Schema-Guided Multi-Domain Dialogue State Tracking with Graph Attention Neural Networks

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, Kai Yu*
MoE Key Lab of Artificial Intelligence
SpeechLab, Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
{chenlusz, boerlv, wangchi16, paul2204, tanbowen, kai.yu}@sjtu.edu.cn

Abstract

Dialogue state tracking (DST) aims at estimating the current dialogue state given all the preceding conversation. For multi-domain DST, the data sparsity problem is also a major obstacle due to the increased number of state candidates. Existing approaches generally predict the value for each slot independently and do not consider slot relations, which may aggravate the data sparsity problem. In this paper, we propose a **S**chema-guided multi-domain dialogue **S**tate **T**racker with graph attention networks (**SST**) that predicts dialogue states from dialogue utterances and schema graphs which contain slot relations in edges. We also introduce a graph attention matching network to fuse information from utterances and graphs, and a recurrent graph attention network to control state updating. Experiment results show that our approach obtains new state-of-the-art performance on both MultiWOZ 2.0 and MultiWOZ 2.1 benchmarks.

1 Introduction

Dialogue state tracking (DST) is a key component in task-oriented dialogue systems which cover certain narrow domains, such as *booking restaurant* and *travel planning*. The goal of DST is to extract user goals or intents hidden in human-machine conversation and represent them as a compact dialogue state, i.e. a set of slots and their corresponding values. For example, as illustrated in Fig. 1, (*slot*, *value*) pairs like (*area*, *west*) are extracted from the dialogue. High-qualified DST is essential for dialogue management (Young et al. 2013; Yu et al. 2014), where dialogue state determines the next machine action and system reply.

Recently, motivated by commercial applications like Apple Siri, Microsoft Cortana, Amazon Alexa, or Google Assistant, there is significant interest in adding numerous domains to these dialogue systems. Therefore, multi-domain based DST becomes crucial.

However, most traditional state tracking approaches focus on a single domain. They extract value for each slot predefined in the domain (Williams et al. 2013; Henderson, Thomson, and Williams 2014a; 2014b). These methods can be directly adapted to multi/mixed-domain conversations by replacing slots in a single domain with *domain-slot* pairs (i.e. domain-specific slots) predefined (Ramadan, Budzianowski, and Gasic 2018; Gao et al. 2019; Wu et al. 2019). Despite its simplicity, this approach for multi-domain DST suffers from two major drawbacks: 1) Relations among domain-specific slots are not considered, e.g., *hotel-price_range* and *restaurant-price_range* are actually the same and domain-independent. The slot relations may also consist of whether two slots are from the same domain and whether two slots have the same value type (e.g. location, day, date, time, number, bool etc.). 2) The approach extracts value for each slot independently, which may fail to capture features from slot co-occurrences. For example, hotels with higher *stars* are usually more expensive (*price_range*). These may aggravate the data sparsity problem.

To tackle these challenges, we emphasize that DST models should incorporate slot relations and support information interactions among different slots. To consider relations among domain-specific slots, we introduce schema graphs. In the graph, each node is a slot, and each edge between two slots means that they are from the same domain, or they have the same value type. To encode the graph and make information interactions among different slots, we adopt graph attention networks (GATs).

In this paper, we propose a schema-guided multi-domain dialogue state tracker with GATs. It is elegant to utilize GATs to extract schema information. Our approaches are evaluated on MultiWOZ 2.0 and MultiWOZ 2.1 benchmarks with extensive experiments. Contributions in this work are summarized as:

- We are the first to incorporate slot relations and model slot interactions in multi-domain DST to the best of our knowledge. This is also the first time that graph neural networks are exploited in DST.
- To fully encode the schema graph and dialogue context (user and system utterances), graph attention matching networks (GAMTs) are introduced in this paper, which include internal and external attention mechanisms.
- To exploit previous states in conversations, a novel recurrent GAT (RGAT) is proposed with gated recurrent units

*Kai Yu is the corresponding author.

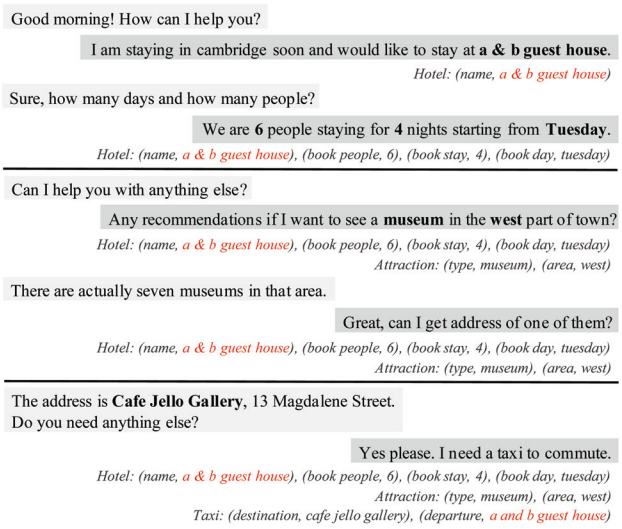


Figure 1: An example of multi-domain dialogues. Utterances at the left and the right sides are from system and user, respectively. The belief state of each domain is represented as a set of (slot, value) pairs. A dialogue state tracker should be able to track all the slot values mentioned previously or changed during the conversation.

(GRUs), which learn to keep or forget history.

- Experimental results show that our approach achieves new state-of-the-art performance (joint goal accuracy) on both MultiWOZ 2.0 (+2.57%) and MultiWOZ 2.1 (+9.63%) benchmarks. Our ablation studies also confirm that the schema graph is important.

2 Background

In this section, we introduce graph attention networks (GATs), which are the basis of our proposed DST models in the next section.

GAT is a special type of graph neural networks (GNNs). GNN is a deep neural network associated with a graph (Scarselli et al. 2009). We first give some notations before describing the details of GNN and GAT. We denote the graph as $G = (V, E)$, where V and E are the set of nodes x_i and the set of edges e_{ij} respectively. $\mathcal{N}(x_i)$ denotes the neighbors of node x_i. $\mathcal{N}_+(x_i)$ is the set including x_i and all neighbors of x_i, i.e. $\mathcal{N}_+(x_i) = \mathcal{N}(x_i) \cup \{x_i\}$.

For each node x_i in the graph, there is an input feature $\mathbf{x}_i$. GNN takes $\mathbf{x}_i$ as the initial embedding $\mathbf{h}_i^0$ for node x_i, then updates its embedding from one step (or layer) to the next with following operations.

Sending Messages At l-th step, each node x_i will send a message $\mathbf{m}_i^l$ to all nodes $x_j \in \mathcal{N}_+(x_i)$:

$$\mathbf{m}_i^l = f_{msg}^l(\mathbf{h}_i^{l-1}), \tag{1}$$

where $f_{msg}^l(\cdot)$ is a message function for each node at l-th step. For simplicity, in GNN a linear transformation $f_{msg}^l(\cdot)$ is often used: $f_{msg}^l(\mathbf{h}_i^{l-1}) = \mathbf{W}_m^l \mathbf{h}_i^{l-1}$, where $\mathbf{W}_m^l$ is a weight matrix for optimization.

Aggregating Messages After sending messages, each node x_i will aggregate messages from its neighbors and itself,

$$\mathbf{e}_i^l = f_{agg}^l(\{\mathbf{m}_j^l | x_j \in \mathcal{N}_+(x_i)\}), \tag{2}$$

where the function $f_{agg}^l(\cdot)$ is the aggregation function for every node at l-th step. The biggest difference between GAT and traditional GNNs is that they have different aggregation functions. In traditional GNNs, all received messages are treated equally, i.e.

$$\mathbf{e}_i^l = f_{agg}^l(\{\mathbf{m}_j^l | x_j \in \mathcal{N}_+(x_i)\}) = \frac{1}{N_i} \sum_{v_j \in \mathcal{N}_+(x_i)} \mathbf{m}_j^l, \tag{3}$$

where N_i is the number of nodes in $\mathcal{N}_+(x_i)$. However, in practice, some messages are more important than others. Similar to *self-attention* model for machine translation (Vaswani et al. 2017), different weights a_{ij}^l are specified to different messages $\mathbf{m}_j^l$ in GAT. Here a_{ij}^l is the normalized similarity of the embedding between the two nodes x_i and x_j in an unified space, i.e.

$$a_{ij}^l = \frac{e^{f_{sim}^l(\mathbf{h}_i^{l-1}, \mathbf{h}_j^{l-1})}}{\sum_{x_k \in \mathcal{N}_+(x_i)} e^{f_{sim}^l(\mathbf{h}_i^{l-1}, \mathbf{h}_k^{l-1})}}, \tag{4}$$

where $f_{sim}^l(\cdot)$ is the similarity function,

$$f_{sim}^l(\mathbf{h}_i^{l-1}, \mathbf{h}_j^{l-1}) = (\mathbf{W}_{a1}^l \mathbf{h}_i^{l-1})^\mathsf{T} (\mathbf{W}_{a2}^l \mathbf{h}_j^{l-1}), \tag{5}$$

where $\mathbf{W}_{a1}^l$ and $\mathbf{W}_{a2}^l$ are learnable weights for projections. Once obtained, these normalized attention coefficients a_{ij}^l are used to compute a linear combination of messages,

$$\mathbf{e}_i^l = f_{agg}^l(\{\mathbf{m}_j^l | x_j \in \mathcal{N}_+(x_i)\}) = \sum_{x_j \in \mathcal{N}_+(x_i)} a_{ij}^l \mathbf{m}_j^l. \tag{6}$$

Note that, in the above equation, a_{ij}^l is a scalar, which means that all dimensions in $\mathbf{m}_j^l$ are treated equally. This may limit the capacity to model complex dependencies. Multi-dimensional (multi-dim) attention is a natural extension of vanilla attention. It has been shown that multi-dim attention can handle context variation and polysemy problems in many NLP tasks (Shen et al. 2018). In this paper, we utilize it as the message aggregating function $f_{agg}^l(\cdot)$. Instead of computing a single scalar score for each message $\mathbf{m}_j^l$ as shown in Eq.(4), multi-dim attention computes a feature-wise score vector $\hat{\mathbf{a}}_j^l$ for each $\mathbf{m}_j^l$,

$$\hat{\mathbf{a}}_{ij}^l = f_{sim}^l(\mathbf{h}_i^{l-1}, \mathbf{h}_j^{l-1}) + f_{md}^l(\mathbf{h}_j^{l-1}), \tag{7}$$

where $f_{sim}^l(\cdot)$ is a scalar and $f_{md}^l(\cdot)$ is a vector. The addition in the equation means the scalar will be added to every element of the vector. $f_{sim}^l(\cdot)$ is used to model the pairwise dependency as shown in Eq.(5), and $f_{md}^l(\cdot)$ is used to estimate the contribution of each feature dimension of $\mathbf{m}_j^l$. It can be a MLP with one hidden layer,

$$f_{md}^l(\mathbf{h}_j^{l-1}) = \mathbf{W}_{md2}^l \sigma(\mathbf{W}_{md1}^l \mathbf{h}_j^{l-1} + \mathbf{b}_{md1}^l) + \mathbf{b}_{md2}^l, \tag{8}$$

where $\mathbf{W}_{md1}^l$, $\mathbf{W}_{md2}^l$, $\mathbf{b}_{md1}^l$ and $\mathbf{b}_{md2}^l$ are learnable parameters, and $\sigma(\cdot)$ is an activation function, e.g. ReLU.

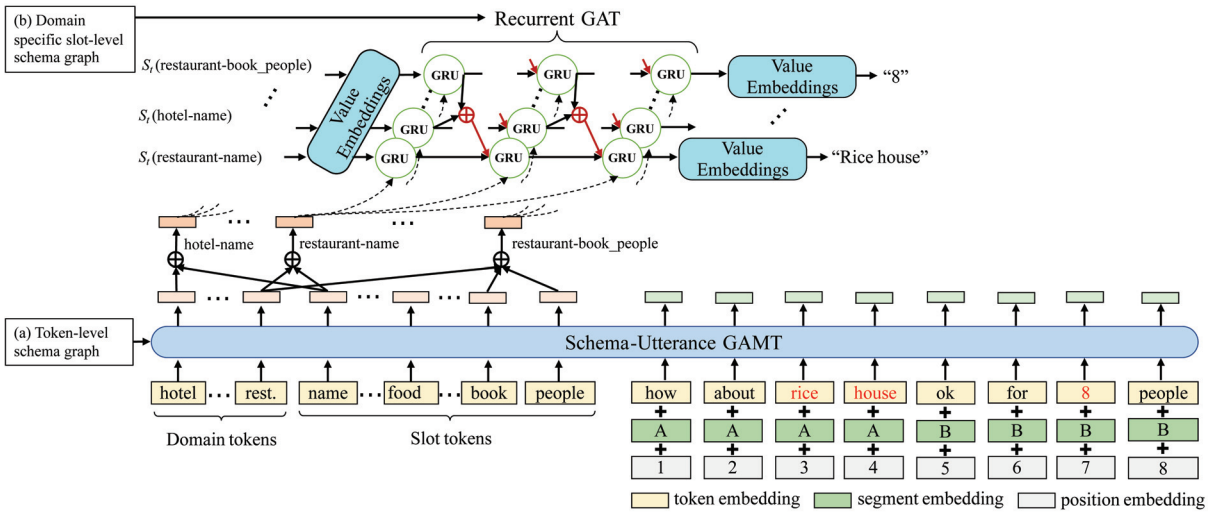


Figure 2: The architecture of the proposed SST model, which includes an embedding layer, a schema-utterance GAMT for context encoding, and a recurrent GAT for state updating. Two different schema graphs at token and slot levels are utilized in the model (see examples in Fig. 3).

Once obtained, $\hat{\mathbf{a}}_{ij}^l$ will be normalized with a *feature-wised multi-dimensional softmax* (MD-softmax) function, which results in a categorical distribution $\mathbf{a}_{ij}^l$ at feature level. Accordingly, Eq. (6) will be revised as follows,

$$\mathbf{e}_i^l = f_{agg}^l(\{\mathbf{m}_j^l | x_j \in \mathcal{N}_+(x_i)\}) = \sum_{x_j \in \mathcal{N}_+(x_i)} \mathbf{a}_{ij}^l \odot \mathbf{m}_j^l, \tag{9}$$

where $\odot$ represents element-wise product of two vectors.

Updating Embedding After aggregating messages, each node will update its embedding from $\mathbf{h}_i^{l-1}$ to $\mathbf{h}_i^l$,

$$\mathbf{h}_i^l = f_{ue}^l(\mathbf{e}_i^l, \mathbf{h}_i^{l-1}). \tag{10}$$

The updating function $f_{ue}^l(\cdot)$ can be a MLP function,

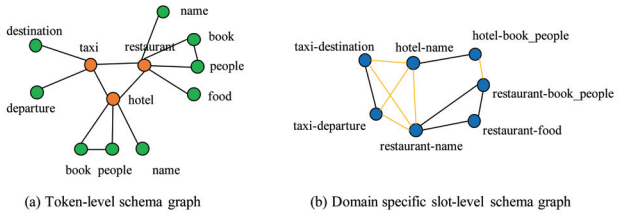
$$f_{ue}^l(\mathbf{e}_i^l, \mathbf{h}_i^{l-1}) = \mathbf{W}_{ue2}^l \sigma(\mathbf{W}_{ue1}^l \mathbf{e}_i^l + \mathbf{b}_{ue1}^l) + \mathbf{b}_{ue2}^l, \tag{11}$$

where $\mathbf{W}_{ue1}^l$, $\mathbf{W}_{ue2}^l$, $\mathbf{b}_{ue1}^l$ and $\mathbf{b}_{ue2}^l$ are learnable parameters. It's notable that $\mathbf{h}_i^{l-1}$ is not directly used in above MLP function, because $\mathbf{e}_i^l$ has already included message from x_i itself. In order to make training stable, we employ a residual connection around two steps/layers, followed by layer normalization, i.e. LayerNorm$(\mathbf{h}_i^l + \mathbf{h}_i^{l-1})$.

After updating node embedding L steps, we will obtain the *context-aware* embedding $\mathbf{h}_i^L$ for each node x_i.

3 Method

In a multi-domain dialogue system, there are a set of domains $\mathcal{D}$ that users and the system can converse about. For each domain $d \in \mathcal{D}$, there are n_d slots. Each slot s corresponds to a specific aspect of the user intent (e.g. *price*) and can take a value (e.g. *cheap*) from a candidate value set defined by a domain ontology. The dialogue state S can be defined as a set of slot-value pairs, e.g. {*price=cheap, area=west*}.



(a) Token-level schema graph (b) Domain specific slot-level schema graph

Figure 3: Two different schema graphs at token and slot levels.

At t-th dialogue turn, the dialogue state is S_t, which is used as a constraint to frame a database query. Based on the results of database query and S_t, the systems give a response U_{t+1}^{sys} to the user. Then the user inputs a new sentence U_{t+1}^{usr}. A state tracker then updates the dialogue state from S_t to S_{t+1} according to U_{t+1}^{sys} and U_{t+1}^{usr}. The whole dialogue process can be represented as $\{S_0, U_1^{sys}, U_1^{usr}, S_1, U_2^{sys}, U_2^{usr}, S_2, \cdots, S_{T-1}, U_T^{sys}, U_T^{usr}, S_T\}$.

Traditionally, lots of DST models predict dialogue state according to the whole dialogue context up to date. They do not explicitly model the dialogue state update process. In contrast, our proposed SST model explicitly updates dialogue state from S_t to S_{t+1} depending on U_{t+1}^{sys} and U_{t+1}^{usr}, i.e.

$$S_{t+1} = f_{sst}(S_t, U_{t+1}^{sys}, U_{t+1}^{usr}). \tag{12}$$

Our SST model consists of two modules: ontology schema and utterance matching module and state updating module. As shown in Fig. 2, the first module learns the context representation for each token in domain schema and utterance and harvests useful information from each other. The second module updates dialogue state from S_t to S_{t+1} according to slot-specified information obtained by the first module.

3.1 Ontology Schema and Utterance Matching with GAMT

In previous work, ontology schema information is not fully utilized in learning dialogue utterance representation. Here we propose graph attention matching networks (GAMTs) to learn the representations of ontology schema and dialogue utterance simultaneously.

We first define a token-level schema graph $G^1 = (V^1, E^1)$ according to the original ontology scheme. An example is shown in Fig. 3(a). The graph nodes consist of all domain tokens, e.g. *taxi, hotel, restaurant*, and all slot tokens, e.g. *name, food, destination*. There are edges between slots and the domain which the slots belong to. If some slots/domain is described with more then one word, e.g. *book_people*, it will be divided into single tokens, e.g. *book, people*, and there is an edge between them.

We define another graph $G^2 = (V^2, E^2)$ according to the dialogue utterance. The graph nodes consist of all words in the latest system response and user utterance pair $(U^{sys}_{t+1}, U^{usr}_{t+1})$. All nodes in the graph are connected.

In GAMT, for each node $x^k_i \in V^k (k = 1, 2)$, the embedding $\mathbf{h}^{k,l}_i$ is updated at each step to take into account not only the aggregated messages from its neighbors but also the cross-graph messages from another graph. The input feature of each node in G^1 is the corresponding token embedding, and the input feature of each node in G^2 is the sum of the token embedding, the segmentation embedding, and the position embedding as shown in Fig. 2. Here we use pre-trained word-embedding as the token embedding and random initialized embedding as the segmentation embedding. They are updated during the training process. We use sine and cosine functions of different frequencies as the position embedding (Vaswani et al. 2017). Compared with GAT, here the process of updating node embedding is revised as follows. Without loss of generality, we will use nodes in G^1 to describe the update process, and the update process for nodes in G^2 is similar.

Sending Messages At l-th step, each node x^1_i in graph G^1 will not only send a message $\mathbf{m}^{1\to1,l}_i$ to all nodes $x^1_j \in \mathcal{N}^1_+(x^1_i)$,

$$\mathbf{m}^{1\to1,l}_i = f^{self,l}_{msg}(\mathbf{h}^{1,l-1}_i), \tag{13}$$

but also send a message to all nodes in graph G^2:

$$\mathbf{m}^{1\to2,l}_i = f^{cross,l}_{msg}(\mathbf{h}^{1,l-1}_i), \tag{14}$$

where the functions $f^{self,l}_{msg}(\cdot)$ and $f^{cross,l}_{msg}(\cdot)$ are different linear transformations.

Aggregating Messages After sending messages, each node x^1_i will aggregate messages from $\mathcal{N}^1_+(x^1_i)$ and V^2 respectively,

$$\begin{aligned}\mathbf{e}^{1\to1,l}_i &= f^{self,l}_{agg}(\{\mathbf{m}^{1\to1,l}_j | x^1_j \in \mathcal{N}^1_+(x^1_i)\}),\\ \mathbf{e}^{2\to1,l}_i &= f^{cross,l}_{agg}(\{\mathbf{m}^{2\to1,l}_j | x^2_j \in V^2\}).\end{aligned} \tag{15}$$

Here we also use mutli-dim attention (Eq. 9) as the aggregating functions $f^{self,l}_{agg}(\cdot)$ and $f^{cross,l}_{agg}(\cdot)$. The parameters of two functions are different.

Updating Embedding After aggregating messages, each node will update its embedding from $\mathbf{h}^{l-1}_i$ to $\mathbf{h}^l_i$,

$$\mathbf{h}^l_i = f^l_{ue}([\mathbf{e}^{1\to1,l}_i, \mathbf{e}^{2\to1,l}_i], \mathbf{h}^{l-1}_i), \tag{16}$$

where $[\cdot,\cdot]$ is the concatenation of two vectors. The function $f^l_{ue}(\cdot)$ is a MLP as shown in Eq. (11).

After updating node embedding L_1 steps with GAMT, we can obtain contextual representations $\mathbf{h}^{1,L_1}_i$ for each node x^1_i in G^1 and $\mathbf{h}^{2,L_1}_j$ for each node x^2_j in G^2. $\mathbf{h}^{1,L_1}_i$ includes not only information from the related tokens (i.e. neighbors) in ontology schema, but also useful information from dialogue utterance. Similarly, $\mathbf{h}^{2,L_1}_j$ includes not only contextual information in dialogue utterance, but also useful information from ontology schema.

We then obtain slot representation vector $\mathbf{g}_s$ based on $\mathbf{h}^{1,L_1}$ for each slot s. Assuming a slot consists of one or more tokens in G^1, a feature-wised score vector is calculated with an MLP for each token, and then normalized with MD-softmax. Finally, the node embeddings $\mathbf{h}^{1,L_1}_i$ for these tokens are weighted with the normalized scores to obtain $\mathbf{g}_s$.

3.2 Dialogue State Updating with RGAT

The state updating module takes $\mathbf{g}_{s_i}$ as input, and updates the dialogue state for each slot s_i. As introduced in Section 1, there may be underlying relations between slots in multi-domain dialogue state tracking. In order to capture the interaction between slots, we propose to use recurrent attention graph neural networks (RGAT) for state updating.

We first define a domain-specific slot-level schema graph G. Fig. 3(b) is an example. The nodes of G consist of all slots, e.g. *hotel-name, taxi-destination*. There is an edge between two slots if they belong to the same domain. If two slots belong to different domains but some of their candidate values are the same, there is also an edge between them. For example, in Fig. 3(b), there is edge between *hotel-name* and *taxi-destination*, because the taxi destination may be a hotel.

The input feature of each node x_i is the embedding vector $\mathbf{v}_i$ of value for the corresponding slot s_i at t-th turn. RGAT takes $\mathbf{v}_i$ as the initial embedding $\mathbf{h}^0_i$ for node x_i (i.e. slot s_i), then updates its embedding from one step to the next similar to GAT. However, the most significant difference between GAT and RGAT is that the parameters of RGAT in all steps are shared, and their node embedding updating functions are different.

Sending Messages At l-th step, each node x_i will send its embedding $\mathbf{h}^{l-1}_i$ as a message $\mathbf{m}^l_i$ to its neighbours $x_j \in \mathcal{N}(x_i)$,

$$\mathbf{m}^l_i = \mathbf{h}^{l-1}_i. \tag{17}$$

Aggregating Messages After sending messages, each node x_i will aggregate messages from all neighbours $x_j \in \mathcal{N}(x_i)$,

$$\mathbf{e}^l_i = f_{agg}(\{\mathbf{m}^l_j | x_j \in \mathcal{N}(x_i)\}). \tag{18}$$

Here the multi-dim attention (Eq. 9) is used as the aggregating function $f_{agg}(\cdot)$. Different from that in GAT and GAMT, the parameters of the function are shared across all steps.

Updating Embedding After aggregating messages, each node will update its embedding from $\mathbf{h}_i^{l-1}$ to $\mathbf{h}_i^l$,

$$\mathbf{h}_i^l = f_{ue}(\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}), \tag{19}$$

where the function $f_{ue}(\cdot)$ is an improved GRU cell:

$$\begin{aligned}
\hat{\mathbf{z}}_{i,1}^l &= \mathbf{W}_{z,1}[\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}] + \mathbf{b}_{z,1}, \\
\hat{\mathbf{z}}_{i,2}^l &= \mathbf{W}_{z,2}[\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}] + \mathbf{b}_{z,2}, \\
\hat{\mathbf{z}}_{i,3}^l &= \mathbf{W}_{z,3}[\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}] + \mathbf{b}_{z,3}, \\
\mathbf{z}_{i,1}^l, \mathbf{z}_{i,2}^l, \mathbf{z}_{i,3}^l &= \text{MD-softmax}(\hat{\mathbf{z}}_{i,1}^l, \hat{\mathbf{z}}_{i,2}^l, \hat{\mathbf{z}}_{i,3}^l), \\
\hat{\mathbf{r}}_{i,1}^l &= \text{sigmoid}(\mathbf{W}_{r,1}[\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}] + \mathbf{b}_{r,1}), \\
\mathbf{r}_{i,2}^l &= \text{sigmoid}(\mathbf{W}_{r,2}[\mathbf{e}_i^l, \mathbf{h}_i^{l-1}, \mathbf{g}_{s_i}] + \mathbf{b}_{r,2}), \\
\tilde{\mathbf{h}}_i^l &= \tanh(\mathbf{W}_h[\mathbf{g}_{s_i}, \mathbf{r}_{i,1}^l \odot \mathbf{e}_i^l, \mathbf{r}_{i,2}^l \odot \mathbf{h}_i^{l-1}]), \\
\mathbf{h}_i^l &= \mathbf{z}_{i,1}^l \odot \tilde{\mathbf{h}}_i^l + \mathbf{z}_{i,2}^l \odot \mathbf{h}_i^{l-1} + \mathbf{z}_{i,3}^l \odot \mathbf{e}_i^l,
\end{aligned} \tag{20}$$

where MD-softmax is the feature-wise multi-dimension softmax function. $\mathbf{z}_{i,1}$, $\mathbf{z}_{i,2}$, and $\mathbf{z}_{i,3}$ are three gates that control information flow. In particular, $\mathbf{z}_{i,1}$ controls information from the latest dialogue context. If $\mathbf{z}_{i,1}$ is $\mathbf{1}$, then the value of corresponding slot s_i will change to new value. $\mathbf{z}_{i,2}$ controls information from its last dialogue state. If $\mathbf{z}_{i,2}$ is $\mathbf{1}$, then the value of corresponding slot s_i will remain unchanged. $\mathbf{z}_{i,3}$ controls information from its neighbours. If $\mathbf{z}_{i,3}$ is $\mathbf{1}$, then the value of corresponding slot s_i will be copied from some value of its neighbours. All parameters $\mathbf{W}$ and $\mathbf{b}$ are shared across steps.

After updating node embedding L_2 steps, we will obtain the embedding $\mathbf{h}_i^{L_2}$ for each node x_i. It is the new hidden state of slot s_i. We calculate the dot product between the hidden vector and each of the embedding vectors of the candidate values. The softmax function is performed with the results to give a distribution of probabilities $\mathbf{p}_{t+1,i}$ for all candidate values, i.e.

$$\mathbf{p}_{t+1,i} = \text{softmax}(\mathbf{E}_v \mathbf{h}_i^{L_2}), \tag{21}$$

where $\mathbf{E}_v$ is the value embedding matrix. Each row corresponds to an embedding vector for a candidate value. $\mathbf{E}_v$ is initialized with pre-trained word embedding. If a slot value consists of more than one word, the initialized embedding is the concatenation of the mean-pooling and max-pooling of all words. The embedding matrix is updated during training. When training the whole DST model, we minimize the cross-Entropy (CE) loss between the predicted probabilities and the given label for all slots.

3.3 General SST models

In previous sections, our SST model only takes dialogue context in the latest turn (i.e. latest user utterance and system response) into consideration for each time. It can be naturally extended to encode dialogue context with more turns, i.e. the DST formula in Eq. (12) can be reformed as follows,

$$S_t = f_{sst\text{-}k}(S_{t-k}, U_{t-k+1}^{sys}, U_{t-k+1}^{usr}, \cdots, U_t^{sys}, U_t^{usr}), \tag{22}$$

where k is the number of selected dialogue turns, and the corresponding SST model is refered to SST-k.

4 Experiments

4.1 Setup

Data We choose MultiWOZ 2.0 (Budzianowski et al. 2018) and MultiWOZ 2.1 (Eric et al. 2019) as our training and evaluation datasets, both of which are task-oriented corpora comprised of dialogues between human and human.

Unlike some common datasets such as DSTC2 which merely focuses on the restaurant search domain, there are 35 slots in MultiWOZ 2.0 with nearly 2000 candidate values over seven domains. Following previous work TRADE (Wu et al. 2019), we only use five of them since the other two domains barely appear, and they are neither in the development set nor in the test set. Besides, it is worth mentioning that there are a certain number of error annotations in this dataset. Firstly, the values in the ontology and the dialog context are not always the same. For example, the value in the context is *eastern* but ontology uses *east* instead. Secondly, delayed markups can sometimes be found in the process of dialog state updates, which exert an influence on model training.

To alleviate the problems mentioned above, MultiWOZ 2.1 is released recently. More than 32% of state annotations have been modified, especially in name slots. It contains 55718 turn-level training samples from 8133 dialogues (only five domains) with 7368 turns of the test set.

Hyperparameters We use pretrained GloVe embeddings (Pennington, Socher, and Manning 2014) and character embeddings (Hashimoto et al. 2017) to represent words both in the dialogue turns and the slot tokens. Then a dense layer with the output size of 180 has been adopted, and the size of all hidden layers is 180. The number of GAMT steps/layers L_1 is three on both datasets. Each model is trained by RMSProp with a learning rate of 0.0001 and a batch size of 32.

4.2 Main Results

Table 1 shows our results on both MultiWOZ 2.0 and MultiWOZ 2.1 test sets. Here joint goal accuracy is used as the evaluation metric. We compare our model with several previous baselines.

The existing models can be divided into two categories, the classification-based models and the generative models. The classification-based ones include: MDBT (Ramadan, Budzianowski, and Gasic 2018), which uses multiple bi-LSTMs to encode utterances of system and user; GLAD (Zhong, Xiong, and Socher 2018), a global-local self-attention model; GCE (Nouri and Hosseini-Asl 2018), a global-conditioned encoder; HJST (Eric et al. 2019) and FJST (Eric et al. 2019), which refer to the Flat Joint State Tracker and the Hierarchical Joint State Tracker respectively. and SUMBT (Lee, Lee, and Kim 2019), a slot-utterance matching belief tracker. To break through the restraint of ontology values, some generative models are proposed. For example, Neural Reading DST model (Gao et al. 2019), which utilizes an attention-based neural network to point to the slot values in the utterances; HyST (Goel, Paul, and Hakkani-Tür 2019), a hybrid joint state tracking model learning the optimal method for each slot type, and TRADE

Table 1: Joint goal accuracy on MultiWOZ 2.1 and MultiWOZ 2.0 test set vs. various approaches as reported in the literature.

DST Models	Joint Acc. MultiWOZ 2.1	Joint Acc. MultiWOZ 2.0
MDBT	-	15.57
GLAD	-	35.57
GCE	-	36.27
HJST	35.55	38.4
FJST	38.0	40.2
SUMBT	-	46.65
Neural Reading DST	36.40	39.41
HyST	38.1	42.33
TRADE	45.6	48.6
SST-1	52.55	47.59
SST-2	**55.23**	**51.17**
SST-3	54.22	49.29

(Wu et al. 2019), a transferable dialogue state generator using a copy mechanism.

Our SSTs belong to classification-based models. Here SST-1, SST-2, and SST-3 are compared with these baselines. From the results, we can find that our best model SST-2 outperforms all baseline models and achieves the new state-of-the-art on both datasets.

Among our models, SST-2 performs best on both datasets. Especially, SST-2 and SST-3 perform much better than SST-1 on MultiWOZ 2.0. The reason is that there are some errors in labels, and one of the most common error types is *delayed markup*, i.e. slot values were annotated one or more turns after the value appeared in the user utterances. For SST-1, when delayed markup appears, SST-1 can't find the value in the last utterance. Therefore, using more dialogue context (e.g. SST-2, SST-3) will benefit training and prediction.

4.3 Analysis

Ablation Analysis Table 2 illustrates the ablation experiment for our SST-2 model. We find a ∼2% drop of performance when the graph relation is removed in RGAT. It shows that information exchange between slots is necessary. In Table 3, we present an example to show the effect of the graph in RGAT. We can see that the model with the graph correctly gets the dialogue state, even though the user does not offer *hotel-book_day* directly. The model makes the accurate reference of the phrase "*for the same day*" through the context, and the value of *train-day* is copied as the value of *hotel-book_day*. However, the model without graph can not predict the correct value of *hotel-book_day*.

We also investigate the effect of interaction between the ontology graph and the utterance. We remove the cross attention between them in all steps except the last step in GAMT and find a ∼1% drop of performance. It indicates that multiple times of interaction can make better alignment between ontology and utterance.

For the steps in RGAT, we tried the number from one to six to observe the change of model performance. We find that SST-2 with 2 and 4 steps achieves the best performance on MultiWOZ 2.0 and MultiWOZ 2.1, respectively. The result on MultiWOZ 2.0 is presented in Figure 4. It indicates that more than 1 step is needed for fusing the history state and the useful information in the current utterance.

Table 2: Ablation study.

Model	Joint Acc. MultiWOZ 2.1	Joint Acc. MultiWOZ 2.0
SST-2	55.23	51.17
- Graph in RGAT	52.84	49.67
- Interaction in GAMT	53.85	50.26

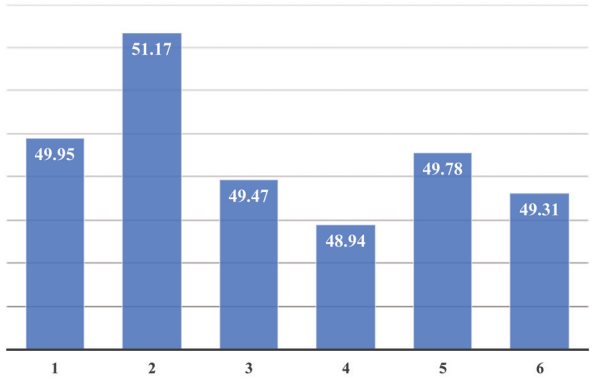


Figure 4: Results of different number of steps in RGAT on MultiWOZ 2.0 test set.

Error Analysis Domain-specific slot accuracy of SST-2 and TRADE [1] on MultiWOZ 2.0 test set is shown in Table 4. We find that our model can achieve higher accuracy than TRADE for most of the slots. However, in the taxi domain, the accuracy for most slots in this domain is lower than TRADE. The reason may be that the values of slots in the taxi domain are much longer (e.g. address, name), which are more suitable for generative models like TRADE.

5 Related Work

Dialogue State Tracking Traditional dialogue state tracking models rely on semantics extracted by natural language understanding to predict the current dialogue states (Young et al. 2013; Williams et al. 2013; Henderson, Thomson, and Williams 2014b; Yu et al. 2015; Sun et al. 2014b; 2014a; Xie et al. 2015; Yu et al. 2016), or jointly learn language understanding in an end-to-end way (Henderson, Thomson, and Young 2014a; 2014b). These methods heavily rely on hand-crafted features and complex domain-specific lexicons for delexicalisation, which are difficult to extend and scale to new domains.

Recently, most works about DST focus on encoding dialogue context with deep neural networks (such as CNN, RNN, LSTM-RNN, etc.) and predicting a value for each possible slot (Mrkšić et al. 2017; Xu and Hu 2018; Zhong, Xiong, and Socher 2018; Ren et al. 2018; Xie et al. 2018). For multi-domain DST, *slot-value* pairs are extended to

[1]The official code https://github.com/jasonwu0731/trade-dst is used.

Table 3: An example of coreference in multi-domain dialogue.

utterances	dialogue state		
	last dialogue state	predicted state (w/o graph)	predicted state (with graph)
sys: would you like to book a room at the acorn guest house ? **user**: yes please book it for 5 people and 4 nights starting from the same day .	train-destination: cambridge train-day: **monday** train-departure: stansted airport train-leaveat: 11:15 hotel-name: acorn guest house	hotel-book stay: 4 hotel-book day: **saturday** hotel-book people: 5 train-destination: cambridge train-day: **monday** train-departure: stansted airport train-leaveat: 11:15 hotel-name: acorn guest house	hotel-book stay: 4 hotel-book day: **monday** hotel-book people: 5 train-destination: cambridge train-day: **monday** train-departure: stansted airport train-leaveat: 11:15 hotel-name: acorn guest house

Table 4: Domain-specific slot accuracy.

Domain-slot name	SST-2	TRADE	Vocab size	**Domain-slot name**	SST-2	TRADE	Vocab size
hotel-pricerange	**0.9851**	0.9753	5	attraction-area	**0.9767**	0.9765	7
hotel-type	**0.9435**	0.9370	3	attraction-name	**0.9313**	0.9221	158
hotel-parking	**0.9761**	0.9693	4	attraction-type	**0.9741**	0.9638	17
hotel-book stay	0.9852	**0.9885**	9	restaurant-food	**0.9775**	0.9744	105
hotel-book day	0.9851	0.9851	7	restaurant-pricerange	**0.9726**	0.9696	4
hotel-book people	0.9819	**0.9834**	10	restaurant-area	**0.9730**	0.9699	6
hotel-area	**0.9779**	0.9719	14	restaurant-name	**0.9133**	0.9116	197
hotel-stars	**0.9857**	0.9800	7	restaurant-book time	0.9837	**0.9849**	68
hotel-internet	**0.9868**	0.9695	3	restaurant-book day	**0.9879**	0.9860	10
hotel-name	0.9213	**0.9344**	100	restaurant-book people	**0.9887**	0.9885	8
train-destination	**0.9776**	0.9761	33	taxi-leaveat	**0.9845**	0.9844	129
train-day	**0.9849**	0.9774	11	taxi-destination	0.9473	**0.9686**	299
train-departure	0.9738	**0.9758**	42	taxi-departure	0.9530	**0.9682**	288
train-arriveby	**0.9841**	0.9761	120	taxi-arriveby	0.9829	**0.9885**	106
train-book people	0.9746	**0.9764**	13	train-leaveat	**0.9673**	0.9623	157

domain-slot-value pairs for target (Ramadan, Budzianowski, and Gasic 2018; Gao et al. 2019; Wu et al. 2019). However, they do not explicitly consider slot relations, which may mitigate the data sparsity problem. They also predict the value for each slot independently, which can not capture label dependency. Beyond DST, relations and similarities between slots are also utilized in language understanding (Zhu and Yu 2018; Zhao, Zhu, and Yu 2019b), while it lacks an interaction mechanism between slots and utterances.

Graph Neural Network Recently, there has been a surge of interest in Graph Neural Network (GNN) approaches for representation learning of graphs (Scarselli et al. 2009; Veličković et al. 2018). They can aggregate information from graph structure and encode node features, which can be exploited to learn to reason and introduce unordered structure information. Many GNN variants are proposed and also applied in various NLP tasks, such as text classification (Yao, Mao, and Luo 2019), machine translation (Marcheggiani, Bastings, and Titov 2018), dialogue policy optimization (Chen et al. 2018b; 2019; 2018a) etc. We are the first to introduce GNN in DST to the best of our knowledge.

6 Conclusion

We introduce a schema-guided multi-domain dialogue state tracker with graph attention networks, which involves slot relations and learns deep feature representations for each slot dependently. All slots from different domains and their relations organized as schema graphs. Graph attention matching network and recurrent graph attention network are proposed to fully encode dialogue utterances, schema graphs, and previous dialogue states. Our approach achieves state-of-the-art joint goal accuracy on both MultiWOZ 2.0 and 2.1 benchmarks. In this paper, the schema is automatically constructed according to onology. In the future, we will investigate to use more complex schema (Rastogi et al. 2019). We will also exploit generative models for value prediction, which could enable the tracker to generate values out of domain ontology. To overcome the data sparsity problem, we would like to try data augmentation methods (Zhao, Zhu, and Yu 2019a; Cao et al. 2019).

Acknowledgments

This work has been supported by the National Key Research and Development Program of China (Grant No. 2017YFB1002102) and the China NSFC projects (No. 61573241). We thank the anonymous reviewers for their thoughtful comments and efforts towards improving this manuscript.

References

Budzianowski, P.; Wen, T.-H.; Tseng, B.-H.; Casanueva, I.; Ultes, S.; Ramadan, O.; and Gašić, M. 2018. MultiWOZ - a large-scale multi-domain wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of EMNLP*, 5016–5026.

Cao, R.; Zhu, S.; Liu, C.; Li, J.; and Yu, K. 2019. Semantic parsing with dual learning. In *Proceedings of ACL*, 51–64.

Chen, L.; Chang, C.; Chen, Z.; Tan, B.; Gašić, M.; and Yu, K. 2018a. Policy adaptation for deep reinforcement learning-based dialogue management. In *Proceedings of ICASSP*, 6074–6078. IEEE.

Chen, L.; Tan, B.; Long, S.; and Yu, K. 2018b. Structured dialogue policy with graph neural networks. In *Proceedings of COLING*, 1257–1268.

Chen, L.; Chen, Z.; Tan, B.; Long, S.; Gasic, M.; and Yu, K. 2019. Agentgraph: Towards universal dialogue management with structured deep reinforcement learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27(9):1378–1391.

Eric, M.; Goel, R.; Paul, S.; Sethi, A.; Agarwal, S.; Gao, S.; and Hakkani-Tur, D. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Gao, S.; Sethi, A.; Aggarwal, S.; Chung, T.; and Hakkani-Tur, D. 2019. Dialog state tracking: A neural reading comprehension approach. *arXiv preprint arXiv:1908.01946*.

Goel, R.; Paul, S.; and Hakkani-Tür, D. 2019. HyST: A hybrid approach for flexible and accurate dialogue state tracking. *arXiv preprint arXiv:1907.00883*.

Hashimoto, K.; Tsuruoka, Y.; Socher, R.; et al. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of EMNLP*, 1923–1933.

Henderson, M.; Thomson, B.; and Williams, J. D. 2014a. The second dialog state tracking challenge. In *Proceedings of SIGDial*, 263–272.

Henderson, M.; Thomson, B.; and Williams, J. D. 2014b. The third dialog state tracking challenge. In *Proceedings of IEEE SLT*.

Henderson, M.; Thomson, B.; and Young, S. 2014a. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *Proceedings of IEEE SLT*.

Henderson, M.; Thomson, B.; and Young, S. 2014b. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of SIGDial*, 292–299.

Lee, H.; Lee, J.; and Kim, T.-Y. 2019. SUMBT: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of ACL*, 5478–5483.

Marcheggiani, D.; Bastings, J.; and Titov, I. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. *arXiv preprint arXiv:1804.08313*.

Mrkšić, N.; Séaghdha, D. Ó.; Wen, T.-H.; Thomson, B.; and Young, S. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of ACL*, 1777–1788.

Nouri, E., and Hosseini-Asl, E. 2018. Toward scalable neural dialogue state tracking model. *arXiv preprint arXiv:1812.00899*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, 1532–1543.

Ramadan, O.; Budzianowski, P.; and Gasic, M. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of ACL*, 432–437.

Rastogi, A.; Zang, X.; Sunkara, S.; Gupta, R.; and Khaitan, P. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.

Ren, L.; Xie, K.; Chen, L.; and Yu, K. 2018. Towards universal dialogue state tracking. In *Proceedings of EMNLP*, 2780–2786.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20(1):61–80.

Shen, T.; Zhou, T.; Long, G.; Jiang, J.; Pan, S.; and Zhang, C. 2018. DiSAN: Directional self-attention network for rnn/cnn-free language understanding. In *Proceedings of AAAI*.

Sun, K.; Chen, L.; Zhu, S.; and Yu, K. 2014a. A generalized rule based tracker for dialogue state tracking. In *Proceedings of IEEE SLT*.

Sun, K.; Chen, L.; Zhu, S.; and Yu, K. 2014b. The SJTU system for dialog state tracking challenge 2. In *Proceedings of SIGDial*, 318–326.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Proceedings of NIPS*, 6000–6010.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In *Proceedings of ICLR*.

Williams, J.; Raux, A.; Ramachandran, D.; and Black, A. 2013. The dialog state tracking challenge. In *Proceedings of SIGDial*, 404–413.

Wu, C.-S.; Madotto, A.; Hosseini-Asl, E.; Xiong, C.; Socher, R.; and Fung, P. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of ACL*.

Xie, Q.; Sun, K.; Zhu, S.; Chen, L.; and Yu, K. 2015. Recurrent polynomial network for dialogue state tracking with mismatched semantic parsers. In *Proceedings of SIGDial*, 295–304.

Xie, K.; Chang, C.; Ren, L.; Chen, L.; and Yu, K. 2018. Cost-sensitive active learning for dialogue state tracking. In *Proceedings of SIGDial*, 209–213.

Xu, P., and Hu, Q. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of ACL*.

Yao, L.; Mao, C.; and Luo, Y. 2019. Graph convolutional networks for text classification. In *Proceedings of AAAI*, 7370–7377.

Young, S.; Gašić, M.; Thomson, B.; and Williams, J. D. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE* 101(5):1160–1179.

Yu, K.; Chen, L.; Chen, B.; Sun, K.; and Zhu, S. 2014. Cognitive technology in task-oriented dialogue systems: Concepts, advances and future. *Chinese Journal of Computers* 37(18):1–17.

Yu, K.; Sun, K.; Chen, L.; and Zhu, S. 2015. Constrained markov bayesian polynomial for efficient dialogue state tracking. *IEEE/ACM Transactions on Audio, Speech and Language Processing* 23(12):2177–2188.

Yu, K.; Chen, L.; Sun, K.; Xie, Q.; and Zhu, S. 2016. Evolvable dialogue state tracking for statistical dialogue management. *Frontiers of Computer Science* 10(2):201–215.

Zhao, Z.; Zhu, S.; and Yu, K. 2019a. Data augmentation with atomic templates for spoken language understanding. In *Proceedings of EMNLP-IJNLP*, 3628–3634.

Zhao, Z.; Zhu, S.; and Yu, K. 2019b. A hierarchical decoding model for spoken language understanding from unaligned data. In *Proceedings of ICASSP*, 7305–7309.

Zhong, V.; Xiong, C.; and Socher, R. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of ACL*, 1458–1467.

Zhu, S., and Yu, K. 2018. Concept transfer learning for adaptive language understanding. In *Proceedings of SIGDial*, 391–399.