

# IMPROVING LONGER-RANGE DIALOGUE STATE TRACKING

Ye Zhang, Yuan Cao, Mahdis Mahdiah, Jeffrey Zhao, Yonghui Wu

Google Research

{yezhan, yuancao, mahdis, jeffreyyzhao, yonghui}@google.com

## ABSTRACT

Dialogue state tracking (DST) is a pivotal component in task-oriented dialogue systems. While it is relatively easy for a DST model to capture belief states in short conversations, the task of DST becomes more challenging as the length of a dialogue increases due to the injection of more distracting contexts. In this paper, we aim to improve the overall performance of DST with a special focus on handling longer dialogues. We tackle this problem from three perspectives: 1) A model designed to enable hierarchical slot status prediction; 2) Balanced training procedure for generic and task-specific language understanding; 3) Data perturbation which enhances the model’s ability in handling longer conversations. We conduct experiments on the MultiWOZ benchmark, and demonstrate the effectiveness of each component via a set of ablation tests, especially on longer conversations.

motivation

method

## 1 INTRODUCTION

Dialog state tracking (DST) is a key component in modern task-oriented dialogue (ToD) systems. The goal of DST is to keep track of a user’s requirements from the conversation between the user and agent, it is therefore critical for a DST model to be accurate in reasoning about the belief states from dialogue history, especially when the dialogue length increases as the conversation further unfolds.

challenge

While significant progress has been made in recent years on DST (Chao & Lane, 2019; Wu et al., 2019; Zhang et al., 2019; Mehri et al., 2020; Hosseini-Asl et al., 2020; Li et al., 2020), mostly driven by the advancements in NLP techniques including pre-training followed by task-specific fine-tuning (Devlin et al., 2018; Liu et al., 2019) and large-scale modeling (Radford et al., 2019; Brown et al., 2020; Raffel et al., 2020), it remains a challenging problem how to properly handle multi-turn, multi-domain dialogues with long contexts. Although a BERT-style pre-trained encoder is a standard component in many of the state-of-the-art DST models and is capable of making precise belief state predictions when the context is short and clean, it can still be easily misguided by distracting information from remote turns hence fails to reason about the current states.

To further improve the DST model quality, especially for dialogues with many turns and covering multiple domains, we believe that one should not only improve the model’s generic and task-specific language understanding ability, but also properly design the model such that it becomes more selective about the information it needs to predict and guards itself against distractions from prolonged dialogue contexts. Following this line of thought, in this paper we investigate the following ways of improvement: 1) Hierarchical prediction of slot status; 2) Training procedure that balances generic and DST-specific language understanding; 3) Data perturbation which enhances a model’s ability of handling longer conversations. Our experimental results on the MultiWOZ benchmark (Budzianowski et al., 2018; Eric et al., 2019) show that not only do these techniques lift the overall accuracy of state tracking, but they are mostly effective in improving longer conversations while the progress made for short ones slow down. Nevertheless, there is still much room for improvement for long-range conversations, which deserves more focus for future research work.

approach

method

---

## 2 MODEL

### 2.1 BASE MODEL

backbone

Our base model is built upon a pre-trained BERT encoder as backbone, on top of which multiple prediction heads are mounted to predict state values and their statuses. Inputs to the model are the conversation histories concatenated with the most recent utterance. Following a similar strategy as Zhang et al. (2019), we divide states into two types, namely categorical and non-categorical slots. Categorical slots are those having a pre-defined, closed-set of candidate values, such as “cheap” and “expensive” for restaurant price range. Non-categorical slots are those whose values cannot be practically pre-defined and better be extracted directly from utterances, for example the departure time of a train.

definition

method

Upon receiving each new utterance, the model first encodes the history together with the utterance. The same encoder is also used to encode the names of each candidate categorical and non-categorical slots. The combination of the representations of candidate slots and utterances are then used to make predictions for the slot values: for categorical slots their categories are predicted by a softmax, and for non-categorical slots sigmoid functions are used to indicate the start and end positions of value spans in the utterances. For each candidate slot, a softmax is also used to predict the state status, which is one of {active, don’t care, inactive}. The predicted values are only considered for those slots whose status are inferred as active.

### 2.2 HIERARCHICAL SLOT STATUS PREDICTION

motive

One problem with the DST model is that the state prediction accuracy heavily hinges on the correctness of the slot status. Rightly predicted slot values are immediately ignored as long as the slots are assigned wrong statuses. In fact, even with the simplest setup of our baseline models, which reaches 48.1% joint-goal-accuracy (JGA) on the MultiWOZ 2.1 (Eric et al., 2019) benchmark, if we do a cheating experiment and use oracle slot statuses instead of their predictions during inference, the JGA immediately rises up to 85.75%! This reveals the criticality of making correct predictions for slot status.

slot status is important

challenge

Status prediction is difficult as a majority of slots are inactive at any turn in a conversation, which creates a severe class imbalance problem for both training and inference procedures. What is more, it becomes even less accurate as the length of a conversation increases, due to the introduction of more distracting information. To mitigate these issues, we propose a hierarchical slot status prediction procedure. We first predict the domain of the current turn, then only predicts status of slots relevant to the active domain. This enables us to narrow down the range of slots to a small subset most likely to be relevant to the current turn, keeping the model more focused and reducing false triggering of distracting slots. As an example, in the MultiWOZ dataset there are 35 slots and 7 domains. Instead of a flat-prediction for all 35 slot statuses simultaneously, by conditioning the slots on domains we only need to consider 5 candidate slots on average. As will be seen in Section 4.2, hierarchical prediction contributes significantly to model quality improvement.

proposal

### 2.3 OVERALL PROCEDURE

We summarize the overall model prediction procedure in Algorithm 1 with the following annotations:

$\mathbf{u}_t$ : Utterance at turn  $t$ .

$\mathbf{h}^{\{c,n,u\}}$ : Vectors of dimension  $d$ , representations for categorical/non-categorical slots and utterances.

$\mathbf{H}^u$ : Matrix of  $L_t \times d$  where  $L_t$  is the total number of input tokens up to turn  $t$ , sequential representations of conversation at time  $t$ .

$\mathbf{H} \oplus \mathbf{h}$ : Adding  $\mathbf{h}$  to each row of  $\mathbf{H}$ .

$\mathbf{s}^{c,n}$ : Name strings for categorical and non-categorical slots.

$\mathcal{C}, \mathcal{N}$ : Categorical and non-categorical slots sets.

---

**Algorithm 1** Model prediction procedure

---

```
1: procedure DST
2:   ▷ Slot encoding
3:    $\mathbf{h}_i^c = \text{Encode}(\mathbf{s}_i^c), i \in \mathcal{C}$ 
4:    $\mathbf{h}_i^n = \text{Encode}(\mathbf{s}_i^n), i \in \mathcal{N}$ 
5:   ▷ Context encoding
6:    $\mathbf{h}_t^u, \mathbf{H}_t^u = \text{Encode}(\mathbf{u}_1, \dots, \mathbf{u}_t)$ 
7:   ▷ Domain prediction
8:    $\mathbf{a}_i^d = \text{Atten}_d(\mathbf{q}, \mathbf{H}_t^u), i \in \mathcal{D}$ 
9:    $\mathbf{d}_t = [\text{sigmoid}(\mathbf{a}_1^d), \dots, \text{sigmoid}(\mathbf{a}_{|\mathcal{D}|}^d)]$ 
10:  ▷ Slot status prediction
11:  for  $i \in \mathcal{D}$  do
12:    if  $\mathbf{d}_t[i] == 1$  then
13:      for  $s \in \mathcal{S}_i$  do
14:         $\mathbf{a}_s^g = \text{Atten}_g(\mathbf{h}_s^{\{c,n\}}, \mathbf{H}_t^u)$ 
15:         $g_s = \text{softmax}_g(\mathbf{h}_s^{\{c,n\}} + \mathbf{a}_s^g)$ 
16:      end for
17:    else
18:       $g_s = \text{"inactive"}, s \in \mathcal{S}_i$ 
19:    end if
20:  end for
21:  ▷ Slot value prediction
22:  for  $i \in \mathcal{C}$  and  $g_i == \text{"active"}$  do
23:     $\mathbf{a}_i^c = \text{Atten}_c(\mathbf{h}_i^c, \mathbf{H}_t^u)$ 
24:     $v_i = \text{softmax}_c(\mathbf{h}_i^u + \mathbf{a}_i^c)$ 
25:  end for
26:  for  $i \in \mathcal{N}$  and  $g_i == \text{"active"}$  do
27:    ▷ Start/end predictions
28:     $p_i^s = \arg \max_{l=1, \dots, L_t} P_s(\mathbf{H}_t^u \oplus \mathbf{h}_i^n)$ 
29:     $p_i^e = \arg \max_{l=1, \dots, L_t} P_e(\mathbf{H}_t^u \oplus \mathbf{h}_i^n)$ 
30:  end for
31: end procedure
```

---

$\mathcal{D}$ : Domain set.

$\mathcal{S}_i$ : Set of slots belonging to the  $i^{th}$  domain.

$\mathbf{q}$ : A learnable query vector for domain attention.

Encode: Encoding function using pre-trained encoders.

$\text{Atten}_{\{d,g,c\}}$ : Attention functions for domain, status and categorical slots.

$\text{softmax}_{\{d,g,c\}}$ : Softmax for domain, status and categorical slots.

$P_{\{s,e\}}$ : Position prediction model producing logits at each input token position.

### 3 TRAINING PROCEDURE

#### 3.1 PRE-TRAINING AND FINE-TUNING

While pre-training followed by fine-tuning has now almost become a standard training paradigm for NLP tasks, to maximize the efficacy of this procedure it also needs to be properly designed to accommodate the problem structure of the task. Similar to the findings of Mehri et al. (2020), we found two setups that contribute significantly to quality improvement, especially for long conversations as we will show in the experimental section:

*proposal*

**Domain closeness:** The datasets used for pre-training are preferred to be as close to the dialogue domain as possible. In our work, we pre-train the encoder on a large-scale dataset used to train the Meena open-domain chat bot (Adiwardana et al., 2020), whose examples are crawled from social media and internet forums. Although the style of conversations included in the dataset is still very different from task-oriented dialogues, we found it works better than pre-training on the original dataset used by BERT, which is collected from sources of general domains like Wikipedia and Bookcorpus which has little overlap with conversation domains. result

**Continued MLM:** The pre-training procedure endows the model with generic language understanding ability, and following common practice one can directly start fine-tuning the pre-trained model on DST task datasets. A model trained in this way, nevertheless, has missed an opportunity of acquiring task-specific language understanding ability before it learns to make state predictions. Therefore in parallel to tuning for the DST task, we continue the application of the masked language model (MLM) loss as an auxiliary objective during the fine-tuning stage, but on the conversations from both MultiWOZ and ToDBERT (Wu et al., 2020). ToDBERT is a collection of public datasets for task-oriented dialogue modeling across multiple domains. ToDBERT matches the MultiWOZ conversation style much better than the Meena dataset used for pre-training, hence it effectively assists the model to gain better understanding in language relevant to the downstream task. The overall size of ToDBERT, however, is tiny compared with the Meena dataset, we therefore only engage this dataset for MLM during fine-tuning. The final loss during the fine-tuning stage is hence the sum of the DST and continued MLM loss. loss

### 3.2 DATA PERTURBATION

To further improve the model’s ability of handling long-range conversations, we intentionally perturb the dataset during the fine-tuning stage by extending the length of conversations with randomly inserted utterances. Within each fine-tuning batch, we randomly choose a subset of examples to be perturbed with probability  $p$ . For each of the chosen examples, we sample  $N$  utterances from ToDBERT or MultiWOZ then insert them at random positions of the original conversation as distracting turns. This improves the model’s robustness in handling longer conversations by forcing them to extract relevant information from longer, noisier inputs. Alternatively, we may also choose to insert synthetic utterances composed of random words sampled from the vocabulary as perturbation, nevertheless we found this not to be working as effectively as inserting dialogue turns, most likely because random words removes turn fluency, confusing instead of helping the model to improve robustness in dialogue understanding. approach  
unhelpful method

## 4 EXPERIMENTS

### 4.1 SETUP

Our experiments are conducted on the MultiWOZ 2.1 dataset. We use the commonly used joint-goal-accuracy (JGA) as metric for DST, defined as the ratio between turns whose all states are correctly predicted and total number of turns. We implemented our approaches using the Lingvo (Shen et al., 2019)<sup>1</sup> framework. metric

**Pre-training:** For the baseline model, we adopt the standard BERT-Base encoder setup described in (Devlin et al., 2018; Zhang et al., 2019). The maximum sequence length is set to be 512 tokens for the training examples. For pre-training with the Meena dataset, while we mostly followed the procedure described in (Adiwardana et al., 2020), we extended the training sample length to include up to 25 utterances instead of the original 7 utterances. We found using longer examples for pre-training improves quality a little. baseline  
result

**Fine-tuning:** For fine-tuning with continued MLM as auxiliary loss, we compare two setups: 1) Only apply MLM to MultiWOZ; 2) Apply MLM to both MultiWOZ and ToDBERT datasets. In both setups, the DST model loss is simply added to the MLM loss to form final loss. Note that we excluded MultiWOZ from the ToDBERT dataset during the fine-tuning. experiment

<sup>1</sup><https://github.com/tensorflow/lingvo>

Model	All	Short 30%	Long 30%
TRADE	46.00	-	-
DS-DST	52.24	-	-
SST	55.23	-	-
TripPy	55.29	-	-
SimpleToD	56.45	-	-
Base	48.12	70.16	20.62
+ hier status	51.81	72.74	25.33
+ continued MLM	53.94	73.92	26.50
+ ToDBERT MLM	55.65	75.42	29.50
+ Meena pre-training	56.64	75.01	30.92
+ insertion	57.12	75.89	31.67

Table 1: JGA of our methods on the Multiwoz 2.1 benchmark. “Short 30%” and “long 30%” refer to the shortest and longest 30% of dialogues (measured by the number of turns) in the test set. See text for more explanation.

*experiment*

**Data perturbation:** We compare multiple setups for the perturbation approach introduced in Section 3.2: 1) For source of insertion contents, we consider randomly sampled utterances from ToDBERT, MultiWOZ, or synthetic utterances composed of random words sampled from the vocabulary; 2) Number of insertion utterances:  $N = \{2, 3, 4\}$ ; 3) Probability of random insertion:  $p = \{0.2, 0.4, 0.6\}$ ; 4) For positions where the sampled utterances are inserted, we consider random boundaries between turns in the original conversation, after only user turns, or agent turns.

**Baselines:** We compare our results with the following baseline models:

TRADE: A transferable dialogue state generator that generates dialogue states from utterances using a copy mechanism (Wu et al., 2019).

DS-DST: A dual strategy model that simultaneously handles non-categorical and categorical slots (Zhang et al., 2019).

SST: A graph attention network that predicts states from utterances and schema graphs containing slot relations in edges (Chen et al., 2020).

Trippy: A triple copy strategy where a slot is filled by one of the three copy mechanisms (Heck et al., 2020).

SimpleToD: Training a simple language model that casts DST as a sequence prediction problem (Hosseini-Asl et al., 2020).

## 4.2 RESULTS

Our main results are presented in Table 1. The rows of the table correspond to the following setups:

Base: The base model introduced in Section 2.1

+hier status: Enabling hierarchical slot status prediction model in Section 2.2

+continued MLM: Apply MLM auxiliary loss on MultiWOZ examples during fine-tuning.

+ToDBERT MLM: Further add ToDBERT examples for MLM auxiliary loss during fine-tuning.

+Meena pre-training: Instead of pre-training the encoder on the BERT corpus, use the Meena dataset instead. All other setups remain unchanged.

+insertion: Enable data perturbation with insertion introduced in Section 3.2

*evaluation*

In order to analyze whether these techniques help more on longer or shorter conversations, we in addition report a breakdown of JGA on the shortest and longest 30% dialogues contained in the MultiWOZ test set, measured by the number of utterances. Among the 7372 turns in the test set, the shortest 30% consists of dialogues having less or equal to 5 utterances, and the longest 30% are those having more or equal to 11 utterances.

### 4.2.1 ANALYSIS

**Length vs. improvement:** The main observation from Table 1 is that the overall DST accuracy consistently improves as each of the techniques was introduced, and the gain is more pronounced on longer conversations. To contrast the differences these techniques made on short and long conversations, we plot the relative JGA gain after adopting each technique in Figure 1. It can be seen that

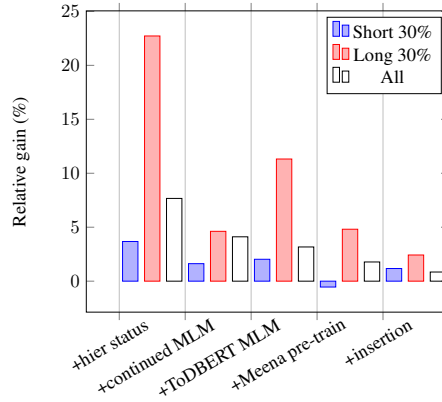


Figure 1: Relative JGA gain on the shortest 30%, longest 30% and all conversations.

each of these techniques, without exception, made more significant improvements on longer conversations than shorter ones, and the biggest relative gain was contributed by enabling hierarchical status prediction.

result

On the other hand, it can also be seen from Table 1 that while JGA for short conversations can reach more than 75%, longer ones barely surpassed 31%. This indicates that further improving long conversations cases are key to the overall DST improvement on benchmarks like MultiWOZ.

**Accuracy of individual components:** To gain deeper understanding of how individual components of the model, namely categorical slot, non-categorical slot and status predictions, contribute to the overall state tracking behavior, we breakdown the accuracy into individual components for both short and long conversations and report them in Table 2. From this table we make the following observations:

1. Unsurprisingly, accuracy is higher for short conversations than longer ones across the board.
2. From the simplest base model to the best model at the bottom of table, the accuracy of categorical slots remains almost constant, for both short (98.74% → 98.95%) and long conversations (98.13% → 98.95%). The improvement on non-categorical slot accuracy, one the other hand, is much more significant, especially for long conversations (95.9% → 97% for short, 89.3% → 92.13% for long). This indicates that categorical slot prediction is more accurate than non-categorical ones, and there is much room for improvement for the latter. It also shows that our proposed techniques mainly improves non-categorical predictions, as conversation becomes longer.
3. The observation on slot status prediction is similar to that of non-categorical slots. Note however that since the absolute values of status accuracy are very high<sup>2</sup>, further improving status accuracy is very challenging, although the benefit will be huge (as we mentioned in Section 2.2). For both short and long conversations, the biggest status accuracy improvement is enabled by hierarchical prediction, and the improvement is larger on long conversations.

Improvement

pros

Improvement

**Data perturbation comparison:** We compare the efficacy of different configurations of the perturbation method in Table 3. Among the variation of perturbation source, probability, number of insertion utterances and positions, the best setup we found is randomly inserting 2 utterances sampled from ToDBERT, with probability 20%. This is a conservative setup that minimally disturbs the original conversation among our variations, which indicates that while perturbation can be helpful in making models more robust in long conversations, too much perturbation confuses the model and can be counterproductive. We also note that although our end-task is DST for MultiWOZ,

result

analysis

<sup>2</sup>Note that the majority of slots are inactive during a conversation, hence even if a model makes trivial, constant status predictions of “inactive”, the accuracy can already reach around 90%.



Model	Cat	Non-cat	Status
Base	98.74	95.9	98.91
	98.13	89.3	95.88
+ hier gate	98.68	95.65	99.02
	98.14	89.14	96.5
+ continued MLM	98.6	95.93	99.06
	98.5	90	96.76
+ ToDBERT MLM	98.71	96.39	99.12
	98.45	90.73	96.9
+ Meena pre-training	98.9	96.47	99.12
	98.77	91.8	96.97
+ insertion	98.95	97	99.15
	98.95	92.13	97.06

Table 2: Breakdown of individual model component (categorical and non-categorical slots, status) prediction accuracy. Each row is divided into two sub-rows, corresponding to the shortest and longest 30% conversations respectively.

Position	JGA	Probability	JGA
Random	57.12	20%	57.12
Agent turn only	56.43	40%	56.73
User turn only	56.65	60%	56.66

(a) Insertion position		(b) Insertion probability	
Source	JGA	Number	JGA
ToDBERT	57.12	2	57.12
Multiwoz	56.41	3	57.02
Random	56.43	4	56.56

(c) Insertion source		(d) Insertion number	
----------------------	--	----------------------	--

Table 3: Different setups of data perturbation.

sampling utterances from MultiWOZ itself didn’t work as well as sampling from ToDBERT. We believe this is because samples from MultiWOZ resemble too much the training example, hence are not as effective in perturbing the model as samples from ToDBERT, a dataset matching the MultiWOZ domain fairly well but not all that similar.

**Examples of improvement:** To inspect the actual effect our proposed techniques have on the prediction of states, in Appendix G we provide several cases for long conversations where errors made by the Meena pre-training model are corrected by the best insertion model.

## 5 RELATED WORK

Rapid progress has been made on DST over the past a couple of years. A few key techniques that drove the progress, among others, include state value copying instead of picking from a pre-defined vocabulary (Wu et al., 2019), application of pre-trained BERT encoder (Chao & Lane, 2019), separation between categorical and non-categorical slots (Zhang et al., 2019), carryover of previously predicted states to avoid repeated prediction via a copy mechanism (Heck et al., 2020), application of large-scale language model for generative stat tracking (Hosseini-Asl et al., 2020), DST-specific pre-training and fine-tuning procedure (Mehri et al., 2020), and various data augmentation approaches (Li et al., 2020).

Perhaps mostly related to our work are (Mehri et al., 2020) and (Li et al., 2020), which also adopted domain-specific pre-training and fine-tuning procedures, as well as data augmentation approaches. Both of these works are based on the Trippy model (Heck et al., 2020) which introduces recurrence by copying previously predicted values, whereas our models are based on a non-recurrent architecture which repeatedly makes all state predictions. What is more, our analysis mainly focuses on the contrast between the effectiveness of various techniques on short and long conversations, which

---

reveals the importance and larger room for improving longer ones. Our work not only corroborates the existing finding, but also provides additional insights and opportunities for further improvement.

## 6 CONCLUSION AND FUTURE WORK

Improving DST accuracy on longer conversations is a key driving force of the overall quality improvement on benchmarks like MultiWOZ. Our study demonstrated that adopting hierarchical slot status prediction, pre-training on datasets close to the conversation domain, continuation of the masked language model objective during fine-tuning, and data perturbation via utterance insertion all improve DST quality significantly, especially on longer conversations. The findings underscores the importance of proper model design to keep it focused on irrelevant information as a conversation extends, as well as training procedures which enhance the model’s ability of understanding longer utterances.

We note however that while we believe our findings are general for DST problems, it is based on a model which repeatedly makes predictions for all states upon receiving every new utterance, which is not a very ideal setup for keeping track of states as the DST task implies a recurrent structure. As next step, we will extend our study to more efficient architectures by introducing such inductive bias for which we believe our findings in the paper will still hold.

## REFERENCES

- Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.
- Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 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 (EMNLP)*, 2018.
- Guan-Lin Chao and Ian Lane. BERT-DST: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In *INTERSPEECH*, 2019.
- Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7521–7528, 2020.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*, 2019.
- Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. Trippy: A triple copy strategy for value independent neural dialog state tracking. *arXiv preprint arXiv:2005.02877*, 2020.
- Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*, 2020.
- Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. Coco: Controllable counterfactuals for evaluating dialogue state trackers. *arXiv preprint arXiv:2010.12850*, 2020.



- 
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*, 2020.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL <http://jmlr.org/papers/v21/20-074.html>.
- Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, Mia X Chen, Ye Jia, Anjuli Kannan, Tara Sainath, Yuan Cao, Chung-Cheng Chiu, et al. Lingvo: a modular and scalable framework for sequence-to-sequence modeling. *arXiv preprint arXiv:1902.08295*, 2019.
- Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. *arXiv preprint arXiv:1905.08743*, 2019.
- Chien-Sheng Wu, Steven Hoi, Richard Socher, and Caiming Xiong. Tod-bert: Pre-trained natural language understanding for task-oriented dialogues. *arXiv preprint arXiv:2004.06871*, 2020.
- Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv preprint arXiv:1910.03544*, 2019.

## A APPENDIX

We show some dialogue examples on which the model before insertion makes the wrong prediction but the model after insertion corrects it. We also provide the slot prediction that the model has corrected.

---

<p>User: I am traveling to cambridge and am interested in trying local restaurants and finding a hotel for my stay.</p> <p>Agent: There are many restaurants. can you please elaborate on what you would like?</p> <p>User: I need to find a hotel with a 3 star rating that includes free wifi.</p> <p>Agent: I have 5 options for you, located all over town. do you have a certain area or price range in mind?</p> <p>User: I want Chinese food in cheap price range in <b>west</b> side of town. a 3 star hotel that is expensive and includes wifi. Also the hotel address, area, and postcode please.</p> <p><b>Before insertion:</b> restaurant-area = ''</p> <p><b>After insertion:</b> restaurant-area = 'west'</p>
<p>User: I would a train. I'm going from birmingham new street and it needs to arrive by 11:00.</p> <p>Agent: What day would you like to travel?</p> <p>User: I am leaving on Saturday, and the train should go to Cambridge.</p> <p>Agent: tr8259 will arrive in cambridge at 10:23. Would you like me to book a ticket for you on that train?</p> <p>User: Not yet. First i would like to get price and departure time.</p> <p>Agent: It will depart at 07:40 and it will cost 60:08 pounds. do you need help with anything else?</p> <p>User: And I need a place to stay.</p> <p>Agent: Okay there are many options. Do you have a price preference or area?</p> <p>User: I would like something in the east. I also prefer a guest house.</p> <p>Agent: There is <b>a and b guest</b> , it is in the east part of town User: That sounds good, I have 7 people in total.</p> <p><b>Before insertion:</b> hotel-name = ''</p> <p><b>After insertion:</b> hotel-name = 'a and b guest'</p>
<p>User: i need a train to cambridge, departing from the peterborough station.</p> <p>Agent: There are a total of 266 trains traveling from peterborough to cambridge. what day would you like to leave?</p> <p>User: I would like to travel on saturday and arrive by 15:15.</p> <p>Agent: i have a train scheduled to depart peterborough at <b>14:19</b> and will arrive in cambridge by 14:38. would you me like to book this for you?</p> <p>User: Yes, for 2 people, please.</p> <p><b>Before Insertion:</b> train leave at = '15:15'</p> <p><b>After Insertion:</b> train leave at = '14:19'</p>

---

Table 4: Dialogues where the model with insertion has corrected the wrong prediction.