Parallel Interactive Networks for Multi-Domain Dialogue State Generation

Junfan Chen
BDBC and SKLSDE
Beihang University, China
chenjf@act.buaa.edu.cn

Richong Zhang*
BDBC and SKLSDE
Beihang University, China
zhangrc@act.buaa.edu.cn

Yongyi Mao
School of EECS
University of Ottawa, Canada
ymao@uottawa.ca

Jie Xu
School of Computing
University of Leeds, United Kingdom
j.xu@leeds.ac.uk

Abstract

The dependencies between system and user utterances in the same turn and across different turns are not fully considered in existing multi-domain dialogue state tracking (MDST) models. In this study, we argue that the incorporation of these dependencies is crucial for the design of MDST and propose Parallel Interactive Networks (PIN) to model these dependencies. Specifically, we integrate an interactive encoder to jointly model the in-turn dependencies and cross-turn dependencies. The slot-level context is introduced to extract more expressive features for different slots. And a distributed copy mechanism is utilized to selectively copy words from historical system utterances or historical user utterances. Empirical studies demonstrated the superiority of the proposed PIN model.

1 Introduction

Spoken dialogue system (SDS) is an application that can help users complete their goals efficiently. An SDS usually has a logic engine, called dialogue manager, which involves two main sub-tasks for determining how the system will respond to the users: dialogue state tracking and dialogue policy learning. The task we discuss in this paper is dialogue state tracking, which allows the system maintaining an internal representation of the state of the dialogue as the dialogue progress (Young et al., 2010).

Dialogue state tracking involving a single domain has been extensively studied and achieved much progress. As a more challenging task, Multi-domain dialogue state tracking (MDST) has been introduced in (Ramadan et al., 2018) and attracts much attention in the research community. Instead of only predicting the $(slot, value)$ pair, in MDST, a model is expected to predict the $(domain, slot, value)$ triplets for each slot in each domain. This task is a great challenge not only because of the large ontology involving 30 slots and exceeding 4500 values (Wu et al., 2019), but also the mixed-domain nature of the dialogues and some complex cases involving cross-turn inference.

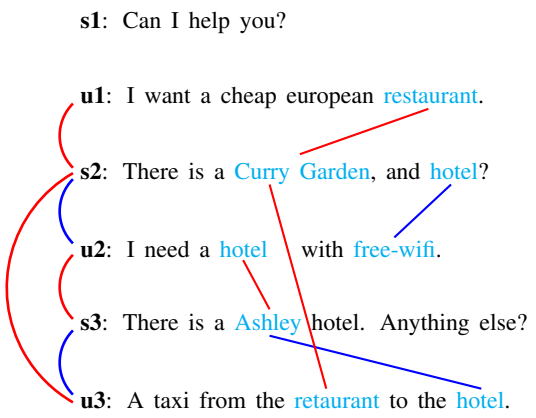

Figure 1: The dependencies between the system utterances and user utterances in a multi-domain dialogue. The red lines imply cross-turn dependencies and the blue lines imply in-turn dependencies.

Several models have been proposed for MDST task and proven to be successful (Mrksic et al., 2015; Ramadan et al., 2018; Goel et al., 2019; Eric et al., 2020; Lee et al., 2019; Wu et al., 2019). Among these models, TRADE (Wu et al., 2019) achieves the state-of-the-art on the MultiWOZ 2.0 dataset (one of the standard MDST datasets) by encoding the entire dialogue history using a bidirectional GRU and incorporating soft-gated copy mechanism to generate the values. Inspired by

*Corresponding author

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1921–1931,
November 16–20, 2020.

TRADE, we purpose to build a more accurate and robust state generator PIN. The motivations of proposing PIN is in two aspects.

One aspect is considering the interactive nature of the dialogues. The interaction of the user and the system is often organized by a question-answering style. It is common in dialogue state tracking that a domain or slot being specified by one of the user or system, then the value being answered by the other. For example, in the dialogue in Figure 1, the user specifies a *Restaurant* domain, and the system answers a restaurant name *Curry Garden*. As is shown in Figure 1, there are two types of dependencies, in-turn dependencies and cross-turn dependencies, both contribute to discovering slot-value pairs. It is worth noting that some hard cases involving inference actually rely on cross-turn dependencies (e.g., the dependency between utterance **s2** and **u3** in Figure 1). Thus a correctly modeling of these dependencies can improve slot-value extraction and cross-turn inference. In this work, we build an Interactive Encoder which completely accords with the dependencies expressed in Figure 1 to jointly model the in-turn dependencies and cross-turn dependencies.

The interactive nature of dialogues also implies that the value for a slot tends to be specified frequently either by a system or by a user. For example, the values for slots involving names, such as *Restaurant-name* and *Hotel-name* are likely to be provided by the system. And the values for the slots like *Hotel-stay* (the days to stay) and *Hotel-people* (the number of people booking for) are usually provided by the user. This observation inspires our designing of the distributed copy mechanism, which allows the state generator choosing to copy words from either the historical system utterances or the historical user utterances.

The other aspect is the slot overlapping problem in MDST. Unlike single-domain DST, slot overlapping is common in MDST, and these overlapping slots share similar values. For example, both the *Restaurant* and *Hotel* domain have a slot *price range* that shares the same values. Under this condition, a generator without considering slot-specific features may mistakenly extract the value of one slot as the value of some other slot. To overcome the slot overlapping problem, we introduce a slot-level context in the state generator.

In summary, we propose a generation-based MDST model which takes into consideration of the interactive nature of dialogues and slot overlapping problem in MDST. The contributions of this work are as follows.

- We propose an interactive encoding method with two parallel double-layer recurrent networks which can jointly model the in-turn dependencies and cross-turn dependencies.
- We introduce the slot-level context into the state generator to accurately generate the values for overlapping slots.
- We present a distributed copy mechanism to selectively copy words from either the historical system utterances or the historical user utterances.

2 Problem Statement

In multi-domain dialogue state tracking, the *state* is usually expressed as a set of $(domain, slot, value)$ triplets. The *domain* refers to the topics of the dialogue, such as the *Restaurant* domain, which indicates that the dialogue involves restaurant booking. The *slot* is an aspect of the user's goals, such as *food*, *area* and *pricerange* in the restaurant-booking dialogues. And the *value* is the user's specific interests, such as *chinese* value for *food* slot that indicates the user is interested in Chinese food. The dialogue state is maintained so as to track the progress of the dialogue. At each turn, the system generates a system utterance in natural language, and the user responds to the system with some sentences, referred to as user utterance. The objective of multi-domain dialogue state tracking is to predict the value of each $(domain, slot)$ pair at each turn given the historical system utterances and user utterances. In this paper, the multi-domain dialogue state tracking is treated as a sequence generation task, where each word of a value is generated from a state generator.

3 Methodology

In this section, we introduce the proposed PIN model. The model consists of four components: Interactive Encoder, Slot-level Context, Value Generator and Slot Gate. We next describe each component in detail.

3.1 Interactive Encoder

Our design of the Interactive Encoder is inspired by the dependencies between the system and user utterances. Specifically, we wish to propose a novel

network structure that completely represents the dependencies expressed in Figure 1. A hierarchical recurrent networks with specific structures have been used to construct the Interactive Encoder, as shown in Figure 2. The Interactive Encoder consists of two parallel hierarchical recurrent networks, one for historical system utterance encoding and another for historical user utterance encoding. The lower layer of the hierarchical recurrent networks allow each word to capture the cross-turn dependencies; and the higher layer of the hierarchical recurrent networks allows each word to capture the in-turn dependencies. In this way, the cross-turn dependencies and in-turn dependencies are jointly modeled.

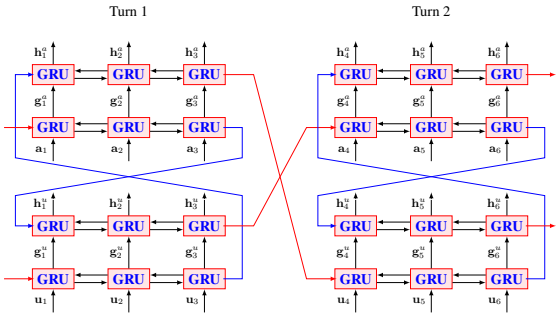

Figure 2: The structure of the Interactive Encoder. Due to space limitation, only two turns are shown. The red arrows emphasize modeling cross-turn dependencies and the blue arrows emphasize modeling in-turn dependencies.

We now present the details of the Interactive Encoder. Let $\mathbf{A}_l = \{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_m\}$ denotes the sequence of word embeddings for the l^{th} system utterance. And $\mathbf{U}_l = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n\}$ denotes the sequence of word embeddings for the the l^{th} user utterance. Here m and n denote the number of words in the l^{th} system utterance and user utterance, respectively.

For later use, we introduce a notation $\mathrm{GRE}(\mathbf{X}, \mathbf{h}; \mathbf{W})$ to indicate the bi-directional GRU encoder (Chung et al., 2014) with inputs $\mathbf{X}$ (sequence of vector representations, such as word embeddings), parameters $\mathbf{W}$ and initialized hidden state $\mathbf{h}$. The Interactive Encoder jointly models the cross-turn dependencies and in-turn dependencies through the following recurrent process.

The Interactive Encoder first let the input word embedding sequences $\mathbf{A}_l$ and $\mathbf{U}_l$ interact with the historical context, allowing the words capturing cross-turn dependencies

$$\begin{aligned} \mathbf{G}_l^a, \mathbf{g}_l^a &= \mathrm{GRE}(\mathbf{A}_l, \mathbf{h}_{l-1}^u; \mathbf{W}_a) \\ \mathbf{G}_t^u, \mathbf{g}_l^u &= \mathrm{GRE}(\mathbf{U}_l, \mathbf{h}_{l-1}^a; \mathbf{W}_u) \end{aligned} \tag{1}$$

where $\mathbf{W}_a$ and $\mathbf{W}_u$ are the parameters of the GRUs, the initialized hidden states $\mathbf{h}_{l-1}^a$ and $\mathbf{h}_{l-1}^u$ are respectively the system context vector and the user context vector generated from the last turn. $\mathbf{G}_l^a$ and $\mathbf{g}_l^a$ denote the entire sequence of output vectors and the last output vector of the GRUs, respectively.

The outputs of the lower-layer GRUs, $\mathbf{G}_l^a$ and $\mathbf{g}_l^a$, are then feed into the higher-layer GRUs to interact with the current context for capturing in-turn dependencies

$$\begin{aligned} \mathbf{H}_l^a, \mathbf{h}_l^a &= \mathrm{GRE}(\mathbf{G}_l^a, \mathbf{g}_l^u; \mathbf{M}_a) \\ \mathbf{H}_l^u, \mathbf{h}_l^u &= \mathrm{GRE}(\mathbf{G}_t^u, \mathbf{g}_l^a; \mathbf{M}_u) \end{aligned} \tag{2}$$

where $\mathbf{M}_a$ and $\mathbf{M}_u$ are the parameters of the higher-layer GRUs, and $\mathbf{h}_l^a$ and $\mathbf{h}_l^u$ are the generated system context vector and user context vector of the current turn. $\mathbf{h}_l^a$ and $\mathbf{h}_l^u$ are then feed into the lower-layer GRUs as the initialized hidden states of the next turn.

With this recurrent architecture, the Interactive Encoder captures the dependencies of the entire dialogue history by rolling from the first turn to the current turn. At the beginning of a dialogue, we set the initialized hidden states as zero vectors, that is $\mathbf{h}_0^a = \mathbf{h}_0^u = \mathbf{0}$.

The outputs from each turn of the dialogue are then concatenated as the system context sequence $\mathbf{H}^a = \{\mathbf{h}_1^a, \mathbf{h}_2^a, \cdots, \mathbf{h}_M^a\}$ and user context sequence $\mathbf{H}^u = \{\mathbf{h}_1^u, \mathbf{h}_2^u, \cdots, \mathbf{h}_N^u\}$. Here M and N denote the total number of words in historical system utterance and historical user utterance, respectively.

3.2 Slot-level Context

The purpose of applying the slot-level context here is to strengthen the context representation with slot specific features and deal with the slot overlapping problem. We employ the attention mechanism to construct the slot-level context. Specifically, for each $(domain, slot)$ pair, we introduce an embedding vector $\mathbf{v}_s$. The slot-level system context $\mathbf{c}_s^a$

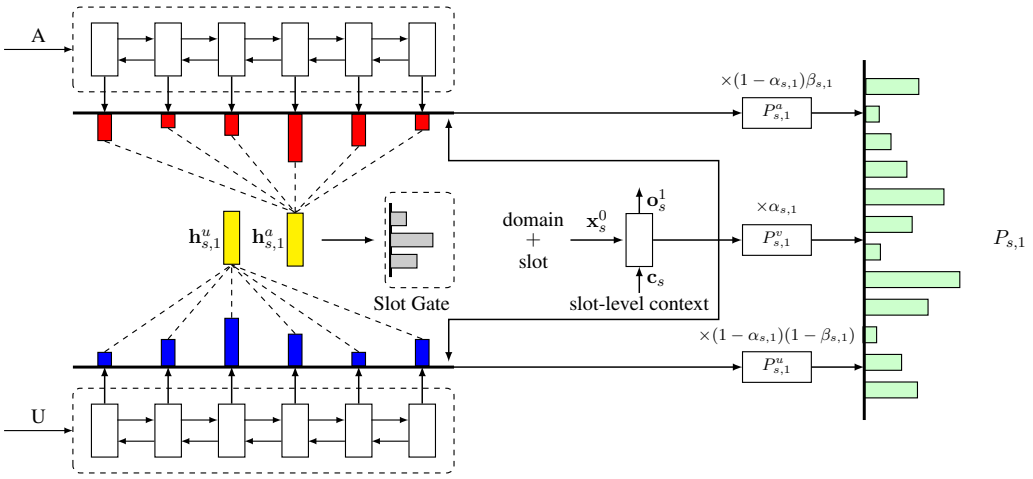

Figure 3: The architecture of the Value Generator and the Slot Gate.

and the slot-level user context $\mathbf{c}^u_s$ are computed by

$$\mathbf{c}^a_s = \sum_{i=1}^{M} \mu_i \mathbf{h}^a_i, \quad \mu_i = \frac{\exp\left(\mathbf{v}_s^T \mathbf{h}^a_i\right)}{\sum_{k=1}^{M} \exp\left(\mathbf{v}_s^T \mathbf{h}^a_k\right)}$$
$$\mathbf{c}^u_s = \sum_{j=1}^{N} \eta_j \mathbf{h}^u_j, \quad \eta_j = \frac{\exp\left(\mathbf{v}_s^T \mathbf{h}^u_j\right)}{\sum_{l=1}^{N} \exp\left(\mathbf{v}_s^T \mathbf{h}^u_l\right)} \tag{3}$$

The slot-level context of the entire dialogue history is then simply the summation of the slot-level system context and the slot-level user context

$$\mathbf{c}_s = \mathbf{c}^a_s + \mathbf{c}^u_s \tag{4}$$

The slot-level context is then feed into the Value Generator as the initialized hidden state for the decoder GRU.

3.3 Value Generator

The Value Generator takes the slot-level context as input and uses a GRU decoder to generate the value sequence for each $(domain, slot)$ pair. Different from the copy mechanism applied in TRADE (Wu et al., 2019) that copying words from the entire dialogue history, in this paper, we propose a distributed copy mechanism that allows the state generator copying words from different sequences. The architecture of the Value Generator is shown in Figure 3. we now describe it in detail.

We use the abbreviation GRD to denote the GRU decoder. At the t^{th} decoding step, the hidden state of the GRU decoder for each $(domain, slot)$ pair s is

$$\mathbf{o}^t_s = \mathrm{GRD}(\mathbf{x}^t_s, \mathbf{o}^{t-1}_s, \mathbf{W}_d) \tag{5}$$

where $\mathbf{x}^t_s$ is the input at the t^{th} step, $\mathbf{o}^t_s$ is the hidden state at the t^{th} step and $\mathbf{W}_d$ is the parameters of the GRU decoder. The hidden state of GRD for each slot is initialized with corresponding slot-level context $\mathbf{c}_s$. The first input $\mathbf{x}^0_s$ is set as the summation of corresponding domain embedding and slot embedding.

We then introduce three distributions on the vocabulary: $P^v_{s,t}$, $P^a_{s,t}$ and $P^u_{s,t}$, for applying distributed copy mechanism. The three distributions represent the probabilities of generating a word from the vocabulary, copying a word from the historical system utterances and copying a word from the historical user utterances, respectively. Let $\mathbf{e}_i$ be the embedding of the i^{th} word in the vocabulary and $|V|$ be the vocabulary size. We use $P_{s,t}[i]$ to denote the i^{th} element in $P_{s,t}$. Then the three distributions are computed by

$$P^v_{s,t}[i] = \frac{\exp\left(\mathbf{e}_i^T \mathbf{o}^t_s\right)}{\sum_{j=1}^{|V|} \exp\left(\mathbf{e}_j^T \mathbf{o}^t_s\right)}$$
$$P^a_{s,t}[i] = \sum_{f(k)=i} \frac{\exp\left((\mathbf{h}^a_k)^T \mathbf{o}^t_s\right)}{\sum_{j=1}^{M} \exp\left((\mathbf{h}^a_j)^T \mathbf{o}^t_s\right)} \tag{6}$$
$$P^u_{s,t}[i] = \sum_{f(k)=i} \frac{\exp\left((\mathbf{h}^u_k)^T \mathbf{o}^t_s\right)}{\sum_{j=1}^{N} \exp\left((\mathbf{h}^u_j)^T \mathbf{o}^t_s\right)}$$

where the function f is used for mapping a distribution on the dialogue-history to corresponding distribution on the vocabulary.

The three distributions, $P^v_{s,t}$, $P^a_{s,t}$ and $P^u_{s,t}$ are then combined by learnable weights. We define $\alpha_{s,t}$ as the weight of generating from the vocabulary and $\beta_{s,t}$ as the weight of choosing to copy a word from the system utterances. For calculat-

ing the weights $\alpha_{s,t}$ and $\beta_{s,t}$, we first generate new feature vectors

$$\mathbf{h}^a_{s,t} = \sum_{i=1}^{M} q^a_{s,t}[i] \cdot \mathbf{h}^a_i, \quad \mathbf{h}^u_{s,t} = \sum_{j=1}^{N} q^u_{s,t}[j] \cdot \mathbf{h}^u_j \quad (7)$$

The weight $\alpha_{s,t}$ and $\beta_{s,t}$ are then computed by

$$\begin{aligned} \alpha_{s,t} &= \sigma(\mathbf{W}_v^T \cdot [\mathbf{x}^t_s, \mathbf{o}^t_s, \mathbf{h}^a_{s,t}, \mathbf{h}^u_{s,t}]) \\ \rho^a_{s,t} &= \mathbf{W}_c^T \cdot [\mathbf{x}^t_s, \mathbf{o}^t_s, \mathbf{h}^a_{s,t}] \\ \rho^u_{s,t} &= \mathbf{W}_c^T \cdot [\mathbf{x}^t_s, \mathbf{o}^t_s, \mathbf{h}^u_{s,t}] \\ \beta_{s,t} &= \frac{\exp(\rho^a_{s,t})}{\exp(\rho^a_{s,t}) + \exp(\rho^u_{s,t})} \end{aligned} \quad (8)$$

where $\mathbf{W}_v$ and $\mathbf{W}_c$ are the parameters of the linear functions, and σ denotes the logistic function.

The final distribution $P_{s,t}$ is then calculated as the weighted sum of distributions $P^v_{s,t}$, $P^a_{s,t}$ and $P^u_{s,t}$ as follows

$$P_{s,t} = \alpha_{s,t} P^v_{s,t} + (1-\alpha_{s,t})(\beta_{s,t} P^a_{s,t} + (1-\beta_{s,t}) P^u_{s,t}) \quad (9)$$

The t^{th} word of the value for $(domain, slot)$ pair s is then generated from distribution $P_{s,t}$. The embedding of the generated word is then used as the next input of the GRU decoder. This generation procedure allows the state generator to generate words from the vocabulary or copy words from either the historical system utterances or the historical user utterances.

3.4 Slot Gate

Following TRADE (Wu et al., 2019), we introduce the slot gate to predict the special values *none* (the value of the slot is not expressed yet) and *dontcare* (the user does not care about the slot) for each $(domain, slot)$ pair. Specifically, the slot gate is a three-class classifier, which aims to identify whether the value *none*, *dontcare* or other value is expressed from the context through a softmax classifier

$$P^c_s = \text{softmax}(\mathbf{W}_s^T \cdot [\mathbf{h}^a_{s,1}, \mathbf{h}^u_{s,1}]) \quad (10)$$

where $\mathbf{W}_s$ is the parameter of the softmax classifier. For a $(domain, slot)$ pair, if the output of the slot gate is *none* or *dontcare*, the generated word sequence from the state generator will be ignored and the corresponding predicted result of the slot gate will be chosen as the value. Otherwise, the generated word sequence from the state generator will be the predicted value for the $(domain, slot)$ pair.

3.5 Loss Function and Optimization

The cross-entropy loss is built for optimizing both the Value Generator and the Slot Gate, simultaneously. Let $\mathcal{S}$ be the total set of $(doamin, slot)$ pairs, and T_s be the number of words in the value for slot $s \in \mathcal{S}$. We define $\mathbf{y}^c_s$ as the ground-truth one-hot label vector of the slot gate and $\mathbf{y}^v_{s,t}$ as the one-hot representation of the t^{th} word in the value of s. The loss function is then defined as

$$\begin{aligned} \mathcal{L} = &\sum_{s \in \mathcal{S}} \sum_{i=1}^{3} -\mathbf{y}^c_s[i] \cdot \log P^c_s[i] \\ &+ \sum_{s \in \mathcal{S}} \sum_{t=1}^{T_s} \sum_{j=1}^{|V|} -\mathbf{y}^v_{s,t}[j] \cdot \log P_{s,t}[j] \end{aligned} \quad (11)$$

The loss function can be optimized by stochastic gradient descent(SGD) method.

4 Experiment

4.1 Datasets

MultiWOZ 2.0. The Multi-Domain Wizard-of-Oz (MultiWOZ 2.0) dataset, collected by (Budzianowski et al., 2018), with conversations spanning over multiple domains and topics, is used to train and evaluate the models. There are total 7 domains with 30 $(domain, slot)$ pairs in the ontology; these $(domain, slot)$ pairs involve $4,510$ values. The dataset contains $10,419$ dialogues with a a total $115,434$ turns; the average turns of dialogue is 13.46. The training, validation and test set contain $8,420$, $1,000$ and $1,000$ dialogues respectively. As is mentioned in (Wu et al., 2019) that *hospital* and *police* domain has very few dialogues and only appear in the training set. We thus follow the dataset setting in (Wu et al., 2019) that only keep five domains (*restaurant*, *hotel*, *attraction*, *taxi*, *train*) in the experiment.

MultiWOZ 2.1. As is pointed out in (Eric et al., 2020), the MultiWOZ 2.0 dataset is faulty in substantial errors in the state annotations and dialogue utterances. In order to clean the dataset, the authors of (Eric et al., 2020) ask crowd-source workers to fix the state annotations and utterances in the original data. As a result, over 32% of state annotations in 40% of the dialogue turns are changed and 146 utterances are fixed. The cleaned dataset is released as the MultiWOZ 2.1 dataset. We also evaluate our models on the MultiWOZ 2.1 dataset.

4.2 Implementation Details

The proposed model is implemented using the Pytorch framework. The code and data are released on the Github page[1]. All the word embeddings are initialized by the concatenation of the pre-trained GloVe embeddings (Pennington et al., 2014) and character n-gram embeddings (Hashimoto et al., 2017). The batch size is set as 32. The dimensions of hidden states in all GRUs are set as 400. The embedding dropout is used in the Interactive Encoder with a dropout rate 0.3. Following (Bowman et al., 2016; Wu et al., 2019), we also adopt the word dropout in the Interactive Encoder to improve the model generalization; and the dropout rate is set as 0.3. At training time, the Value Generator uses Teacher-forcing (Williams and Zipser, 1989) with a probability 0.5. The greedy search (Vinyals and Le, 2015) is used in the decoding process. We use the Adam optimizer (Kingma and Ba, 2015) to optimize the model with an initialized learning rate 0.001.

4.3 Evaluation Metrics

The standard metrics joint goal accuracy and goal accuracy are used to evaluate the multi-domain dialogue state tracking performance. The joint goal accuracy denotes the proportion of dialogue turns where the values of all the $(domain, slot)$ pairs are correctly predicted. While goal accuracy is the proportion of slots whose values are correctly predicted.

4.4 Baseline Models

The recently proposed dialogue state tracking models are used for comparison. The models dealing with dialogue state tracking through building classifiers on predefined ontology include the MDBT (Ramadan et al., 2018), GLAD (Zhong et al., 2018), GCE (Nouri and Hosseini-Asl, 2018), SUMBT (Lee et al., 2019), FJST (Eric et al., 2020), HJST (Eric et al., 2020) and SST (Chen et al., 2020). The models utilizing the copy system include PtrNet (Xu and Hu, 2018). The models incorporating both classifiers and copy system include HyST (Goel et al., 2019), DSTreader (Gao et al., 2019), TRADE (Wu et al., 2019), DST-Picklist (Zhang et al., 2019) and MERET (Huang et al., 2020).

To investigate how much the proposed interactive encoder and distributed copy mechanism contributes to the PIN model, we also report the results of two ablated version of the PIN model: PIN–inter and PIN–dcopy. The PIN–inter model removes the interaction between the two parallel encoders in PIN and allows them to be independent. And the PIN–dcopy model copies words from the entire dialogue history instead of applying the distributed copy.

[1]https://github.com/BDBC-KG-NLP/PIN_EMNLP2020

Table 1: Evaluation on the MultiWOZ 2.0 dataset.

Model	Joint Goal (%)	Goal (%)
MDBT	15.57	89.53
PtrNet	30.28	93.85
GLAD	35.57	95.44
GCE	36.27	**98.42**
HJST	38.40	-
DSTreader	39.41	-
FJST	40.20	-
HyST	42.33	-
HyST(ensemble)	44.22	-
SUMBT	42.40	-
DSTreader+JST	47.33	-
TRADE	48.62	96.92
MERET	50.91	97.07
SST	51.17	-
PIN–inter	51.95	97.24
PIN–dcopy	50.57	97.06
PIN	**52.44**	97.28

4.5 Experimental Results

Evaluation on the MultiWOZ 2.0 dataset. The evaluation results on the MultiWOZ 2.0 dataset are shown in Table 1. We observe that most of the models building classifiers and the models using the copy system to generate the states are inferior to the models utilizing both the classifiers and the copy system. As mentioned in (Eric et al., 2020), the models building upon a copy system have an advantage in extracting values from the dialogue history but struggle to predict values that do not exist in the dialogue history. Thus it is reasonable that models combining copy systems with state classifiers achieve better performance. Compared with the baseline model TRADE and the previous state-of-the-art model SST, PIN achieves significant 3.82% and 1.27% performance gain. This fact demonstrates that the modeling of the interaction dependencies, the slot-level context and the distributed copy mechanism help improve state generation.

Table 2: Evaluation on the MultiWOZ 2.1 dataset.

Model	Joint Goal (%)	Goal (%)
HJST	35.55	-
DST Reader	36.40	-
FJST	38.00	-
HyST	38.10	-
TRADE	45.60	96.55
DST-Picklist	53.30	-
SST	**55.23**	-
PIN–inter	47.36	96.90
PIN–dcopy	47.29	96.91
PIN	48.40	**97.02**

Evaluation on the MultiWOZ 2.1 dataset. The evaluation results on the MultiWOZ 2.1 dataset are shown in Table 2. The consistent performance drop is caused by changing a value to a *dontcare* or *none* label as explained in (Eric et al., 2020). The PIN model outperforms the previous models except for the DST-Picklist and SST model, which indicates the effectiveness of the model design. Although DST-Picklist and SST achieve better performance than PIN, DST-Picklist takes a lot of human efforts in dividing the slots into span-based or picklist-based slots and SST requires extra relation information among the slots. PIN's performance drop in the ablated version (PIN-inter and PIN-dcopy) on both datasets demonstrates the necessity of encoder-interaction and distributed copy.

Table 3: The evaluation results of overlapping slots and non-overlapping slots on the MultiWOZ 2.1 dataset. 1:*Restaurant*, 2:*Hotel*, 3:*Attraction*, 4:*Train*, 5:*Taxi*.

Slot	Domains	TRADE	PIN
area	1,2,3	86.2	**86.4**
book people	1,2,3	92.0	**95.1**
price range	2,3	84.2	**89.7**
book day	2,3	96.4	**96.8**
departure	4,5	89.0	**90.9**
destination	4,5	91.6	**92.4**
leave at	4,5	65.1	**66.7**
arrive by	4,5	82.4	**84.7**
book time	1	**92.7**	91.8
food	1	**92.8**	92.6
parking	2	80.1	**81.2**
book stay	2	**96.4**	96.2
internet	2	78.8	**81.2**

4.6 Evaluation on the Overlapping Slots

In multi-domain dialogue state tracking, domains may have overlapping slots. One of the motivations for building the PIN model is to handle the slot overlapping problem with a slot-level context. Thus we report the goal accuracy on overlapping slots and non-overlapping slots in Table 3 for further analysis on PIN. Table 3 shows that slot overlapping (involve at least two domains) usually appears among similar domains, such as (*Restaurant*, *Hotel*, *Attraction*) and (*Train*, *Taxi*). The PIN model achieves much higher goal accuracy than TRADE on all overlapping slots, compared with non-overlapping slots. This result demonstrates the effectiveness of the slot-level context on extracting distinctive features for each slot so that the values for overlapping slots are correctly predicted.

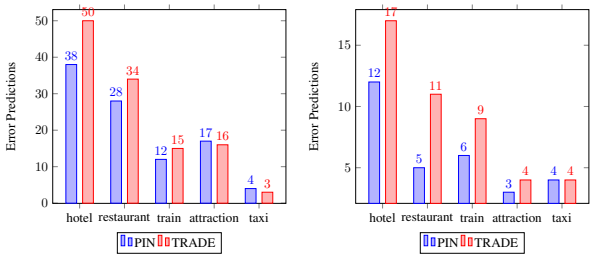

Figure 4: Error analysis on the dialogue turns involving in-turn dependencies (left) and cross-turn dependencies (right). We report the number of error predictions for TRADE and PIN on each domain.

4.7 The Effectiveness of the Interactive Encoder

To study the Interactive Encoder in handling in-turn and cross-turn dependencies in MDST, we make a error analysis on a subset of the test data. We first sample 100 dialogue turns from the MultiWOZ 2.1 test set. Then the wrongly predicted $(domain, slot, value)$ triplets for TRADE and PIN are selected and each of the triplets is marked according to the dependencies (in-turn or cross-turn) involved. The statistics of these error predictions are shown in Figure 4. We observe that whether in the dialogue turns involving in-turn dependencies or cross-turn dependencies, the PIN model creates much fewer prediction errors than the TRADE model, especially on $hotel$ domain and $restaurant$ domain. These results demonstrate the effectiveness of the Interactive Encoder in capturing the in-turn and cross-turn dependencies.

Domain: Restaurant **Slot: food** **Value: european** $\alpha = 0.37$	
System Utterances $\beta = 0.033$	PAD ; ok , i found the cambridge lodge restaurant . would you like ····· i would suggest ····· . would you like me to make a reservation ? ·····
User Utterances $1 - \beta = 0.967$	i am looking for a european restaurant in the west of cambridge; ····· i really need someplace expensive , it is a special occasion for me ·····

Figure 5: An example of dialogues and prediction of PIN. The red color represent the copy probability of the word. And the copy probability of the word *reservation* in system utterances is 0.668, the copy probability of the word *european* in the user utterances is 0.507.

4.8 The Function of the Distributed Copy Mechanism

Unlike the traditional copy mechanism that only copies words from one sequence, the distributed copy mechanism in the PIN model can copy words from two separate sequences considering the interactive nature of dialogues. The example in Figure 5 shows a case that the traditional copy mechanism will make a wrong prediction, but the distributed copy mechanism will correctly predict. The dialogue in Figure 5 is a sample from the Restaurant domain in the test set. In this example, we want to predict the value of the *food* slot. As the wight $\alpha = 0.37$, the generator has a higher probability of copying a word from the dialogue history. In the total dialogue history, if we ignore the wight β, which determines whether to copy from the historical system utterance or the historical user utterance, the generator will copy the wrong word *reservation* from the entire dialogue history because the word *reservation* has higher copy probability 0.668 than 0.507 of the word *european*. This wrong prediction will happen in the traditional copy-based model. But in PIN, the word to be copied also depends on the sequence-selection weight β. With a probability 0.967 to copy the word from the historical user utterance, the correct value *european* will be copied according to Equation 9. This case demonstrates the effectiveness of the distributed copy mechanism.

5 Related Works

The dialogue state tracking (DST) problem has attracted the research community for years. The traditional DST models focus on single domain dialogue state tracking (Thomson and Young, 2010; Wang and Lemon, 2013; Lee and Kim, 2016; Liu and Perez, 2017; Jang et al., 2016; Shi et al., 2016; Vodolán et al., 2017; Yu et al., 2015; Henderson et al., 2014; Zilka and Jurcícek, 2015; Mrksic et al., 2017; Xu and Hu, 2018; Zhong et al., 2018; Ren et al., 2018). Some of these models solve DST problem by incorporating a natural language understanding (NLU) module (Thomson and Young, 2010; Wang and Lemon, 2013) or jointly modeling NLU and DST (Henderson et al., 2014; Zilka and Jurcícek, 2015), which rely on hand-crafted features or delexicalisation features. Other models adopt the representation learning approach and incorporate neural networks to extract features and track the dialogue states (NBT (Mrksic et al., 2017), GLAD (Zhong et al., 2018), StateNet (Ren et al., 2018), PtrNet (Xu and Hu, 2018) and SUMBT (Lee et al., 2019)). Although these models have achieved remarkable success in single-domain DST , they can not be capable enough in multi-domain DST.

Recently, the multi-domain DST attracts more attention than the single-domain DST in the research community. The first work involving state tracking in multiple domains is (Mrksic et al., 2015). This work proposes a pre-training procedure to improve the performance on a new domain. The work of (Rastogi et al., 2017) uses bi-directional GRU to extract features and predict the value by a candidate scoring model. The MDBT (Ramadan et al., 2018) model applies multiple bi-directional-LSTM to jointly track the domain and states. It adopts semantic similarity between the ontology and utterances and allows parameter sharing across domains. The HyST (Goel et al., 2019) model com-

bines a classification-based system and an n-gram copy-based system to deal with multi-domain dialogue state tracking problem. The FJST and HJST model presented in (Eric et al., 2020) employ flatten structured LSTM and hierarchical structured LSTM to encode the dialogue history respectively. The TRADE model (Wu et al., 2019) combines the soft-copy mechanism to generate states and a slot gate to classify special values for each slot. These models motivate our design of the PIN model.

Another idea related to our design of PIN is hierarchical recurrent networks. The hierarchical recurrent networks have been used for dialogue representation in HRED (Serban et al., 2016) and VHRED (Serban et al., 2017). Although our model has a slight flavor of a hierarchical structure (since a sentence-level encoding is sent to another GRU as its initial state), our model is very different from the hierarchical recurrent networks. Specifically, in PIN, the inputs and outputs for each GRU layer are both at the word level; and the GRU layers are parallel, albeit interacting. This is distinct from the hierarchical recurrent networks, where a GRU layer takes word-level inputs, and outputs at the sentence level; then the sentence-level representations are used as the inputs to the next GRU layer.

6 Conclusion

This paper studies the problem of state generation for multi-domain dialogues. Existing generation-based models fail to model the dialogue dependencies and ignore the slot-overlapping problem in MDST. To overcome the limitation of existing models, we present novel Parallel Interactive Networks (PIN) for more accurate and robust dialogue state generation. The design of the PIN model is inspired by the interactive nature of the dialogues and the overlapping slots in the ontology. The Interactive Encoder characterizes the cross-turn dependencies and the in-turn dependencies. The slot-overlapping problem is solved by introducing the slot-level context. Furthermore, a distributed copy mechanism is introduced to perform a selective copy from either the historical system utterances or the historical user utterances. Empirical studies on two benchmark datasets demonstrate the effectiveness of the PIN model.

Acknowledgment

This work is supported partly by the National Natural Science Foundation of China (No. 61772059, 61421003), by the Beijing Advanced Innovation Center for Big Data and Brain Computing (BDBC), by the Fundamental Research Funds for the Central Universities, by the Beijing S&T Committee (No. Z191100008619007) and by the State Key Laboratory of Software Development Environment (No. SKLSDE-2020ZX-14).

References

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5016–5026.

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 *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7521–7528. AAAI Press.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428. European Language Resources Association.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tür. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue, SIGdial 2019, Stockholm, Sweden, September 11-13, 2019*, pages 264–273. Association for Computational Linguistics.

Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 1458–1462. ISCA.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1923–1933.

Matthew Henderson, Blaise Thomson, and Steve J. Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the SIGDIAL 2014 Conference, The 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 18-20 June 2014, Philadelphia, PA, USA*, pages 292–299.

Yi Huang, Junlan Feng, Min Hu, Xiaoting Wu, Xiaoyu Du, and Shuo Ma. 2020. Meta-reinforced multi-domain state generator for dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7109–7118. Association for Computational Linguistics.

Youngsoo Jang, Jiyeon Ham, Byung-Jun Lee, Youngjae Chang, and Kee-Eung Kim. 2016. Neural dialog state tracker for large ontologies by attention mechanism. In *2016 IEEE Spoken Language Technology Workshop, SLT 2016, San Diego, CA, USA, December 13-16, 2016*, pages 531–537.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Byung-Jun Lee and Kee-Eung Kim. 2016. Dialog history construction with long-short term memory for robust generative dialog state tracking. *D&D*, 7(3):47–64.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: slot-utterance matching for universal and scalable belief tracking. In *ACL 2019*, pages 5478–5483.

Fei Liu and Julien Perez. 2017. Dialog state tracking, a machine reading approach using memory network. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 305–314.

Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Pei-hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2015. Multi-domain dialog state tracking using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 794–799.

Nikola Mrksic, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve J. Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1777–1788.

Elnaz Nouri and Ehsan Hosseini-Asl. 2018. Toward scalable neural dialogue state tracking model. *CoRR*, abs/1812.00899.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Osman Ramadan, Pawel Budzianowski, and Milica Gasic. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 432–437.

Abhinav Rastogi, Dilek Hakkani-Tür, and Larry P. Heck. 2017. Scalable multi-domain dialogue state tracking. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*, pages 561–568.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2780–2786.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3295–3301. AAAI Press.

Hongjie Shi, Takashi Ushio, Mitsuru Endo, Katsuyoshi Yamagami, and Noriaki Horii. 2016. Convolutional neural networks for multi-topic dialog state tracking. In *Dialogues with Social Robots - Enablements, Analyses, and Evaluation, Seventh International Workshop on Spoken Dialogue Systems, IWSDS 2016, Saariselkä, Finland, January 13-16, 2016*, pages 451–463.

Blaise Thomson and Steve J. Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech & Language*, 24(4):562–588.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *CoRR*, abs/1506.05869.

Miroslav Vodolán, Rudolf Kadlec, and Jan Kleindienst. 2017. Hybrid dialog state tracker with ASR features. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 205–210.

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 *Proceedings of the SIGDIAL 2013 Conference, The 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 22-24 August 2013, SUPELEC, Metz, France*, pages 423–432.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

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 *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 808–819.

Puyang Xu and Qi Hu. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1448–1457.

Steve J. Young, Milica Gasic, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for pomdp-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

Kai Yu, Kai Sun, Lu Chen, and Su Zhu. 2015. Constrained markov bayesian polynomial for efficient dialogue state tracking. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 23(12):2177–2188.

Jianguo 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. *CoRR*, abs/1910.03544.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive dialogue state tracker. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*.

Lukás Zilka and Filip Jurcícek. 2015. Incremental lstm-based dialog state tracker. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*, pages 757–762.