Multi-Domain Dialogue State Tracking – A Purely Transformer-Based Generative Approach

Yan Zeng
DIRO, Université de Montréal
yan.zeng@umontreal.ca

Jian-Yun Nie
DIRO, Université de Montréal
nie@iro.umontreal.ca

Abstract

We investigate the problem of multi-domain Dialogue State Tracking (DST) with open vocabulary. Existing approaches exploit BERT encoder and copy-based RNN decoder, where the encoder first predicts the state operation, and then the decoder generates new slot values. However, in this stacked encoder-decoder structure, the operation prediction objective only affects the BERT encoder and the value generation objective mainly affects the RNN decoder. In this paper, we propose a purely Transformer-based framework that uses BERT as both encoder and decoder. In so doing, the operation prediction objective and the value generation objective can jointly optimize our model for DST. At the decoding step, we reuse the hidden states of the encoder in the self-attention mechanism of the corresponding decoder layer to construct a flat model structure for effective parameter updating. Experimental results show that our approach substantially outperforms the existing state-of-the-art framework, and it also achieves very competitive performance to the best ontology-based approaches.

1 Introduction

Dialogue state tracking (DST) is a core component in task-oriented dialogue systems. Accurate DST is crucial for appropriate dialogue management, where the user intention is an important factor that determines the next system action. Figure 1 shows an example, where the goal is to extract the output state, i.e. (domain, slot, value) tuples, from the dialogue history and previous dialog state.

Studies on DST have started on ontology-based DST (Henderson et al., 2014; Mrkšić et al., 2017). Recent studies focus on the open-vocabulary setting, where the possible values are not pre-defined and need to be directly extracted (generated) from the input. The current state-of-the-art (Kim et al., 2019; Zhu et al., 2020; Zeng and Nie, 2020) employs BERT (Devlin et al., 2018) as the encoder and stack a RNN-based decoder upon BERT outputs. DST is thus split into two steps with two separate objectives. First, BERT encodes dialogue history and previous dialogue state, and decides whether a (domain, slot) pair needs to update the value. This process employs the State Operation Prediction (SOP) objective. Second, the RNN decoder generates a slot value for a specific (domain, slot) pair. This process uses the Value Generation (VG) objective. By exploiting BERT as the encoder, recent studies have substantially outperformed previous RNN-only framework (RNN encoder and decoder) (Wu et al., 2019). Nevertheless, the framework has some potential problems that need to be solved. First, to improve DST performance, it is better to jointly optimize both SOP and VG processes. However, these two processes work separately and are optimized for two different objectives. Specifically, the SOP objective only affects the BERT encoder, while the VG objective majorly affects the RNN decoder due to the stacked encoder-decoder structure used in the framework, making it less effective in updating the BERT encoder parameters (Liu et al., 2018). Second, in the framework, the encoder is pre-trained while the decoder is not - it is trained...
from scratch. This problem has been observed in some previous studies (Kim et al., 2019). The proposed solution is to employ two different optimizers for the encoder and the decoder in the training process. Although the problem is somewhat less apparent in the experimental results, this solution has indeed further separated the encoder and the decoder, or the SOP and VG processes.

We do not agree that separating the two processes in whatever manner is a good solution to the problem. The radical solution should connect the SOP and VG as much as possible and train them through a joint optimization. Our assumption is that a DST model that shares parameters between the encoder and the decoder could be optimized more effectively.

Inspired by recent progress in dialogue response generation based on pre-trained language models, we propose a purely Transformer-based framework for DST that fully exploits BERT as both the encoder and the decoder. When used as encoder, it processes state operation prediction as in previous works. When using it as decoder, we utilize different input representations to denote the target (decoding) side and left-to-right self-attention mask (i.e. attention is allowed only to previous positions) to avoid information leak. Therefore, the SOP objective and the VG objective affect both the encoder and the decoder, i.e. jointly fine-tuning BERT for DST. Furthermore, instead of a stacked encoder-decoder structure as in previous studies, our purely Transformer-based framework enables a flat structure by re-using the hidden states of the encoder in the self-attention mechanism of the corresponding decoder layer. This makes parameter updating in encoder more effective.

When directly employing the above purely Transformer-based generative framework to DST, we observe however that the model performance drops sharply. The possible reason is that DST is not a genuine generation task, and should not directly inherit an architecture designed for the generation task, which usually needs to cope with the entire input. For example, generating a dialogue response needs to be consistent with the dialogue history, or in the task of generating a translation, translating a sentence usually needs to translate each word on the encoder (input) side. In contrast, in DST, the value to generate only relates to a very small fraction of the model input (within dialogue history and previous dialogue state) that usually consists of one or few tokens. Asking a DST model to take into account the whole dialogue and the previous state may blur the focus.

To solve the problem, we make the following adaptation to DST by borrowing ideas of the existing state-of-the-art framework: For a specific (domain, slot) pair, our decoder only re-uses the hidden states of the most relevant inputs. After an exhaustive search, our experiments show that re-using dialogue of the current turn and the slot state for the specific (domain, slot) pair yields the best performance, which substantially outperforms previous works and only needs a third of the training iterations. Focusing on these relevant elements in the input is the key for the success of DST.

The contributions of this work are as follows 1:

• We propose a purely Transformer-based generative framework for DST. The framework jointly optimizes the state operation prediction and value generation processes. It also has a flat encoder-decoder architecture allowing for more effective parameter updating.

• To accommodate the difference we observed between DST and a genuine generation task, we only re-use the encoder states of dialogue of the current turn and the slot state for a specific (domain, slot) pair in the decoder.

• Our method (Transformer-DST) achieves a new state-of-the-art performance on two public datasets (MultiWOZ 2.0 and MultiWOZ 2.1) on DST, with accuracy of 54.64% and 55.35%. Furthermore, our model can converge to its best setting much faster and in a more stable manner than the existing framework. This shows the efficiency and robustness of the joint optimization of operation prediction and value generation.

2 Related Work

Traditional DST approaches rely on ontology. They assume that not only the domain-slot structure is known, but also the possible values are pre-defined in an ontology. The goal of DST in this context can be simplified into a value classification/ranking task for each slot (Henderson et al., 2014; Mrkšić et al., 2017; Zhong et al., 2018; Ren et al., 2018; Ramadan et al., 2018; Shan et al., 2020). These studies showed the great impact of ontology on

1We will release our codes later.
DST. A recent work (Shan et al., 2020) combining ontology and contextual hierarchical attention has achieved high performance on the public datasets MultiWOZ 2.0 and MultiWOZ 2.1. In real application situations, however, one cannot always assume that ontology is available (Xu and Hu, 2018; Wu et al., 2019). In many cases, slot values are discovered through the conversation rather than predefined (e.g. taxi departure places).

Open-vocabulary DST addresses this problem: it tries to generate or extract a slot value from the dialogue history (Lei et al., 2018; Gao et al., 2019a; Ren et al., 2019). In this work, we focus on open-vocabulary DST. In this setting, DST is usually split into two steps: 1) an encoder reads the model input and decides whether a (domain, slot) pair needs to update value at this turn; 2) a decoder then generates a new value for the (domain, slot) pair. Some existing approaches employ a RNN-only framework with a RNN encoder and a RNN decoder. For example, Wu et al. (2019) encodes the dialogue history using a bi-directional GRU and decodes the value for a (domain, slot) pair using a copy-based GRU decoder. Recent work uses BERT as the encoder to predict state operations and self-attention masks (constant). Furthermore, our framework is a flat encoder-decoder structure by re-using hidden states of encoder in the self-attention mechanism of the corresponding decoder layer. In contrast, previous approaches usually employ a stacked encoder-decoder structure where the decoder utilizes only the outputs of the encoder. This has been shown to be less effective to update encoder parameters especially when using a multi-layer Transformer encoder (Liu et al., 2018). In the next section, we describe our purely transformer-based model for DST.

3 Method

For multi-domain DST, a conversation with \( T \) turns can be represented as \((D_1, S_1), (D_2, S_2), ..., (D_T, S_T)\), where \( D_t \) is the \( t \)-th dialogue turn consisting of a system utterance and a user response, \( S_t \) is the corresponding dialogue state. We define \( S_t \) as a set of \((d_j, s_j, v_j)\) where \( J \) is the total number of (domain, slot) pairs, i.e. \( S_t \) records slot values of all (domain, slot) pairs. If no information is given about \((d_j, s_j)\), \( v_j \) is NULL. In general, we have a limited number of domains and slots in conversations, but a much larger number of slot values. For instance, in MultiWOZ dataset, there are 5 domains, 17 slots, 30 (domain, slot) pairs, and more than 4500 different values.

The goal of DST is to predict \( S_t \) given \tangled{(D_1, S_1), ..., (D_{t-1}, S_{t-1}), (D_t)}\}, i.e. we want to extract the state for the current turn \( t \) of dialogue, given the previous dialogue history and dialogue states. Following Kim et al. (2019) which showed that the most useful information for DST is the recent dialogue turns and the last dialogue state, we only use \( D_{t-1}, D_t, \) and \( S_{t-1} \) to predict \( S_t \). The three elements are concatenated as our model input. Figure 2 gives an overview of our framework.

3.1 State Operation Prediction

**Encoder** The input to the encoder is the concatenation of \( D_{t-1}, D_t, \) and \( S_{t-1} \). Each \((d_j, s_j, v_j)\) tuple in \( S_{t-1} \) is represented by \([\text{SLOT}] \oplus d_j \oplus - \oplus s_j \oplus - \oplus v_j\), where \( \oplus \) denotes token concatenation, and \([\text{SLOT}] \) and \( - \) are separation symbols. Notice that \( s_j \) and \( v_j \) might consist of several tokens. As illustrated in Figure 2, the representations at \([\text{SLOT}] \) position \( \{x^T_{sl_j} | 1 \leq j \leq J\} \) are used for state oper-
CARRYOVER, DELETE, DONTCARE, and UPDATE. Based on the encoder outputs \( \{ x_{ij}^l \mid 1 \leq j \leq J \} \), a MLP layer performs operation classification for each [SLOT]. Specifically, CARRYOVER means to keep the slot value unchanged. DELETE changes the value to NULL, and DONTCARE changes the value to DONTCARE, which means that the slot neither needs to be tracked nor considered important at this turn (Wu et al., 2019). Only if UPDATE is predicted, the decoder generates a new slot value for the (domain, slot) pair.

### 3.2 Slot Value Generation

**Decoder** It applies different type embeddings to represent the input and left-to-right self-attention mask. Furthermore, it re-uses hidden states of encoder in the multi-head self-attention mechanism to construct a flat encoder-decoder structure making parameter updating in the encoder more effective:

\[
\text{C}^l = \text{Concat}(\text{head}_1, \ldots, \text{head}_k) \quad (1)
\]

\[
\text{head}_j = \text{softmax}\left(\frac{Q_j K_j^T}{\sqrt{d_k}} + M^x \right) V_j \quad (2)
\]

where \( Q_j, K_j, V_j \in \mathbb{R}^{n \times d_k} \) are obtained by transforming \( \mathbf{X}^{l-1} \in \mathbb{R}^{|x| \times d_k} \) using \( W_j^Q, W_j^K, W_j^V \in \mathbb{R}^{d_v \times d_k} \) respectively. The self-attention mask matrix \( M^x \in \mathbb{R}^{|x| \times |x|} \) (with \( M_{ij} \in \{0, -\infty\} \)) determines whether a position can attend to other positions. Namely, \( M_{ij}^x = 0 \) allows the \( i \)-th position to attend to \( j \)-th position and \( M_{ij}^x = -\infty \) prevents from it.

In the state operation prediction process, \( M_{ij}^x = 0 \) \( \forall i, j \).

Some hidden states of the encoder will be re-used in the decoder. The outputs of encoder are denoted as \( \mathbf{X}^e = \{ x_{ds}^e, x_1^e, \ldots, x_{ds}^e, \ldots, x_{dj}^e, \ldots \} \), which will be used for operation prediction.

**Objective** Following Gao et al. (2019a) and Kim et al. (2019), we use four discrete state operations:
as the encoding process. In the decoder, the self-attention mask matrix is \( \mathbf{M}_y \in \mathbb{R}^{y \times (|x|+y)} \) and we set \( \mathbf{M}_{ij} = 0 \) if \( j \leq i \). Note that since we directly re-using the hidden states of the encoder, they have already encoded the entire input using bi-directional attention to some extent.

**Objective** The objective of the value generation process is the auto-regressive loss of generated slot values comparing to the ground-truth slot values as in previous work. The final training objective is the sum of the state operation prediction loss and the value generation loss.

4 Experiments

4.1 Datasets

We use two publicly available datasets MultiWOZ 2.0 (Budzianowski et al., 2018) and MultiWOZ 2.1 (Eric et al., 2019) in our experiments. These datasets introduce a new DST task – DST in mixed-domain conversations. For example, a user can start a conversation by asking to book a hotel, then book a taxi, and finally reserve a restaurant. MultiWOZ 2.1 is a corrected version of MultiWOZ 2.0. We use the script provided by Wu et al. (2019) and Kim et al. (2019) to preprocess the datasets, which retain only five domains (restaurant, train, hotel, taxi, and attraction). The final test datasets contain 5 domains, 17 slots, 30 (domain, slot) pairs, and more than 4500 different values. Appendix A gives more statistics of the datasets.

4.2 Implementation Details

Our model is implemented based on the open-source code of SOM-DST (Kim et al., 2019) \(^2\), which using BERT encoder and a copy-based RNN decoder. Our model is initialized with BERT (base, uncased), and it is used for both the encoder and the decoder. We set the learning rate and warmup proportion to 3e-5 and 0.1. We use a batch size of 16. The model is trained on a P100 GPU devices for 15 epochs (a half of the iterations of SOM-DST). In the inference, we use the previously predicted dialogue state as input instead of the ground-truth, and we use greedy decoding to generate the slot value.

4.3 Baselines

We compare the performance of our model, called Transformer-DST, with both ontology-based models and open vocabulary-based models.

**FJST** (Eric et al., 2019) uses a bi-directional LSTM to encode the dialogue history and a feed-forward network to choose the value of each slot.

**HJST** (Eric et al., 2019) encodes the dialogue history using an LSTM like FJST but utilizes a hierarchical network.

**SUMBT** (Lee et al., 2019) uses BERT to initialize the encoder. Then, it scores each candidate slot-value pair using a non-parametric distance measure.

**HyST** (Goel et al., 2019) utilizes a hierarchical RNN encoder and a hybrid approach to incorporate both ontology-based and open vocabulary-based settings.

**DS-DST** (Zhang et al., 2019) uses two BERT-based encoders and designs a hybrid approach for ontology-based DST and open vocabulary DST. It defines picklist-based slots for classification similarly to SUMBT and span-based slots for span extraction as DST Reader.

**DST-Picklist** (Zhang et al., 2019) uses a similar architecture to DS-DST, but it performs only predefined ontology-based DST by considering all slots as picklist-based slots.

**DSTQA** (Zhou and Small, 2019) formulates DST as a question answering problem – it generates a question asking for the value of each (domain, slot) pair. It heavily relies on a predefined ontology.

**SST** (Chen et al., 2020) utilizes a graph attention matching network to fuse utterances and schema graphs, and a recurrent graph attention network to control state updating.

**CHAN-DST** (Shan et al., 2020) employs a contextual hierarchical attention network based on BERT and uses an adaptive objective to alleviate the slot imbalance problem by dynamically adjust the weights of slots during training.

**DST-Reader** (Gao et al., 2019a) formulates the problem of DST as an extractive question answering task – it uses BERT contextualized word embeddings and extracts slot values from the input by predicting spans.

**DST-Span** (Zhang et al., 2019) applies BERT as the encoder and then uses a question-answering method similar to DST-Reader.

**TRADE** (Wu et al., 2019) encodes the dialogue history using a bi-directional GRU and decodes the value for each state using a copy-based GRU decoder.

**NADST** (Le et al., 2020) uses a transformer-based non-autoregressive decoder to generate the

\(^2\)https://github.com/clovaai/som-dst
| Model                      | BERT used | MultiWOZ 2.0 | MultiWOZ 2.1 |
|----------------------------|-----------|--------------|--------------|
| HST (Eric et al., 2019)   |           | 38.40        | 35.55        |
| FIST (Eric et al., 2019)  |           | 40.20        | 38.00        |
| SUMBT (Lee et al., 2019)  | √         | 42.40        | -            |
| HyST (Goel et al., 2019)  |           | 42.33        | 38.10        |
| DS-DST (Zhang et al., 2019)| √        | -            | 51.21        |
| DST-Picklist (Zhang et al., 2019)| √ | -            | 53.30        |
| DSTQA (Zhou and Small, 2019) |          | 51.44        | 51.17        |
| SST (Chen et al., 2020)   |           | 51.17        | 55.23        |
| CHAN-DST (Shan et al., 2020) | √        | 52.68        | 58.55        |
| DST-Reader (Gao et al., 2019b) |         | 59.41        | 56.40        |
| DS-T-Span (Zhang et al., 2019) | √       | -            | 40.39        |
| TRADE (Wu et al., 2019)   |           | 48.60        | 45.60        |
| COMER (Ren et al., 2019)  | √         | 48.79        | -            |
| NADST (Le et al., 2020)   |           | 50.52        | 49.04        |
| SAS (Hu et al., 2020)     |           | 51.03        | -            |
| SOM-DST (Kim et al., 2019) | √       | 51.38        | 52.57        |
| CSFN-DST (Zhu et al., 2020) | √        | 51.57        | 52.88        |
| Graph-DST (Zeng and Nie, 2020) | √    | 52.78        | 53.85        |
| Transformer-DST (ours)    | √         | 54.64        | 55.35        |
| - Full re-use             |           | 25.31        | 27.83        |
| - w/ RNN decoder          |           | 49.59        | 51.44        |

Table 1: Joint goal accuracy (%) on the test set of MultiWOZ. Results for the baselines are taken from their original papers. Full re-use is exploiting all hidden states of encoder. w/ RNN decoder is SOM-DST trained for the same iterations as ours.

| Model      | Attr. | Hotel | Rest. | Taxi | Train |
|------------|-------|-------|-------|------|-------|
| Transformer-DST | 71.11 | 52.01 | 69.54 | 55.92 | 72.40 |
| w/ RNN decoder | 69.21 | 47.60 | 64.89 | 56.39 | 69.14 |
| Graph-DST  | 68.06 | 51.16 | 64.43 | 57.32 | 73.82 |

Table 2: Domain-specific accuracy on MultiWOZ 2.1.

The ground truth values. The accuracy of baseline models is taken from their original papers.

As shown in the table, our Transformer-DST model achieves the highest joint goal accuracy among open-vocabulary DST: 54.64% on MultiWOZ 2.0 and 55.35% on MultiWOZ 2.1 using much few training iterations. Our model even outperforms all ontology-based methods on MultiWOZ 2.0. These latter benefit from the additional prior knowledge, which simplifies DST into a classification/ranking task. On MultiWOZ 2.1, only one ontology-based method – CHAN-DST achieves a higher accuracy than our model.

In Table 2, we show the joint goal accuracy for each domain. The results show that Transformer-DST outperforms previous state-of-the-art framework (w/ RNN decoder) in all domains except in Taxi where a slight decrease is observed. Graph-DST introduces the Dialogue State Graph to encode co-occurrence relations between domain-domain, slot-slot and domain-slot. This method outperforms our approach in Taxi and Train domains. According to the statistics about domains (Appendix A), we can see that Taxi and Train always co-occur with other domains. Therefore, leveraging extra knowledge about co-occurrence relations is particularly helpful for these domains. Our Transformer-DST does not exploit such knowledge. However, this is possible, and we leave it to the future work.

In the following sub-sections, we will examine current state.

SAS (Hu et al., 2020) uses slot attention and slot information sharing to reduce redundant information’s interference and improve long dialogue context tracking.

COMER (Ren et al., 2019) uses BERT-large as the encoder and a hierarchical LSTM decoder.

SOM-DST (Kim et al., 2019) employs BERT as the encoder and a copy-based RNN decoder.

CSFN-DST (Zhu et al., 2020) introduces the Schema Graph considering relations among domains and slots. They also applies BERT encoder and a copy-based RNN decoder.

Graph-DST (Zeng and Nie, 2020) introduces the Dialogue State Graph in which domains, slots and values from the previous dialogue state are connected. They also utilize BERT encoder and a copy-based RNN decoder as in previous work.

4.4 Experimental Results

We report the joint goal accuracy of our model and the baselines on MultiWOZ 2.0 and MultiWOZ 2.1 in Table 1. Joint goal accuracy measures whether all slot values predicted at a turn exactly match the ground truth values. The accuracy of baseline models is taken from their original papers.
several questions: 1) how does joint optimization help the model to converge fast? 2) how is the model efficiency? 3) what is the impact of re-using different parts of the model input?

4.5 Joint Optimization Effectiveness

Figure 3 shows the joint goal accuracy on training set (5k samples) and test set of each epoch. We train Transformer-DST for 15 epochs and w/ RNN decoder (i.e. SOM-DST) for 30 epochs as suggested in the original paper. We can observe that our model performance increases faster than w/ RNN decoder at the beginning, and from 5th to 15th epoch the increase rate of the two frameworks are close. At about 15th epoch, our performance generally stops increasing on both training set and test set.

On the training set, w/ RNN decoder start to out-perform our model (15th epoch) from 22th epoch. On the test set, however, Transformer-DST is consistently better than w/ RNN decoder. This observation suggests that w/ RNN decoder may suffer more from the over-fitting problem than our model.

We also observe that both training curve and test curve of our framework are smoother than w/ RNN decoder, which indicates that our training process is more stable and robust. We can see a larger fluctuation in the curves of w/ RNN decoder.

4.6 Inference Efficiency Analysis

As we have shown that our approach needs much fewer training iterations to achieve state-of-the-art performance, in this subsection we analyze the efficiency at inference/test time. We show in Table 3 the latency of our method and some typical models measured on P100 GPU with a batch size of 1. Since our approach first predicts which (domain, slot) pairs need to be updated, it is about 1 time faster than TRADE (using RNN-only framework) that generates the values of all the (domain, slot) pairs at every turn of dialogue. However, Transformer-DST utilizes a multi-layer Transformer (12 layers) for decoding, which makes it 3 times slower than w/ RNN decoder (i.e. SOM-DST). Overall, when latency is a critical factor in an application, it may be better to use SOM-DST or even NADST (using non-autoregressive decoder). In other cases or having a fast GPU device, the gain in accuracy of Transformer-DST is worth the higher cost in time. More comparison on Inference Time Complexity (ITC) (Ren et al., 2019) of Transformer-DST and baseline models is provided in Appendix B.

| Model                  | Accuracy | Latency |
|------------------------|----------|---------|
| TRADE                  | 45.60    | 450ms   |
| NADST                  | 49.04    | 35ms    |
| w/ RNN decoder         | 51.44    | 50ms    |
| Transformer-DST (Ours) | 55.35    | 210ms   |

Table 3: Average inference time per dialogue turn on MultiWOZ 2.1 test set.

4.7 Re-using Hidden States of Encoder

In our preliminary experiments, we re-use all hidden states of the encoder in the decoder, and the model performance drops sharply comparing to w/ RNN decoder. Since DST is not a genuine generation task such as dialogue response generation that requires to be consistent to the entire dialogue history or machine translation in which every words on the source side need to be translated, we consider re-using only a fraction of the hidden states. In SOM-DST, the RNN decoder only uses $x_{cls}^{L}$ (the final hidden state at the first position) as the summary of the entire model input and $x_{cls}^{L}$ (the final hidden state at the $j$-th [SLOT] position) as the summary of the $j$-th (domain, slot, value). Inspired by this, we conduct an exhaustive search on which hidden states should be re-used. The results are listed in Table 4. We can see that re-using encoding hidden states of the current dialogue turn $D_{i}$ and $j$-th [SLOT] achieves the best performance.
| Transformer-DST       | Joint Accuracy |
|----------------------|----------------|
| Full re-use          | 27.83          |
| $D_{t-1} + D_t + [SLOT]$ | 53.08          |
| $D_t + [SLOT]$       | **55.35**      |
| $D_t + [SLOT] + (d,s,v)$ | 52.67          |
| $D_t + [SLOT] + (d,s)$ | 52.40          |
| $[SLOT] + (d,s)$     | 53.83          |

Table 4: Joint goal accuracy on MultiWOZ 2.1.

5 Conclusion

The existing state-of-the-art approaches to DST in open-vocabulary setting exploited BERT encoder and copy-based RNN decoder. The encoder predicts state operation, and then the decoder generates new slot values. However, the operation prediction objective affects only the BERT encoder and the value generation objective mainly influences the RNN decoder because of the stacked model structure.

In this paper, we proposed a purely Transformer-based framework that uses BERT for both the encoder and decoder. The operation prediction process and the value generation process are jointly optimized. When decoding, we re-use the hidden states of encoder in the self-attention mechanism of the corresponding decoder layer to construct a flat encoder-decoder structure for effective parameter updating. Our experiments on MultiWOZ datasets show that our model substantially outperforms the existing state-of-the-art framework, and it also achieves very competitive performance to the best ontology-based approaches.

Some previous works in DST has successfully exploited using extra knowledge, e.g. a Dialogue State Graph (in Graph-DST). Such a graph could also be incorporated into our framework to further enhance its performance. We leave it to our future work.

References

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadán, and Milica Gašić. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

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, pages 7521–7528.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tür. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoun Chung, and Dilek Hakkani-Tür. 2019a. Dialog state tracking: A neural reading comprehension approach. *arXiv preprint arXiv:1908.01946*.

Xiang Gao, Sungjin Lee, Yizhe Zhang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2019b. Jointly optimizing diversity and relevance in neural response generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1229–1238.

Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. *arXiv preprint arXiv:1907.00883*.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299.

Jiaying Hu, Yan Yang, Chencai Chen, Zhou Yu, et al. 2020. Sas: Dialogue state tracking via slot attention and slot information sharing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6366–6375.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2019. Efficient dialogue state tracking by selectively overwriting memory. *arXiv preprint arXiv:1911.03906*.

Hung Le, Richard Socher, and Steven CH Hoi. 2020. Non-autoregressive dialog state tracking. *arXiv preprint arXiv:2002.08024*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. *arXiv preprint arXiv:1907.07421*.

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 *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447.
Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788.

Osman Ramadan, Paweł Budzianowski, and Milica Gašić. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 432–437.

Liliang Ren, Jianmo Ni, and Julian McAuley. 2019. Scalable and accurate dialogue state tracking via hierarchical sequence generation. *arXiv preprint arXiv:1909.00754*.

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*, pages 2780–2786.

Yong Shan, Zekang Li, Jinchao Zhang, Fandong Meng, Yang Feng, Cheng Niu, and Jie Zhou. 2020. A contextual hierarchical attention network with adaptive objective for dialogue state tracking. *arXiv preprint arXiv:2006.01554*.

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. *arXiv preprint arXiv:1905.08743*.

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 (Volume 1: Long Papers)*, pages 1448–1457.

Yan Zeng and Jian-Yun Nie. 2020. Multi-domain dialogue state tracking based on state graph.

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 dialogue state tracking. *arXiv preprint arXiv:1910.03544*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467.
A Dataset Statistics

MultiWOZ 2.1 is a refined version of MultiWOZ 2.0 in which the annotation errors are corrected. Some statistics of MultiWOZ 2.1 are reported here.

| Domain Transition | First  | Second | Third | Count |
|-------------------|--------|--------|-------|-------|
| restaurant        | train  | -      | 87    |       |
| attraction        | train  | -      | 80    |       |
| hotel             | -      | -      | 71    |       |
| train             | attraction | -    | 71    |       |
| train             | hotel  | -      | 70    |       |
| restaurant        | -      | -      | 64    |       |
| train             | restaurant | -  | 62    |       |
| hotel             | train  | -      | 57    |       |
| taxi              | -      | -      | 51    |       |
| attraction        | restaurant | -  | 38    |       |
| restaurant        | attraction | taxi | 35    |       |
| restaurant        | attraction | -    | 31    |       |
| train             | -      | -      | 31    |       |
| hotel             | attraction | -  | 27    |       |
| restaurant        | hotel  | -      | 27    |       |
| restaurant        | hotel  | taxi  | 26    |       |
| attraction        | hotel  | taxi  | 24    |       |
| attraction        | restaurant | taxi | 23    |       |
| hotel             | restaurant | -  | 22    |       |
| attraction        | hotel  | -      | 20    |       |
| hotel             | attraction | taxi | 16    |       |
| hotel             | restaurant | taxi | 10    |       |

Table 5: Statistics of domain transitions that correspond to more than 10 dialogues in the test set of MultiWOZ 2.1. Train domain always co-occurs with another domain. Taxi always co-occurs with another two domains.
| Domain   | Slots                                                                 | Train | Valid | Test  |
|----------|----------------------------------------------------------------------|-------|-------|-------|
| Attraction | area, name, type                                                   | 8,073 | 1,220 | 1,256 |
| Hotel    | price range, type, parking, book stay, book day, book people, area, stars, internet, name | 14,793 | 1,781 | 1,756 |
| Restaurant | food, price range, area, name, book time, book day, book people   | 15,367 | 1,708 | 1,726 |
| Taxi     | leave at, destination, departure, arrive by                       | 4,618  | 690   | 654   |
| Train    | destination, day, departure, arrive by, book people, leave at      | 12,133 | 1,972 | 1,976 |

Table 6: Data statistics of MultiWOZ 2.1 including domain and slot types and number of turns in train, valid, and test set.
## B Inference Time Complexity (ITC)

| Model               | Inference Time Complexity | Best   | Worst   |
|---------------------|---------------------------|--------|---------|
| SUMBT               | $\Omega(JM)$              | $O(JM)$|         |
| DS-DST              | $\Omega(J)$               | $O(JM)$|         |
| DST-picklist        | $\Omega(JM)$              | $O(JM)$|         |
| DST Reader          | $\Omega(1)$               | $O(J)$  |         |
| TRADE               | $\Omega(J)$               | $O(J)$  |         |
| COMER               | $\Omega(1)$               | $O(J)$  |         |
| NADST               | $\Omega(1)$               | $O(1)$  |         |
| ML-BST              | $\Omega(J)$               | $O(J)$  |         |
| SOM-DST             | $\Omega(1)$               | $O(J)$  |         |
| CSFN-DST            | $\Omega(1)$               | $O(J)$  |         |
| Graph-DST           | $\Omega(1)$               | $O(J)$  |         |
| Transformer-DST (ours) | $\Omega(1)$           | $O(J)$  |         |

Table 7: Inference Time Complexity (ITC) of our method and baseline models. We report the ITC in both the best case and the worst case for more precise comparison. $J$ indicates the number of slots, and $M$ indicates the number of values of a slot.