# Zero-Shot State Tracking and User Adoption Tracking on Schema-Guided Dialogue

**Shuyu Lei**[1,2], **Shuaipeng Liu**[1], **Mengjun Sen**[1], **Huixing Jiang**[1], **Xiaojie Wang**[2]
[1]Meituan-dianping Group, Beijing, China
[2]Beijing University of Posts and Telecommunication, Beijing, China
{leishuyu, xjwang}@bupt.edu.cn,
{liuhuaipeng, shenmengjun, jianghuixing}@meituan.com

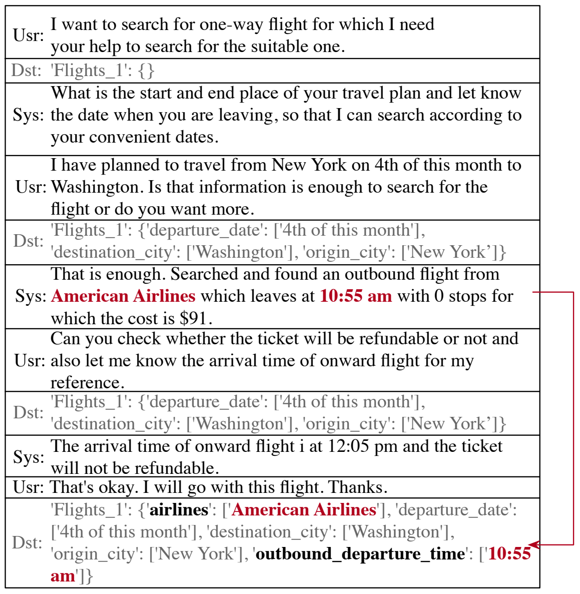
## Abstract

The schema-guided dialogue state tracking as one track in DSTC8 has several challenges. The foremost challenge is that some domains have little to no training data available. The second challenge is that the system offered slots are updated to the state spanning multiple utterances. Thus, the traditional methods only encoded the last two utterances that may ignore such offered slots. To address the above issues, this paper presents a pipeline state tracking model contained three parts: (*i*) a zero-shot categorical state tracker is modelled to track the categorical slots, (*ii*) a zero-shot free-form state tracker is modelled to track free-form slots, (*iii*) an adoption state tracker is modelled to detect in which turn the system offered slots are adopted by the user. The proposed pipeline model fine-tuned on the BERT-Base model achieves 73.03% joint goal accuracy (the 5th place) and 92.49% average goal accuracy (the 3rd place) in the schema-guided dialogue state tracking challenge.

## Introduction

Virtual assistants such as Google Assistant, Siri, Alexa aim to help user accomplish specific goals by a natural language interface. In these virtual assistants, dialogue state tracking estimates the user goals based on the conversation history and applies these maintained goals to call services/APIs to help user accomplish specific task. Thus, dialogue state tracker plays a vital role in virtual assistants.

In recent studies (Henderson, Thomson, and Young 2014; Mrkšić et al. 2017; Zhong, Xiong, and Socher 2018; Wu et al. 2019; Gao et al. 2019), deep learning based approaches for dialogue state tracking have shown promising results. This line of work has been facilitated by the dialogue corpora such as DSTC2 (Williams et al. 2013), WOZ (Wen et al. 2017) and MultiWOZ (Budzianowski et al. 2018). However, prior dialogue corpora only cover few domains and define a single static API per domain, which ignore that multiple services with heterogeneous interfaces has overlapping functionality in the real world. To study on aforementioned issues, the schema-guided dialogue dataset is proposed in (Rastogi et al. 2019). This dataset is designed to serve as an



| | |
|---|---|
| Usr: | I want to search for one-way flight for which I need your help to search for the suitable one. |
| Dst: | 'Flights_1': {} |
| Sys: | What is the start and end place of your travel plan and let know the date when you are leaving, so that I can search according to your convenient dates. |
| Usr: | I have planned to travel from New York on 4th of this month to Washington. Is that information is enough to search for the flight or do you want more. |
| Dst: | 'Flights_1': {'departure_date': ['4th of this month'], 'destination_city': ['Washington'], 'origin_city': ['New York']} |
| Sys: | That is enough. Searched and found an outbound flight from **American Airlines** which leaves at **10:55 am** with 0 stops for which the cost is $91. |
| Usr: | Can you check whether the ticket will be refundable or not and also let me know the arrival time of onward flight for my reference. |
| Dst: | 'Flights_1': {'departure_date': ['4th of this month'], 'destination_city': ['Washington'], 'origin_city': ['New York']} |
| Sys: | The arrival time of onward flight i at 12:05 pm and the ticket will not be refundable. |
| Usr: | That's okay. I will go with this flight. Thanks. |
| Dst: | 'Flights_1': {'**airlines**': ['**American Airlines**'], 'departure_date': ['4th of this month'], 'destination_city': ['Washington'], 'origin_city': ['New York'], '**outbound_departure_time**': ['**10:55 am**']} |

Figure 1: An example of dialogue state tracking in a conversation. The red arrows on the right are the case that the offered slots from the system are updated to the state based on user adoption spanning multiple utterances. (Usr: user, Sys: system, Dst: dialogue state tracking)

effective testbed for intent prediction, slot filling, state tracking in large-scale virtual assistants. In this dataset, slots are categorized into two types: categorical slot (i.e. slot takes one of a finite set of possible values) and free-form slot (i.e. slot can take any string value). These descriptions of two types of slots are present in a guided schema. This schema-guided dialogue dataset, containing over 16k multi-domain conversations spanning 16 domains, has several challenges.

The foremost challenge is that constantly increasing number of services over a large number of domains produce unseen services and domains. Prior studies (Mrkšić et al. 2017; Zhong, Xiong, and Socher 2018; Gao et al. 2019; Zhang et al. 2019) only focus on the domains where training data is available. The second challenge is that the offered slots from the system are updated to the state by user adoption spanning multiple utterances as shown in Figure 1. The slots such as "airlines" and "outbound_departure_time" are offered by the system after calling an API and expected to be updated to the state. Then the user may request more information about recommended item and adopt it in the next few utterances such as "That's okay. I will go with this flight. Thanks.". Since these cases are not present in MultiWOZ dataset, previous studies (Mrkšić et al. 2017; Zhong, Xiong, and Socher 2018; Zhang et al. 2019) on MultiWOZ pay less attention on such update.

To tackle the aforementioned challenges, we emphasize that the models for state tracking should have ability to generalize in zero-shot settings. Additionally, a model should detect in which turn the offered slots from the system are adopted by the user.

In this paper, we propose a pipeline state tracking model contained three parts: a zero-shot categorical state tracker, a zero-shot free-form state tracker, and an adoption state tracker. The zero-shot categorical state tracker is applied to track the categorical slots. The zero-shot free-form state tracker is modelled to track free-form slots. The adoption state tracker is modelled to detect in which turn the offered slots from the system are adopted by user. The proposed pipeline model only fine-tuned on BERT-Base model achieves 73.03% joint goal accuracy (i.e. the 5th place) and 92.49% average goal accuracy (i.e. the 3rd place) in the challenge. To summary, our main contributions in this work are two-folds:

- We propose a zero-shot categorical state tracker and a zero-shot free-form state tracker to tackle the zero-shot challenge. The zero-shot categorical state tracker is modelled as a binary classifier that infers whether a possible value for given slot is provided in last two utterances or not. During inference, the possible values for given slot are ranked based on the modelled classifier. The zero-shot free-form state tracker is modelled to point out the slot values from utterances as the method in machine comprehension (i.e. slot description as question and user utterance as passage).
- We propose an adoption state tracker to detect in which turn the offered slots from the system are adopted by the user, which is modelled as binary classifier by fine-tuning an individual BERT model.

## Related work

Dialogue state tracking (DST) is a core component in task-oriented dialogue systems. Traditional DST approaches usually rely on hand-crafted features or domain-specific lexicons (Henderson, Thomson, and Young 2014; Wen et al. 2017), which are difficult to extend to new domains. Recent data-driven deep learning methods for DST have achieved promising results. Previously, DST methods assume a fixed ontology in which all slots and their possible values are known, then these methods perform classification over the set of all possible values of a slot or individually score all possible slot values (Mrkšić et al. 2017; Zhong, Xiong, and Socher 2018; Ren et al. 2018; Lee, Lee, and Kim 2019). In reality, however, obtaining the list of all possible values taken by some slots is not feasible because this could be very large (restaurant name, city etc.), unbounded (date, time, username etc.) or dynamic (movie, song etc.). Hence, such approaches are not practical for deployment in virtual assistants operating over real-world services. To tackle the aforementioned issues, a class of methods derived candidate slot values from either a predefined ontology or by extraction of a word or n-grams in the dialog context (Rastogi, Hakkani-Tür, and Heck 2017; Goel et al. 2018). Another type of approaches formulated DST as an extractive machine reading comprehension problem in which the answer of the question is a text span in the passage. These methods extract slot values through span matching with start and end positions in the dialog context (Xu and Hu 2018; Gao et al. 2019). Concretely, (Gao et al. 2019) first use a slot carryover model to decide whether to carryover a slot value from the last turn, then it predicts the slot value as a span of tokens within the dialog if the slot carryover model predicts label "span". However, it is hard to find a particular string in the dialogue context due to the diversity of value descriptions. For instance, the ontology has moderate but the dialog context has moderately. To address this issue, (Goel, Paul, and Hakkani-Tür 2019) introduce a hybrid approach (HyST) which learns the appropriate method for each slot type (i.e. open and closed vocabulary), and (Zhang et al. 2019) proposed a Dual-Strategy for DST (DS-DST) which track each slot by classifying over a candidate-value list defined in the ontology and finding text spans in the dialogue context. The most similar technique to our work is the DS-DST. In contrast, we design a zero-shot categorical state tracker and a zero-shot free-form state tracker for tracking categorical slots and non-categorical slots defined in the service schema, respectively. Besides, DS-DST only apply last two utterances may discard such offered slots from the system, while our work propose an adoption state tracker additionally to detect the user adoption for the offered slots from the system.

Pre-trained models such as BERT (Devlin et al. 2019), XLnet (Yang et al. 2019) and RoBERTa (Liu et al. 2019) have showed promising performances on many down-stream tasks. Recent studies (Lee, Lee, and Kim 2019; Chao and Lane 2019; Zhang et al. 2019) that utilize BERT to encode the dialogue history or slot information have achieved considerable improvements. Among them, SUMBT (Lee, Lee, and Kim 2019) employs BERT to extract representations of candidate values. BERT-DST (Chao and Lane 2019) adopts BERT to encode the inputs of the user turn as well as the previous system turn. (Zhang et al. 2019) utilizes a fixed BERT model to output the representations of each candidate slot value and use the other fine-tuned BERT to encode the domain-slot types and dialog contexts. Considering these pre-trained models may help the model to generalize to unseen services and domains. Thus, our proposed pipeline state tracking model is based on the pre-trained model (i.e. BERT (Devlin et al. 2019)) to achieve the generalization performance in zero-shot settings.

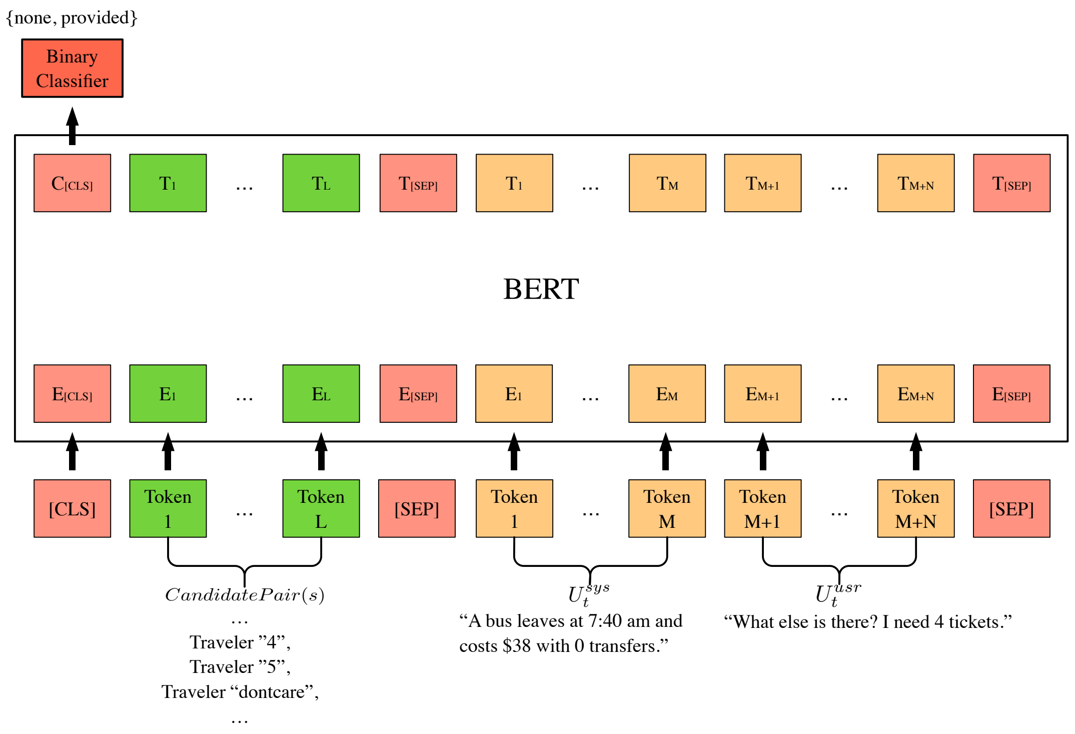


Figure 2: Zero-shot categorical state tracker.

## Models

The proposed pipeline dialogue state tracking model is introduced in this section, which comprises three parts: a zero-shot categorical state tracker, a zero-shot free-form state tracker, and an adoption state tracker. The zero-shot categorical state tracker is modelled to track the categorical slots. The zero-shot free-form state tracker is modelled to track free-form slots. The adoption state tracker is modelled to determine in which turn the offered slots from the system are adopted by user. These models are described in detail respectively as follows.

### Zero-Shot Categorical State Tracker

The zero-shot categorical state tracker is modelled to track the categorical slots. Considering the generalization in zero-shot settings, the proposed zero-shot categorical state tracker is modelled as a binary classifier that infers whether a possible value $V_m^j$ is provided by the user in last two utterances $(U_t^{sys}, U_t^{usr})$ or not, where $t$ denotes the number of turns, $V_m^j$ denotes a possible value from slot $S_m$, $U_t^{sys}$ is the system utterance at turn $t$, and $U_t^{usr}$ is the user utterance at turn $t$. During the inferences process, if all the possible values are not provided in utterances $(U_t^{sys}, U_t^{usr})$, then the slot value of $S_m$ from the last turn is transmitted to the current turn, if at least one possible value is predicted as provided in utterances $(U_t^{sys}, U_t^{usr})$, then the possible value with highest probability on provided is updated to the current turn.

To model this binary classifier for zero-shot categorical state tracker, a pre-trained BERT is applied as shown in Figure 2. This pre-trained BERT encodes the slot $S_m$, the possible value $V_m^j$, the system utterance $U_t^{sys}$ and the user utterance $U_t^{usr}$, and outputs the distribution over two types of labels $\{none, provided\}$, where the class label of "provided" represents that the possible value $V_m^j$ is provided in utterances $(U_t^{sys}, U_t^{usr})$. Concretely, the aforementioned input is tokenized and organized into two sequences $A_{sc}$ and $B_{sc}$ as follows:

$$A_{sc} = S_m \oplus V_m^j, \tag{1}$$

$$B_{sc} = U_t^{sys} \oplus U_t^{usr}, \tag{2}$$

where $\oplus$ denotes the sequence concatenation. The embedded representations of entire input are obtained through pre-trained BERT as follows:

$$\boldsymbol{H}_j = \text{BERT}([\text{CLS}] \oplus A_{sc} \oplus [\text{SEP}] \oplus B_{sc} \oplus [\text{SEP}]), \tag{3}$$

where $\boldsymbol{h}_j^{\text{CLS}} \in \mathbb{R}^d$, the final state corresponding to the [CLS] token, is the aggregated representation of the input token sequence. The $\boldsymbol{h}_j^{\text{CLS}}$ is utilized to perform binary classification, and the distribution over two labels of $\{none, provided\}$ is calculated as follows:

$$\hat{\boldsymbol{y}}_j^{sc} = \text{softmax}(\boldsymbol{W}_{sc}\boldsymbol{h}_j^{\text{CLS}} + \boldsymbol{b}_{sc}) = [p_j^0, p_j^1] \in \mathbb{R}^2, \tag{4}$$

where $\boldsymbol{W}_{sc} \in \mathbb{R}^{2\times d}$ and $\boldsymbol{b}_{sc} \in \mathbb{R}^2$ are learnable weights and bias, respectively. $p_j^1$ is the probability that $V_m^j$ is predicted as the "provided".

The loss of zero-shot categorical state tracker for slot $S_m$ at turn $t$ and the total loss of a given active service are com-

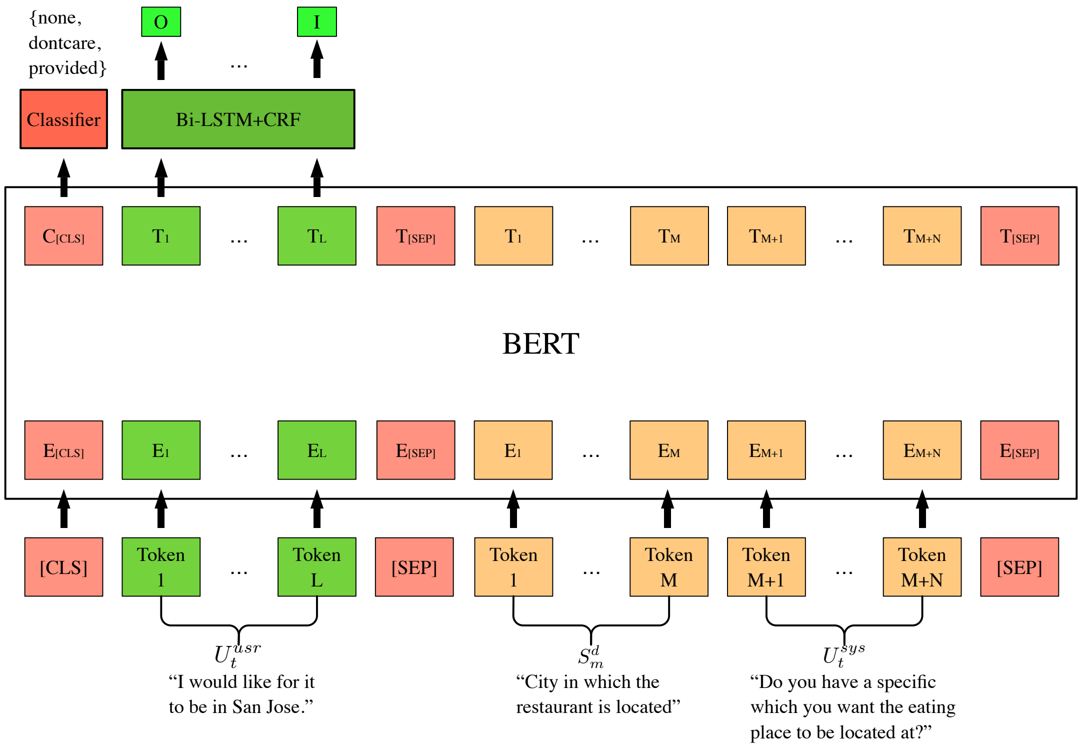


Figure 3: Zero-shot free-form state tracker.

puted as follows:

$$\mathcal{L}_{sc}^{m} = -\sum_{j=1}^{J} y_j^{sc} \log(\hat{y}_j^{sc}), \tag{5}$$

$$\mathcal{L}_{sc} = \sum_{m=1}^{M} \mathcal{L}_{sc}^{m}, \tag{6}$$

where $y_j^{sc}$ is the "one-hot vector" of ground truth label.

## Zero-shot Free-Form State Tracker

The zero-shot free-form state tracker is modelled to track free-form slots. Considering the generalization in zero-shot settings, the proposed free-form categorical state tracker is modelled as the Machine Reading Comprehension (MRC) model like the studies in (Gao et al. 2019; Chao and Lane 2019). Different from above studies, the user utterance $U_t^{usr}$ is used as the passage, and the slot description $S_m^d$ concatenated with the system utterance $U_t^{sys}$ is applied as the question in our proposed model. The proposed model outputs a categorical distribution over three types of labels $\{none, dontcare, provided\}$, meanwhile, the proposed model outputs the IOB (in-out-begin) sequence labels for the user utterance $U_t^{usr}$ to point out slot values. During the inferences process, if the input sequence is predicted as label "none", then the slot value $V_m$ from the last turn is transmitted to the current turn, if the input sequence is predicted as label "dontcare", then the "dontcare" is used as the slot value $V_m$, if the input sequence is predicted as label "provided", then the slot value $V_m$ is pointed out based on the IOB (in-out-begin) sequence labels.

To model this multi-task state tracker, a pre-trained BERT is applied as shown in Figure 3. This pre-trained BERT encodes the user utterance $U_t^{usr}$, the slot description $S_m^d$ and the system utterance $U_t^{sys}$, and outputs the distribution $\hat{\boldsymbol{y}}_m^{st}$ over three types of labels $\{none, dontcare, provided\}$. The proposed model also outputs the IOB sequence labels $Y_m^* = (y_1, \dots, y_L)$, which are obtained by mapping the sequence representations from BERT to the labels through a Bi-LSTM layer and a Conditional Random Fields (CRF) layer. Concretely, the input is tokenized and organized into two sequences $A_{st}$ and $B_{st}$ as follows:

$$A_{st} = U_t^{usr}, \tag{7}$$

$$B_{st} = S_m^d \oplus U_t^{sys}, \tag{8}$$

The embedded representations of entire input are obtained as the same way in formula 3. The aggregated representation $\boldsymbol{h}_m^{\text{CLS}}$ is utilized to perform classification, and the distribution over three labels of $\{none, dontcare, provided\}$ is calculated as follows:

$$\hat{\boldsymbol{y}}_m^{st} = \text{softmax}(\boldsymbol{W}_{st}\boldsymbol{h}_m^{\text{CLS}} + \boldsymbol{b}_{st}) = [p_m^0, p_m^1, p_m^2] \in \mathbb{R}^3, \tag{9}$$

where $\boldsymbol{W}_{st} \in \mathbb{R}^{3\times d}$ and $\boldsymbol{b}_{st} \in \mathbb{R}^3$ are learnable weights and bias, respectively. The sequence representations $\boldsymbol{H}_m$ from BERT is used to predict the sequence labels. This sequence representations $\boldsymbol{H}_m$ is projected to a matrix of scores $\boldsymbol{P}_m \in$

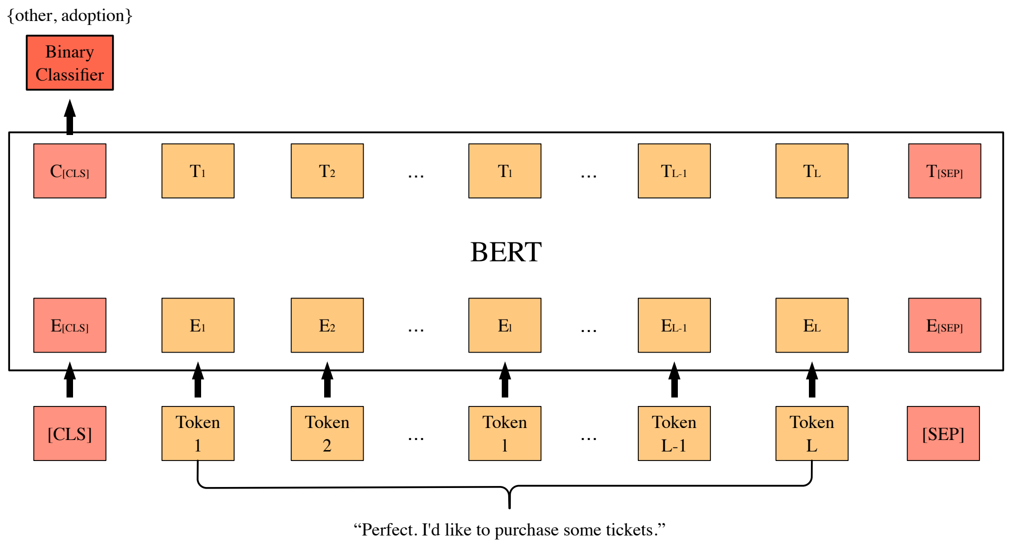


Figure 4: The adoption state tracker.

$\mathbb{R}^{L\times K}$ by the bidirectional LSTM, which is computed as follows:

$$\boldsymbol{P}_m = BiLSTM(\boldsymbol{H}_m), \tag{10}$$

where $L$ is the sequence length of user utterance $U_t^{usr}$, $K$ denotes the number of distinct tags and $P_{l,k}$ corresponds to the score of the $k^{th}$ tag of the $l^{th}$ token in user utterance $U_t^{usr}$. The scores $\boldsymbol{P}_m$ is used to obtained the total score over all the possible tag sequences by the CRF layer, which is computed as follows:

$$S(\hat{Y}_m) = \sum_{l=0}^{L} A_{y_l,y_{l+1}} + \sum_{l=1}^{L} P_{l,y_l}, \hat{Y}_m \in \hat{\boldsymbol{Y}}_m, \tag{11}$$

where $\hat{Y}_m = (\hat{y}_1, \ldots, \hat{y}_L)$ denotes a possible tag sequence, $\hat{\boldsymbol{Y}}_m$ denotes the set of all possible tag sequences, and $\boldsymbol{A}$ denotes a matrix of transition scores such that $A_{i,j}$ presents the score of a transition from tag $i$ to tag $j$. The output sequence $Y_m^*$ is predicted as the sequence with the maximum score, which is computed as follows:

$$Y_m^* = \arg\max_{\hat{Y}_m \in \hat{\boldsymbol{Y}}_m} S(\hat{Y}_m) \tag{12}$$

The output of classification and the output of sequence labels are trained jointly as a multi-task learning. The total loss of these two parts is computed as follows:

$$\mathcal{L}_{st}^m = -\sum_{m=1}^{M} (y_m^{st} \log(\hat{y}_m^{st}) + S(Y_m) - \log(\sum_{\hat{Y}_m \in \hat{\boldsymbol{Y}}_m} S(\hat{Y}_m)))), \tag{13}$$

where $y_m^{st}$ is the "one-hot vector" of ground truth label and $Y_m$ is the ground truth sequence labels.

## Adoption State Tracker

Since the offered slots from system may delay being updated to state based on the user adoption for those slots as shown in Figure 1, an adoption state tracker is modelled to detect in which turn offered slots are adopted by the user. To this end, the proposed model is modelled by a pre-trained model that encodes the user utterance $U_t^{usr}$ and outputs the distribution over two type of labels $\{other, adoption\}$ as show in Figure 4. During the inferences, if the user utterance is predicted as label "adoption", then the offered slots from system are updated to dialogue state, for another case that the offered slots are already informed by the user, the offered slots can not be updated to the slots, if the input is predicted as label "other", then the model does nothing to the offered slots. The embeded representations of user utterance $U_t^{usr}$ are obtained through pre-trained BERT as follows:

$$\boldsymbol{H} = \text{BERT}([\text{CLS}] \oplus token(U_t^{usr}) \oplus [\text{SEP}]), \tag{14}$$

The aggregated representation $\boldsymbol{h}^{\text{CLS}}$ is utilized to perform classification, and the distribution over two labels of $\{other, adoption\}$ is calculated as follows:

$$\hat{\boldsymbol{y}}^{ua} = \text{softmax}(\boldsymbol{W}_{ua}\boldsymbol{h}^{\text{CLS}} + \boldsymbol{b}_{ua}) = [p^0, p^1] \in \mathbb{R}^2, \tag{15}$$

where $\boldsymbol{W}_{ua} \in \mathbb{R}^{2\times d}$ and $\boldsymbol{b}_{ua} \in \mathbb{R}^2$ are learnable weights and bias, respectively, and $p^1$ is the probability that user utterance $U_t^{usr}$ is predicted as the "adoption".

The loss of adoption state tracker for one dialogue is computed as follows:

$$\mathcal{L}_{ua}^m = -\sum_{t=1}^{T} y_t^{ua} \log(\hat{y}_t^{ua}), \tag{16}$$

$$\tag{17}$$

where $y_t^{ua}$ is the "one-hot vector" of golden label. Since these labels is not provided in the dataset, a simple rule is applied to obtain the adoption labels as described in next section.

Table 1: The overall statistics of schema-guided dialogue dataset.

| | Training | Dev | Test |
|---|---|---|---|
| No. of dialogues | 16,142 | 2,482 | 4,201 |
| No. of turns | 329,964 | 48,726 | 84,594 |
| Total domains | 16 | 16 | 18 |
| Total services | 26 | 17 | 21 |

Table 2: Model performance on dev sets and test sets (Avg GA: average goal accuracy, Joint GA: joint goal accuracy).

| | Dev | | Test | |
|---|---|---|---|---|
| | Avg GA | Joint GA | Avg GA | Joint GA |
| Baseline | 0.694 | 0.383 | 0.740 | 0.411 |
| Our model | 0.943 | 0.786 | 0.925 | 0.730 |

## Experiment

### DataSet

Schema-guided dialogue dataset (Rastogi et al. 2019) is the largest public task-oriented dialogue corpus, which contains over 16k multi-domain conversations spanning 16 domains. The overall statistics of the training, dev and test sets are shown in Table 1. Different form prior datasets such as DSTC2 (Williams et al. 2013), WOZ (Wen et al. 2017) and MultiWOZ (Budzianowski et al. 2018), schema-guided dialogue dataset has multiple services with overlapping functionality and is designed to serve as an effective testbed for state tracking in large-scale virtual assistants. In this dataset, the evaluation sets contain unseen services. The slots are categorized into two types: categorical slot (i.e. slot takes one of a finite set of possible values) and free-form slot (i.e. slot can take any string value). The descriptions of these two types slots are listed in the schema. The dialogue state tracking labels are used to train the zero-shot categorical state tracker and the zero-shot free-from state tracker. Since these labels is not provided in the dataset, a simple rule is applied to label user utterances. This rule is that the different between current slot value labels and last slot value labels are compared, if the newly emerging slot value is equal to the offered slot value, then the user utterance is labelled as "adoption". In other case, the user utterance is labelled as "other".

Table 3: Zero-shot performance on dev sets and test sets (Avg GA: average goal accuracy, Joint GA: joint goal accuracy).

| | Dev | | Test | |
|---|---|---|---|---|
| | Avg GA | Joint GA | Avg GA | Joint GA |
| All | 0.943 | 0.786 | 0.925 | 0.730 |
| Unseen | 0.904 | 0.635 | 0.910 | 0.675 |

### Implementation details

We use the BERT-Base (Uncased), which has 12 hidden layers of 768 units and 12 self-attention heads for lower-cased input text, to model proposed state trackers. We use Adam with learning rate of 2e-5 and batch size of 32 to fine-tune the pre-trained models. For the rest of hyperparameters, we use the same settings as proposed in (Devlin et al. 2019). For each state tracker, we fine-tune the model with 3 epochs. For the adoption state tracker, the offered slots can be accessed from the system action, which are provided by the datasets. For multi-domain dialogues, we also align the slots from different services with pre-defined rules. These predefined rules are extracted from the training set automatically through a simple criterion. This extracting criterion is that if the value of some slots in downstream service is equal to one value in upstream service, the corresponding projection from the slot in upstream service to the slot in downstream is appended to the mapping rule set.

Table 4: Ablation study of improvement on incorporating with adoption state tracker on dev sets (Avg GA: average goal accuracy, Joint GA: joint goal accuracy).

| | Avg GA | Joint GA |
|---|---|---|
| Our model | 0.943 | 0.786 |
| -Adaption state tracker | 0.879 | 0.640 |

Table 5: Ablation study of different pointing methods for free-form slots.

| | F1 score | Precision | Recall |
|---|---|---|---|
| LSTM+CRF | 0.976 | 0.989 | 0.981 |
| Start+End | 0.737 | 0.937 | 0.739 |

### Results

We use two evaluation metrics, average goal accuracy and joint goal accuracy to evaluate the performance on schema-guided dialogue state tracking. For average goal accuracy, we compute the accuracy for the slots which have a non-empty assignment in the ground truth dialogue state. For joint goal accuracy, we compute the average accuracy of predicting all slot assignments for a turn correctly. In these two evaluation metrics, we use the fuzzy matching score as the correctness for non-categorical slots.

We make a comparison with the baseline model as described in (Rastogi et al. 2019). The baseline model consists of two modules: a schema embedding module and a state update module. The schema embedding module encodes all slots and slot values for categorical slots present in the schema into an embedded representation. The state update module uses the utterance and the schema embeddings together to obtain state predictions.

To have a fair comparison, we use the same data partitioning as the baseline model in (Rastogi et al. 2019). As shown in Table 2, our proposed model achieves the best performance on both dev sets and test sets, 94.3% on average goal accuracy and 78.6% on joint goal accuracy on dev sets, 92.5% on average goal accuracy and 73.0% on joint goal accuracy on test sets.

Considering the zero-shot performance, we show the results on unseen service on both dev sets and test sets in Table 3. We can observe that the proposed state trackers also achieve competitive performance on average goal accuracy.

Table 6: A good case for the “Services_4” service, which is one service in the dev set and not present in the training set (SYS: system, USR: human user) the dialogue id is “3_00036” in dev set.

| | Utterance | State |
|---|---|---|
| USR: | Hello! I’m looking to make an appointment with a Psychiatrist in Santa Rosa. Can you help? | “”city”: “”Santa Rosa”, ”type”: “”Psychiatrist” |
| SYS: | I’d be happy to help. I have compiled a list of 6 therapists that may meet your needs. May I suggest Beck Jennifer P.? She is a Psychiatrist located in Santa Rosa. | |
| USR: | Can you tell me the address of her office? | “city”: “Santa Rosa”, “”type”: “Psychiatrist” |
| SYS: | It is 1400 North Dutton Avenue #6. | |
| USR: | That’s not the one I was thinking of. Maybe it was a Psychologist. Can you find a Psychologist in Santa Rosa? | “city”: “Santa Rosa”, “type”: “Psychiatrist” |
| SYS: | Yes, Bert Epstein is a Psychologist located in Santa Rosa. | |
| USR: | Are there any other psychologists in Santa Rosa? | “city”: “Santa Rosa”, “type”: “Psychiatrist” |
| SYS: | Blank Gary A is also a licensed Psychologist with an office in Santa Rosa. | |
| USR: | Can you tell me what the address of that office is? | “city”: “Santa Rosa”, “type”: “Psychiatrist” |
| SYS: | Certainly. The address is 2455 Bennett Valley Road # B208. | |
| USR: | Yes, that’s it. | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A” |
| SYS: | Would you like to make an appointment now? | |
| USR: | Yes, please make any appointment for me. | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A” |
| SYS: | Which date and time would work best with your schedule? | |
| USR: | Can you see if there is anything available for 18:00 on the 4th? | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A”, “appointment_date”: “the 4th”, “appointment_time”: “18:00” |
| SYS: | You’d like to make at appointment with Blank Gary A next Monday at 6 pm. Is that correct? | |
| USR: | I’m sorry. I have a meeting that day. Can you see if there are any openings tomorrow at quarter past 10 in the morning? | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A”, “appointment_date”: “tomorrow”, “appointment_time”: “quarter past 10 in the morning” |
| SYS: | Let’s review those details. You’d like the appointment for 10:15 am tomorrow. Have I got that right? | |
| USR: | Yes, that would work for me. | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A”, “appointment_date”: “tomorrow”, “appointment_time”: “10:15 am” |
| SYS: | Your appointment has been confirmed. You will get an email confirmation shortly. | |
| USR: | Can you tell me the phone number of the office? | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A”, “appointment_date”: “tomorrow”, “appointment_time”: “10:15 am” |
| SYS: | Yes, the contact number is 707-526-2525. | |
| USR: | Great! Thank you for all of your help. | “city”: “Santa Rosa”, “type”: “Psychiatrist”, “therapist_name”: “Blank Gary A”, “appointment_date”: “tomorrow”, “appointment_time”: “10:15 am” |
| SYS: | You’re welcome. Have a nice day! | |

## Ablation Study

We conduct an ablation experiment on dev sets for considering the improvement on incorporating with adoption state tracker. We compare the proposed model that discards adoption state tracker. The results of the ablation experiment are shown in Table 4. We observe that the model with adoption state tracker outperforms the model without adoption state tracker. Specifically, the model incorporating with adoption state tracker achieves 6.4% improvement on average goal accuracy and 14.6% improvement on joint goal accuracy. The adoption state tracker improves the performance on both average goal accuracy and joint goal accuracy effectively.

We also conduct an ablation experiment on dev sets for considering the different performance between the approach of labelling IOB sequence for a slot value and the approach of pointing out the start-end token of a slot value. We evaluate this performance through comparing the F1 score on slot filling task. The results of different pointing methods are shown in Table 5. We employ the evaluation scripts provided by organizers to compute the F1 scores. To our surprise, we observe that the pointing method with labelling IOB sequence (i.e. LSTM+CRF) outperforms the method of pointing start and end token positions.

## Good case study

We also give a specific case to visualize the effectiveness of proposed model in zero-shot settings as shown in Table 6. For “Services” domain, the “Services_4” service is only present in the dev set. Although “Services_4” service has the similar schema with the “Services_*” services in training set, they also have some different slots, for instance the “type” slot in “Services_4” is totally different. From the example in Table 6, we observe that our proposed model by the way of the guided schema can tackle the unseen service effectively. In addition, the adoption state tracker can detect in which turn the system offered slots are updated to state precisely.

## Conclusion

In this paper, we propose two zero-shot state trackers, a categorical state tracker for categorical slots and a free-form state tracker for free-form slots, to tackle the zero-shot setting on dialogue state tracking. In addition, we propose an adoption state tracker to detect in which turn the offered slots from the system are adopted by the user to address the problem that such offered slots may delay being updated to state more than two utterances. Through combining above three proposed state trackers as a pipeline, we achieve 73.03% joint goal accuracy (i.e. the 5th place) and 92.49% average goal accuracy (i.e. the 3rd place) on the 4th track in DSTC8.

## Acknowledgments

The authors would like to thank the reviewers for their comments and suggestions. We would like to thank Professor Caixia Yuan for her insightful comments on the paper.

## References

Budzianowski, P.; Wen, T.-H.; Tseng, B.-H.; Casanueva, I.; Ultes, S.; Ramadan, O.; and Gasic, M. 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*, 5016–5026.

Chao, G., and Lane, I. 2019. BERT-DST: scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *CoRR* abs/1907.03040.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. 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)*, 4171–4186.

Gao, S.; Sethi, A.; Agarwal, S.; Chung, T.; and Hakkani-Tur, D. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, 264–273.

Goel, R.; Paul, S.; Chung, T.; Lecomte, J.; Mandal, A.; and Hakkani-Tur, D. 2018. Flexible and scalable state tracking framework for goal-oriented dialogue systems. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS).*

Goel, R.; Paul, S.; and Hakkani-Tür, D. 2019. Hyst: A hybrid approach for flexible and accurate dialogue state tracking. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (INTERSPEECH).*

Henderson, M.; Thomson, B.; and Young, S. 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)*, 292–299. Philadelphia, PA, U.S.A.: Association for Computational Linguistics.

Lee, H.; Lee, J.; and Kim, T.-Y. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL).*

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

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 the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1777–1788.

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.*

Rastogi, A.; Hakkani-Tür, D.; and Heck, L. 2017. Scalable multi-domain dialogue state tracking. In *Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU).*

Ren, L.; Xie, K.; Chen, L.; and Yu, K. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.*

Wen, T.; Vandyke, D.; Mrkšíc, N.; Gašíc, M.; Rojas-Barahona, L.; Su, P.; Ultes, S.; and Young, S. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017-Proceedings of Conference*, volume 1, 438–449.

Williams, J.; Raux, A.; Ramachandran, D.; and Black, A. 2013. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, 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 the 57th Annual Meeting of the Association for Computational Linguistics*, 808–819. Florence, Italy: Association for Computational Linguistics.

Xu, P., and Hu, Q. 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).*

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; and Le, Q. V. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237.*

Zhang, J.-G.; Hashimoto, K.; Wu, C.-S.; Wan, Y.; Yu, P. S.; Socher, R.; and Xiong, C. 2019. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv preprint arXiv:1910.03544.*

Zhong, V.; Xiong, C.; and Socher, R. 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)*, 1458–1467. Melbourne, Australia: Association for Computational Linguistics.