://dialogue.mi.eng.cam.ac.uk/index.php/corpus/

beddings which concatenated the Glove embeddings (Pennington et al., 2014) and the character n-gram embeddings (Hashimoto et al., 2017). For the two-layer graph convolutional networks, the dimension of the hidden units for the first layer is set to 512, and the dimension of the node embeddings is set to 400. We initialize the input adjacency matrix $\boldsymbol{A}$ by row-normalization. We use the node embeddings to convert dialogue states into 400-dimensional vector representations. During training, we set the dropout with 0.2 ratio. The λ in Eq. (2) is set to 2. The model is trained by using the Adam optimizer (Kingma and Ba, 2015) with a batch size of 32. We apply early stopping based on the joint goal accuracy. In this paper, GCDST refers to the model proposed at Section 2 with slot-connection and hard copy mechanism if not clearly stated.

Baselines We compare with the following models for multi-domain DST.

- MDBT: It leverages semantic interactions between dialogue utterances and ontology terms to learn the shared representations between slots across domains (Ramadan et al., 2018).
- GLAD: By utilizing system actions and user utterances, this model builds global modules to share parameters among slot-value pairs and local modules to learn slot-specific features (Zhong et al., 2018).
- GCE: Based on GLAD, this model replaces the slot-dependent RNN with a global conditioning encoder. It is the state-of-the-art model of single-domain DST (Nouri and Hosseiniasl, 2018).
- SpanPtr: This model uses pointer networks to generate both start and end positions to perform index-based copying (Xu and Hu, 2018).
- TRADE: This model utilizes a copy mechanism that shares parameters across domains, to generate dialogue states from user utterances (Wu et al., 2019).

3.2 Experimental Results

We compare our GCDST with the previous work in Table 2. The results show that GCDST achieves the best performances of joint accuracy of 50.68% on MultiWOZ 2.0 and 46.09% on MultiWOZ 2.1, outperforming the baseline (TRADE) with absolute improvements about 2% on MultiWOZ 2.0 and 1% on MultiWOZ 2.1, respectively. Different from existing multi-domain DST models, we do not use complex decoding algorithms (GLAD) and parameter-sharing mechanism (TRADE). We attribute the performance improvements to the straightforward graph structures, by which it could represent and transfer information among dialogue state nodes via GCN effectively. Moreover, the hard copy mechanism copies the values from previous predicted states directly, which maintains the consistency of the predicted states. The results demonstrate the effectiveness of GCDST on capturing information on multiple domain-slot pairs from dialogues and utilizing the states from historical turns.

Model	MultiWOZ 2.0	MultiWOZ 2.1
MDBT	15.57	-
SpanPtr	30.28	-
GLAD	35.57	-
GCE	36.27	-
TRADE	48.62	44.98*
GCDST	**50.68**	**46.09**

Table 2: Comparison of multi-domain DST models on MultiWOZ 2.0 and 2.1. *: We get the result with the open-sourced model provided by Wu et al. (2019) but on our pre-possessed dataset, while the result reported in Eric et al. (2019) paper is 45.6%.

Connection Type	MultiWOZ 2.0	MultiWOZ 2.1
Slot	**50.68**	**46.09**
Value	49.12	46.04
Slot/Value	49.16	45.30
Domain	45.64	44.72

Table 3: Comparison of different edge connections of GCDST with hard copy on MultiWOZ 2.0 and 2.1.

3.3 Analysis and Discussion

Effects of edge connection In Section 2.1, we propose four types of edge connection for state graph construction, including slot-connection, value-connection, slot/value-connection, and domain-connection. As shown in Table 3, GCDST with slot-connection achieves the best performance on both MultiWOZ 2.0 and 2.1. In addition, the other two connection types by value (value-connection and slot/value-connection) achieve comparative performances. We argue that similar contextualized representations exist between dialogue states that have the same slot or value. For instance, in Figure 1, the states *restaurant-*

State encoder	MultiWOZ 2.0	MultiWOZ 2.1
Hard copy mechanism	**50.68**	**46.09**
Attention-based	50.02	45.99
w/o	49.02	45.17

Table 4: Comparison of different state decoders of GCDST with slot-connection.

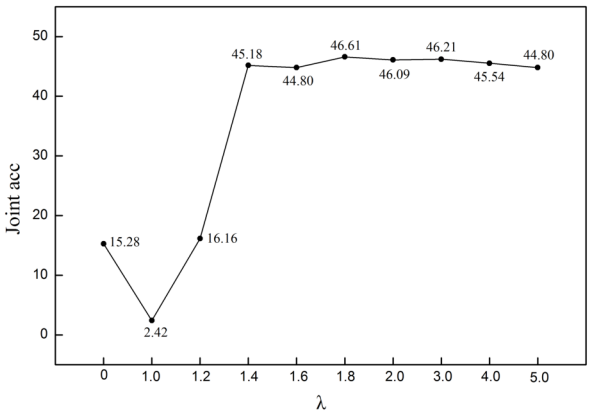

Figure 4: Performances of GCDST with respect to the hyper-parameter λ in Eq.(2) on MultiWOZ 2.1.

book_people and *hotel-book_people* have the same slot *book_people*, so the information between them can be transmitted via the state graph. There are similar effects on the value connection-based graph. However, domain-connection obtains worse performances. It makes sense that different types of states are difficult to share expressions even in the same domain.

Effects of decoder for history states To consult the historical states directly, we introduce two kinds of state encoders, hard copy mechanism and attention-based method, to predict the current states (Section 2.3). Table 4 shows the performances of GCDST with different state encoders. We observe that 1) both of the state encoders improve GCDST, and 2) the GCDST model with hard copy mechanism is slightly better than that with attention-based method. We argue that the hard copy mechanism directly copies the states without considering the hidden states of the utterance encoder and the state encoder, which increases the learning burden of the decoder.

Effects of hyper-parameter λ According to Kipf and Welling (2017), we introduce a trade-off parameter λ into Eq.(2), which balances the impacts between self-loops and neighboring node connections by the adjacent matrix of GCN. To evaluate its effects, we verify GCDST on MultiWOZ 2.1 by varying λ in range [0, 5]. As shown in Figure 4, the joint accuracy suffers from a significant decrease when $\lambda = 1$, which indicates that there is not equal importance between self-connections and edges to neighboring nodes. We also find that the performances become stable when $\lambda \geq 1.4$. Actually, Kipf and Welling (2017) consider that the λ *plays a similar role as the trade-off parameter between supervised and unsupervised loss in the typical semi-supervised setting*. We will try to find the reason for this interesting phenomenon in future work.

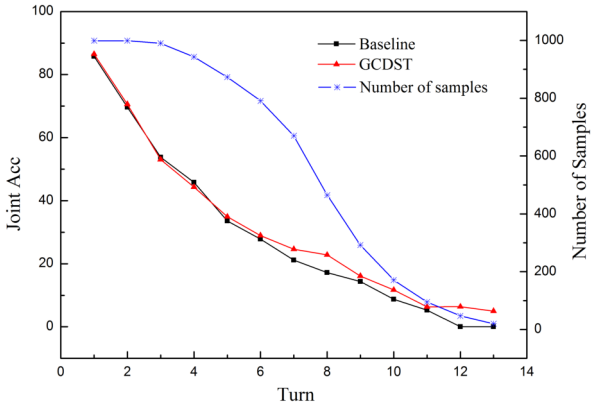

Figure 5: Performances of GCDST and the baseline under different numbers of turns on MultiWOZ 2.1. The samples with more than 13 turns are ignored as there are only 6 of them in total.

Effects of context length Figure 5 illustrates how the performances of DST models change with respect to the context length (turns of dialogue history) on MultiWOZ 2.1. We can see a consistent trend of both the baseline and GCDST: 1) As the conversation progresses through more turns, the performances of both GCDST and the baseline decrease, which suffers from predicting dialogue states for longer context obviously. 2) GCDST and the baseline achieve comparable performance with short dialogue history (turn ≤ 6). As the conversation goes on, GCDST performs better with longer dialogue contexts. We argue that the baseline encodes the previous utterances by only RNN, which might lose some useful information in context. By contrast, GCDST uses an extra encoder to model the previous states and exploits hard copy mechanism to duplicate words from historical state, which can alleviate the forgetting problem to some extent. 3) According to statistics, to the cases that GCDST correctly predicts while the baseline fails to, the average length is 4.48 turns. On the contrary,

ID	Conversation	Model	Turn(s)	Prediction (domain-slot-value)	Result
1	... Sys-2: What are you looking for? Usr-2: Let's start with a moderately priced place to eat. ... Usr-10: I appreciate that .	Baseline	2-7	restaurant-price range-moderate	✓
		Baseline	8-10	restaurant-price range-$\langle N.A. \rangle$	✗
		GCDST	2-10	restaurant-price range-moderate	✓
2	... Usr-4: I am also looking for the hotel a and b guest house. Sys-6: The a&b guesthouse does offer free wifi! Usr-6: Thank you.	Baseline	4-5	hotel-name-a and b guest house	✓
		Baseline	6	hotel-name-$\langle N.A. \rangle$	✗
		GCDST	4-6	hotel-name-a and b guest house	✓
3	Usr-1: Are there any 4 star hotels which are moderately priced? Sys-2: We have 11 guest houses which are moderately priced. Usr-2: That's good. ...	Baseline	2-6	hotel-type-$\langle N.A. \rangle$	✓
		GCDST	2	hotel-type-guest house	✗
		GCDST	3-6	hotel-type-guest house	✗

Table 5: Examples of the predicted dialogue states of GCDST and the baseline. The provenances of the correct predictions are underlined in conversations. $\langle N.A. \rangle$ denotes the model predicts nothing in the current turn.

the average length of the cases only predicted by the baseline is 3.96 turns. It indicates that GCDST is good at dealing with long contexts.

3.4 Case Study

We list three examples of the results on MultiWOZ 2.1, as shown in Table 5. For Case 1, the value *moderate* of the slot *attraction-area* is mentioned at the 2nd turn in the conversation. After the 8th turn, the baseline cannot correctly predict the value for the state due to the long context, while GCDST still predicts it correctly at the 10th turn. It indicates that GCDST can process the longer context, because this model copies values turn by turn by copy mechanisms. For Case 2, the expressions of the slot *hotel-name* are different between the 4th turn (*a and b guest house*) and the 6th turn (*a&b guesthouse*). The baseline can predict correctly in the 4th turn but come to nothing in the 6th turn, which might be due to the misleading by the distinct utterance. In the same case, GCDST gets the correct value by copying it from the previous dialogue state. It indicates that our proposed model can address the issue of expression diversity to some extent. For Case 3, however, the copy mechanisms also might copy an incorrect state from the previous, when the model predicts by mistake.

4 Related Work

Dialogue State Tracking Early research on dialogue state tracking mainly adopted various kinds of natural language understanding modules to extract semantic features from user utterances (Williams and Young, 2007; Thomson and Young, 2010; Henderson et al., 2012; Wang and Lemon, 2013; Williams, 2014). These feature-engineering based approaches heavily rely on hand-crafted complex features which are domain-specific and easily give rise to error propagation. Then, a class of typical methods directly infer dialogue states by semantic dictionaries and delexicalization with the conversation history and the user utterances (Henderson et al., 2014b; Zilka and Jurcicek, 2015; Mrkšić et al., 2015). Although these models possess generalization capability to some extent, it is difficult to obtain a relatively full dictionary. Meanwhile the number of slot value candidates could be large and variable.

With the increasing technological sophistication of neural networks, the mainstream DST approaches turn to neural-based representation learning models, which represent a dialogue state as a distribution over all slot value candidates that are defined in the ontology. Amongst these, neural belief tracker (Mrkšić et al., 2017) is a typical CNN-based DST model which regards DST as a binary classification task to determine whether each slot-value pair in the predefined ontology is represented in the conversation. There are lots of alternative neural-based frameworks presenting to the DST task (Wen et al., 2017; Lei et al., 2018; Ren et al., 2018; Xu and Hu, 2018). However, the aforementioned approaches only focus on the single-domain DST task, which is difficult to extend and scale

from one domain to another.

Multi-domain DST In recent years, more and more researchers are devoted to multi-domain DST. Rastogi et al. (2017) adopted bi-directional GRUs to share parameters across slots and transfer the parameters to a previously unseen domain. Ramadan et al. (2018) estimated the semantic similarities and modeled the interactions between user utterances and the ontology terms to determine which information could be transferred across domains. Zhong et al. (2018) presented global modules to share parameters between slots. Based on their work, Nouri and Hosseiniasl (2018) introduced recurrent networks to further improve performance. Wu et al. (2019) proposed a copy mechanism to generate dialogue states from user utterances and system responses. Such mechanism ensures the knowledge transfer when predicting the unseen (domain,slot,value) triples. Le et al. (2020) introduced a non-autoregressive method into dialogue state tracking to accelerate the state decoding.

Recently, many studies have proposed effective solutions to the multi-domain DST task from various aspects, including the dual strategy model (Zhang et al., 2019), the QA-based model (Zhou and Small, 2019), the memory-based model (Kim et al., 2020), the multi-attention-based model (Budzianowski et al., 2020), the copy strategy model (Heck et al., 2020), and the graph attention neural networks (Chen et al., 2020). Although there is still a performance gap between the proposed model and some of the above models, we argue that the principal motivation of this paper is to verify the effectiveness of graph neural networks and copy mechanisms on multi-domain DST, but not more complicated settings or techniques.

Graph Convolutional Networks for NLP Recently, Graph Convolutional Networks (GCN) (Kipf and Welling, 2017), one typical variant of Graph Neural Networks (GNN) (Cai et al., 2018; Zhou et al., 2018), has been receiving a considerable amount of attention and been widely applied to many NLP tasks such as semantic role labeling (Marcheggiani and Titov, 2017), relation extraction (Zhang et al., 2018; Sun et al., 2019), and question answering (Tu et al., 2019; De Cao et al., 2019). In this paper, we utilize GCN to encode structured information into state node representations.

Copy mechanism It is a useful way to keep the context consistent in sequence-to-sequence frameworks (Zeng et al., 2016; Eric and Manning, 2017; Song et al., 2018). In text summarization, Gu et al. (2016) first introduced copying into a sequence-to-sequence framework to copy a word from the source passage. In machine translation, copy mechanisms often copy rare words (Luong et al., 2015; Gulcehre et al., 2016). Similar to previous studies, we use copy mechanisms to extract the state from the last turn to keep the dialogue state consistent.

5 Conclusion

This paper presents a graph-based and copy-augmented multi-domain DST model (GCDST). In particular, GCDST constructs a graph to transfer knowledge among states with the same slots or values across domains by GCN and encodes the history utterances and states by two independent encoders. Furthermore, we add the hard copy mechanism to directly copy states from the last turn for the decoder. Empirical studies on the MultiWOZ 2.0 / 2.1 dialogue datasets suggest that GCDST outperforms previous systems substantially for the multi-domain DST task. Further analysis demonstrates the positive effects of graph representations for information transferring across domains and advantages of copy mechanisms for state tracking of long-distance dialogue history.

Acknowledgments

We would like to thank the anonymous reviewers who have given valuable and constructive comments as well as insightful suggestions for improving this paper. Bowei Zou is the corresponding author.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *EMNLP*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2020. Ma-dst: Multi-attention based scalable dialog state tracking. In *AAAI*.

Hongyun Cai, Vincent W Zheng, and Kevin Chenchuan Chang. 2018. A comprehensive survey of graph embedding: problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *AAAI*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *NAACL*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv*.

Mihail Eric and Christopher D Manning. 2017. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. In *EACL*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *ACL*.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *EMNLP*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *SIGDIAL*.

Matthew Henderson, Milica Gasic, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young. 2012. Discriminative spoken language understanding using word confusion networks. In *Spoken Language Technology Workshop*.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014a. The second dialog state tracking challenge. In *SIGDIAL*.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. Word-based dialog state tracking with recurrent neural networks. In *SIGDIAL*.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. *arXiv*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Hung Le, Richard Socher, and Steven CH Hoi. 2020. Non-autoregressive dialog state tracking. In *ICLR*.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *ACL*.

Minh-Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *ACL*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *EMNLP*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Multi-domain dialog state tracking using recurrent neural networks. In *ACL*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *ACL*.

Elnaz Nouri and Ehsan Hosseiniasl. 2018. Toward scalable neural dialogue state tracking model. In *NeurIPS*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Osman Ramadan, Paweł Budzianowski, and Milica Gašić. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *ACL*.

Abhinav Rastogi, Dilek Hakkani-Tür, and Larry Heck. 2017. Scalable multi-domain dialogue state tracking. In *ASRU*.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *EMNLP*.

Kaiqiang Song, Lin Zhao, and Fei Liu. 2018. Structure-infused copy mechanisms for abstractive summarization. In *CoLING*.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. In *ACL*.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. *Computer Speech & Language*.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *ACL*.

Zhuoran Wang and Oliver Lemon. 2013. A simple and generic belief tracking mechanism for the dialog state tracking challenge: on the believability of observed information. In *SIGDIAL*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*.

Jason D. Williams. 2014. Web-style ranking and SLU combination for dialog state tracking. In *SIGDIAL*.

Jason D. Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *ACL*.

Puyang Xu and Qi Hu. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *ACL*.

Steve Young and Blaise Thomson. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*.

Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. 2016. Efficient summarization with read-again and copy mechanism. In *ICLR*.

Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. 2019. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv*.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *EMNLP*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *ACL*.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2018. Graph neural networks: A review of methods and applications. *arXiv*.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. In *arXiv*.

Lukas Zilka and Filip Jurcicek. 2015. Incremental lstm-based dialog state tracker. In *ASRU*.