Multi-Task Pre-Training for Plug-and-Play Task-Oriented Dialogue System

Anonymous ACL submission

Abstract

Pre-trained language models have been recently shown to benefit task-oriented dialogue (TOD) systems. Despite their success, existing methods often formulate this task as a cascaded generation problem which can lead to error accumulation across different sub-tasks and greater data annotation overhead. In this study, we present PPTOD, a unified plug-and-play model for task-oriented dialogue. In addition, we introduce a new dialogue multi-task pre-training strategy that allows the model to learn the primary TOD task completion skills from heterogeneous dialog corpora. We extensively test our model on three benchmark TOD tasks, including end-to-end dialogue modelling, dialogue state tracking, and intent classification. Experimental results show that PPTOD achieves new state of the art on all evaluated tasks in both high-resource and low-resource scenarios. Furthermore, comparisons against previous SOTA methods show that the responses generated by PPTOD are more factually correct and semantically coherent as judged by human annotators.

1 Introduction

Task-oriented dialogue is often decomposed into three sub-tasks: (1) dialogue state tracking (DST) for tracking user’s belief state; (2) dialogue policy learning (POL) for deciding which system action to take; (3) natural language generation (NLG) for generating dialogue response (Young et al., 2013).

Traditional approaches (Smith and Hipp, 1995; Young et al., 2013) adopt a modularized pipeline that addresses different sub-tasks with distinct dedicated modules. In contrast, recent systems (Wen et al., 2017; Eric et al., 2017; Lei et al., 2018; Shu et al., 2019) integrate all functionalities required to hold a dialogue into neural network models. With the advances in pre-trained language models (PLMs) (Radford et al., 2019; Devlin et al., 2019; Raffel et al., 2020), different systems based on PLMs have been proposed (Hosseini-Asl et al., 2020; Lin et al., 2020; Peng et al., 2021; Liu et al., 2021). Despite their differences, most existing methods formulate task-oriented dialogue as a cascaded generation problem, that is, the model can only solve latter sub-tasks by conditioning on the outputs of previous ones. For instance, to generate the response (NLG), the model must rely on the outputs of previous sub-tasks (i.e., DST and POL).

While impressive results are reported (Hosseini-Asl et al., 2020; Peng et al., 2021), we identify three major limitations in the cascaded formulation of their system design. (1) Firstly, as the model solves all sub-tasks in a sequential order, the errors accumulated from previous steps are propagated to latter steps (Li et al., 2017; Liu and Lane, 2018). (2) Secondly, the training data must be annotated for all sub-tasks. Such annotation requirement significantly increases the data curation overhead. More importantly, it precludes the model from using the large amount of existing data that is partially annotated (e.g., data only annotated with DST or NLG). (3) Thirdly, the results of different sub-tasks must be generated in a cascaded order which inevitably increases the system inference latency.

In this study, we propose a novel Plug-and-Play Task-Oriented Dialogue (PPTOD) system. Figure 1 depicts an illustration of our approach. As seen, we integrate different dialogue modules (e.g. DST, POL, and NLG) into a unified model. Motivated by the concept of in-context learning (Brown et al., 2020), to steer the model to solve different TOD sub-task, we plug a task-specific natural language instruction, termed as prompt, into the dialogue context as the model input. This way, the generations of different sub-tasks are decoupled, leading to a greater flexibility of the model that brings us at least two advantages: (1) As different sub-tasks are solved separately, the model can learn from data that is partially annotated for different sub-tasks.
(e.g., DST and NLG). (2) The outputs of different sub-tasks are generated in parallel which alleviates the problem of error accumulation and reduces the system inference latency.

Inspired by recent success of dialogue language model pre-training (Zhang et al., 2020c; Wu et al., 2020; Peng et al., 2021), we propose a dialogue multi-task pre-training strategy that equips our model with the primary TOD task completion skills. Specifically, initialized with T5 (Raffel et al., 2020), we pre-train our model on a heterogeneous set of dialogue corpora that consist of partially-annotated data. To build the pre-training corpora, we collect and combine eleven human-written multi-turn dialogue corpora. The collected datasets are partially annotated for some of the TOD-related tasks, including natural language understanding (NLU), dialogue state tracking (DST), dialogue policy learning (POL), and natural language generation (NLG). In total, the pre-training corpora contain over 2.3M utterances across over 80 domains (see more details in Table 1). When applying the pre-trained PPTOD to a new task, we fine-tune it using the same learning objective as in the pre-training stage.

We evaluate PPTOD on a wide range of benchmark TOD tasks, including end-to-end dialogue modelling, dialogue state tracking, and intent classification. Comparisons against previous state-of-the-art approaches show that PPTOD achieves better performance in both full-training and low-resource settings as judged by automatic and human evaluations. In summary, our contributions are:

- A novel model, PPTOD, that effectively leverages pre-trained language models for task-oriented dialogue tasks.
- A new dialogue multi-task pre-training strategy that augments the model’s ability with heterogeneous dialogue corpora.
- Extensive evaluations on three benchmark TOD tasks reporting state-of-the-art results in both full-training and low-resource settings.
- In-depth analysis that further reveals the merits of our model design and the proposed multi-task pre-training strategy.

2 Related Work

**Task-Oriented Dialogue.** Task-oriented dialogue aims at accomplishing user’s goal. Traditional systems (Williams and Young, 2007; Young et al., 2013) adopt a pipelined approach that requires dialogue state tracking for understanding user’s goal, dialogue policy learning for deciding which system action to take, and natural language generation for generating dialogue responses.

Recently, to simplify the modelling effort, researchers have shifted their attention to building neural network models that address the TOD sub-tasks (Wen et al., 2017; Eric et al., 2017; Lei et al., 2018; Liang et al., 2020). With the advances in pre-trained language models (PLMs), Budzianowski and Vulić (2019) first applied the GPT-2 model for the NLG task. Lin et al. (2020) and Yang et al. (2021) moved one step forward and utilized pre-trained language models to solve all TOD sub-tasks conditioned on the history of oracle belief states. Based on the GPT-2 model, Hosseini-Asl et al.
(2020) proposed a cascaded model, SimpleTOD, that addresses all TOD sub-tasks without using the oracle information. To improve the system performance, Peng et al. (2021) and Liu et al. (2021) applied dialogue pre-training over external dialogue corpora. However, both methods require the pre-training data to be fully annotated for all TOD sub-tasks (i.e., DST, POL, and NLG) which greatly limits the amount of data they can use. Additionally, Liu et al. (2021) achieved better results with noisy channel model that requires two additional language models for outputs re-scoring. Unlike their approach, we address the task of task-oriented dialogue with a single unified model.

Language Model Pre-training. The research community has witnessed remarkable progress of pre-training methods in a wide range of NLP tasks, including language understanding (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019) and text generation (Radford et al., 2019; Lewis et al., 2020; Rafail et al., 2020).

In the dialogue domain, many models are pre-trained on open-domain conversational data like Reddit. Based on GPT-2, Transfertransfo (Wolf et al., 2019) achieves good results on ConvAI-2 competition. As another extension of GPT-2, DialoGPT (Zhang et al., 2020c) performs well in generating open-domain dialogue response. ConverRT (Henderson et al., 2020) is a language model with dual-encoder built for the task of response selection. PLATO (Bao et al., 2020) pre-trains a model with discrete latent variable structure for the response generation task. Wu et al. (2020) adapts BERT with TOD pre-training and achieves strong performances on four dialogue understanding tasks.

Pre-training on Supplementary Data. Recent work (Phang et al., 2018; Aghajanyan et al., 2021) found that supplementary training on the tasks with intermediate-labelled data improves the performance of the fine-tuned models on GLUE natural language understanding benchmark (Wang et al., 2018). Our work studies a similar supplementary training setup with intermediate-labelled data for task-oriented dialogue systems. Unlike previous work, we use a single multi-task model for all relevant sub-tasks in task-oriented dialogue systems.

### 3 Methodology

In this section, we first discuss the datasets and learning objective used in the proposed dialogue multi-task pre-training. Then we introduce how to apply the pre-trained PPTOD for a new task.

#### 3.1 Pre-training Datasets

To construct the pre-training corpus, we collect eleven human-written multi-turn task-oriented dialogue corpora, including MetaLWOZ (Lee et al., 2019), SNIPS (Coucke et al., 2018), CLINC (Larson et al., 2019), ATIS (Amin, 2019), KVRET (Eric et al., 2017), WOZ (Mrkšić et al., 2017), MSR-E2E (Li et al., 2018), Frames (El Asri et al., 2017), TaskMaster (Byrne et al., 2019), and Schema-Guided (Rastogi et al., 2020). In total, there are over 2.3M utterances across 80 domains. In Table 1, we provide the details of data annotations and utterance/domain statistics of all datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Data Annotation</th>
<th>Utter.</th>
<th>Dom.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NLU</td>
<td>DST</td>
<td>POL</td>
</tr>
<tr>
<td>MetaLWOZ</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>SNIPS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CLINC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ATIS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>KVRET</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WOZ</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CamRest676</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MSR-E2E</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Frames</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TaskMaster</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Schema-Guided</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 1: The summary of data annotations and number of utterances (Utter.) as well as domains (Dom.) for all pre-training corpora. All datasets are partially annotated for some of the TOD-related tasks, including natural language understanding (NLU), dialogue state tracking (DST), dialogue policy learning (POL), and natural language generation (NLG).

#### 3.2 Dialogue Multi-Task Pre-training

Motivated by previous work (McCann et al., 2018; Keskar et al., 2019; Rafail et al., 2020) that unify multiple NLP tasks into a common format, we cast all TOD-related tasks that we consider into the same plug-and-play text generation problem. To specify the target task, we plug a task-specific prompt into the dialogue context as the model input. Figure 1 depicts an illustration of our approach.

In the multi-task pre-training stage, each training sample is represented as:

\[ d = (z_t, x, y), \quad (1) \]

where \( t \) denotes the TOD task that the sample \( d \) belongs to, and \( t \in \{ \text{NLU, DST, POL, NLG} \} \). \( z_t \) is...
the task-specific prompt of the form “translate dialogue to A:”, with A corresponding to “user intent”, “belief state”, “dialogue act”, and “system response” for the tasks of NLU, DST, POL, and NLG, respectively. \(x\) denotes the input dialogue context which is a concatenation of all previous utterances in the dialogue - both system’s and user’s. And \(y\) denotes the target output text.

As an example presented in Figure 1, to perform the user intent classification task (i.e., NLU), the model is fed with the sequence “translate dialogue to user intent: \{user\} Tell me the weather forecast for Lecanto, Georgia.” and is trained to generate the user intent label text “\{get_weather\}”.

**Learning.** The model is trained with a maximum likelihood objective. Given the training sample \(d = (z_t, x, y)\), the objective \(\mathcal{L}_\Theta\) is defined as

\[
\mathcal{L}_\Theta = -\sum_{i=1}^{\left|y\right|} \log P_{\Theta}(y_i|y_{<i}; z_t, x), \quad (2)
\]

where \(\Theta\) is the model parameters.

In the multi-task pre-training stage, the model is trained to perform all TOD-related tasks with data annotated for different tasks. To optimize the model parameters \(\Theta\), we use mini-batch based optimization approach as shown in Algorithm 1.

### 3.3 Fine-Tuning to a New Task

When applying the pre-trained PPTOD to a new downstream task with task-specific labelled data, we use the same learning objective Eq. (2) as in the dialogue multi-task pre-training stage.

### 3.4 Implementation Details

In this work, we report results of PPTOD with three model sizes: PPTOD\(_{\text{small}}\), PPTOD\(_{\text{base}}\), and PPTOD\(_{\text{large}}\). These three models are initialized with T5-small, T5-base, and T5-large models (Raffel et al., 2020) that contain \(\sim60M\), \(\sim220M\), and \(\sim770M\) parameters, respectively. We pre-train the model with different configurations on our collected pre-training corpora for 10 epochs. The training samples are truncated to ensure a maximal length of 1024. The models are trained using Adam optimizer (Kingma and Ba, 2015) with a learning rate of 5e-5 and a batch size of 128. Our implementation is based on the Huggingface Library (Wolf et al., 2019a).

### 4 Experiments

We test PPTOD on three benchmark TOD tasks: (1) end-to-end dialogue modelling; (2) dialogue state tracking; and (3) user intent classification.

#### 4.1 End-to-End Dialogue Modelling

End-to-end dialogue modelling aims at evaluating the model in the most realistic, fully end-to-end setting, where the generated dialogue states are used for the database search and response generation (Zhang et al., 2020b; Hosseini-Asl et al., 2020).

##### 4.1.1 Dataset and Evaluation Metric

We conduct experiments on the benchmark MultiWOZ 2.0 (Budzianowski et al., 2018) and 2.1 (Eric et al., 2020) datasets. In MultiWOZ, the generation of response is not only related to the dialogue context, but also grounded on the database (DB) state. The DB state is automatically retrieved from a pre-defined database using the generated dialogue state (DST). Following previous studies, during inference, PPTOD first predicts the DST result to retrieve the DB state. Then, based on the retrieved DB state and the dialogue context, the results of POL and NLG are generated in parallel. In Section 5, we further compare the performance of our model with or without using the DB state as input.

For evaluation, we follow the original MultiWOZ guidance for all individual metrics: **Inform**, **Success**, and **BLEU** (Papineni et al., 2002). An overall measurement, i.e., combined score (Mehri et al., 2019), is also reported which is defined as **Combined** = (Inform + Success) × 0.5 + BLEU.

##### 4.1.2 Baselines

We compare PPTOD with several strong baselines, including Sequicity (Lei et al., 2018), MD-Sequicity (Zhang et al., 2020b), DAMD (Zhang et al., 2019a), and our dialogue pre-training corpora.

\[\text{Note that, there is no overlap between the MultiWOZ dataset and our dialogue pre-training corpora.}\]
To investigate the generalization ability of PPTOD, we evaluate it in a more challenging low-resource scenario. Following previous studies, we train our model on MultiWOZ 2.0 by varying the percentage of training data, ranging from 1% (~80 samples) to 20% (~1600 samples). We compare our model with several strong baselines, including MD-Sequicity, DAMD, SOLOIST, and MinTL.\footnote{We did not compare results with TOP+NOD \cite{Liu2021} since the authors did not release their code and models.}

In each low-resource setting, we train our model five times with different random seeds and different selection of training data. The average scores over five runs are presented in Table 3.\footnote{Detailed numerical results can be found in Appendix B.} As seen, PPTOD consistently outperforms all baseline models by a large margin. Notably, our performance gain is even larger when fewer samples are used for training. This indicates that PPTOD better leverages the prior knowledge from pre-training therefore achieving better results in the extreme low-resource settings. Furthermore, with 20% of training data, PPTOD can achieve results that are comparable to the scores of systems like SOLOIST that are trained with full dataset as reported in Table 2.

### 4.2 Dialogue State Tracking

Next, we evaluate PPTOD for the dialogue state tracking task. The experiments are conducted on the benchmark MultiWOZ 2.0 \cite{Budzianowski2018} and 2.1 \cite{Eric2020} datasets. For evaluation, the joint goal accuracy is reported.
We compare PPTOD with a wide range of existing methods that can be categorized into two classes: (1) classification-based approaches and (2) generation-based approaches. Table 4 shows the DST results. Compared to other generation-based approaches, PPTOD\textsubscript{large} obtains the highest accuracy on both datasets. The performance of our model is lower than the SOTA classification-based approaches. However, these methods operate on a fixed ontology and perform prediction over a pre-defined set of slot-value pairs (Zhang et al., 2019; Chen et al., 2020; Shan et al., 2020; Zhou et al., 2021). This idea of fixed ontology is not scalable, as in real world applications, the ontology is subject to constant change (Heck et al., 2020). In contrast, PPTOD directly generates the outputs, making it more adaptive and generalizable to new ontology labels in real world applications.

### 4.2.2 Low-Resource Evaluation

To investigate how well PPTOD performs with limited training samples on the downstream task, we evaluate it in a simulated low-resource setting. Specifically, we train the model on MultiWOZ 2.0 by varying the percentage of training data (i.e., 1%, 5%, 10%, and 20%). We compare PPTOD with three strong generation-based baselines, including SimpleTOD, MinTL, and SOLOIST, using the official code released by the authors.

Table 5 shows the experimental results. As seen, in all settings, PPTOD outperforms other baselines by a large margin. In the extreme scenario, with only 1% of training data, PPTOD surpasses the strongest SOLOIST model by 18 points of accuracy. This demonstrates that our model is more generalizable and can be better applied to new tasks where the amount of training data is limited.

#### 4.3 Intent Classification

The goal of intent classification, i.e., NLU, is to classify the user’s intent based on the user’s utterance. We conduct experiments on the benchmark Banking77 dataset (Casanueva et al., 2020) that contains data with 77 different intents. Following previous studies (Casanueva et al., 2020; Peng et al., 2021), we test our model in both full training and low-resource settings. In the low-resource setting, we vary the number of training samples per intent from 10 to 30. The standard classification accuracy is reported for evaluation.

We compare PPTOD with several strong baselines, including BERT-Fixed, BERT-Tuned, USE+ConveRT (Casanueva et al., 2020), USE (Yang et al., 2020), ConveRT (Henderson et al., 2020), and SOLOIST (Peng et al., 2021). It is worth mentioning that all compared baselines are classification-based approach that uses classifier with a softmax layer to make the prediction over the pre-defined intent set. In contrast, as described in section §3.2, PPTOD solves the classification task as a generation problem by directly generating the text of intent label. Therefore, when adapting to a new classification task, PPTOD is more flexible and no extra model parameters are required.

In the experiments, we train PPTOD for five runs with different selection of training data and random initializations.

Table 5: Low-resource DST Evaluation: The means and standard deviations over five runs are reported.

Table 4: DST results. †: the models require a full pre-defined ontology for all possible domain-slot pairs.
Table 6: Comparison between plug-and-play and cascaded generation. ↑: higher is better and ↓: lower is better.

Table 7: Results on Banking77 dataset. † and ‡ are cited from Casanueva et al. (2020) and Peng et al. (2021).

5 Further Analysis

In this section, we present further discussions and empirical analyses of the proposed model.

5.1 Plug-and-Play vs Cascaded Generation

First, we compare our plug-and-play generation with the cascaded generation that is adopted by most existing studies. To this end, we fine-tune a T5-small model (without dialogue multi-task pre-training) on MultiWOZ 2.0 by either using the plug-and-play or the cascaded formulation. Moreover, we also examine the effect of DB state on the model performance. Specifically, for the plug-and-play model, when utilizing DB state, it first predicts the dialogue state (DST) to retrieve the DB state from the pre-defined database. Then, based on the DB state and dialogue context, the output of POL and NLG are generated in parallel. When ignoring the DB state, the plug-and-play model generates DST, POL, and NLG results in a fully paralleled fashion.

For evaluation, we report the results on end-to-end dialogue modelling task. In addition, we report the average inference latency and relative speedup of each model. We compare our ablated models with two strong baselines, SOLOIST and MinTL. Table 6 presents the results. As seen, the plug- and-play models yield better results than their cascaded counterparts. One reason is that, for cascaded models, the previously generated results are explicitly used as model input for latter sub-tasks, which leads to error accumulation. Moreover, we see that using DB state generally improves the model performance for both plug-and-play and cascaded models as it provides the model with more grounding information. Furthermore, with DB state, our plug-and-play model achieves better overall score than MinTL with an around 4× speedup. This suggests that the plug-and-play formulation benefits the model both in terms of the generation accuracy as well as the inference latency.

5.2 Multi-Task Pre-Training Investigation

Next, we provide further analyses on the dialogue multi-task pre-training strategy. To quantify the importance of different pre-training data, we pre-train the T5-small model using data that is annotated for individual TOD-related task (i.e., NLU, DST, POL, and NLG). After pre-training, we then evaluate the models on three downstream TOD tasks using MultiWOZ 2.0 and Banking77 datasets. For end-to-end dialogue modelling and dialogue state tracking, we test the model in both 1% and full training settings. For intent classification, we measure the accuracy of models trained with either 10 training samples per intent or full training samples.

Table 8 presents the results with the first row showing the performance of vanilla T5-small model. As seen, without any pre-training, the
vanilla T5-small model performs poorly in the low-resource setting of all evaluated tasks. This suggests that the prior knowledge from pre-training is indispensable for the model to achieve strong performances in the low-resource scenarios.

Moreover, we see that pre-training with data annotated for individual TOD-related task helps the model to attain better result in the corresponding downstream task. For example, pre-training with DST data notably improves the model performance in the downstream DST task both in low-resource and full-training settings. Similarly, pre-training with NLG data helps the model to get better BLEU score in the end-to-end dialogue modelling task.

Lastly, we see that the PPTODsmall model attains the best results on most of the evaluation metrics. This suggests that the pre-training data with different annotations are compatible with each other and the joint utilization of all pre-training data helps the model to achieve the best overall performance.

### 5.3 Human Evaluation

We also conduct a human evaluation with the help of graders proficient in English using an internal evaluation platform. For evaluation, we randomly selected 50 dialogue sessions from the test set of MultiWOZ 2.0 dataset. We compare the results generated by the PPTODbase model against the results from the SOLOIST model. All generated results, plus the reference, are evaluated by five graders on a 3-point Likert scale (0, 1, or 2) for each of the following features:

- **Understanding**: Whether the system correctly understands the user’s goal.
- **Truthfulness**: Whether the system’s response is factually supported by the reference.
- **Coherency**: Whether the system’s response is semantically coherent with the context.

More evaluation details are provided in the Appendix C.

Table 9 lists the results, with the first row showing strong inter-annotator agreements as measured by Fleiss’ kappa coefficient (Fleiss et al., 1971). Comparing with SOLOIST, our model achieves better scores on all metrics. Moreover, on the truthfulness and coherency metrics, our model significantly outperforms SOLOIST as judged by Sign Test (p-value < 0.05), suggesting that PPTOD generates more factually correct and semantically coherent responses. Finally, we note that on the fluency metric, both systems perform comparably with the reference (p-value > 0.4). This shows that the fluency of such systems is largely guaranteed by the prior syntactic knowledge from pre-trained language models, which suggests that future research should focus more on the other aspects of dialog systems.

### 6 Conclusion

In this paper, we propose PPTOD, a unified model that supports both task-oriented dialogue understanding and response generation in a plug-and-play manner. In addition, we introduce a new dialogue multi-task pre-training strategy to further augment our model’s ability in completing TOD-related tasks. Extensive experiments and analysis are conducted on three benchmark TOD tasks in both high-resource and low-resource settings. The automatic and human evaluations demonstrate that PPTOD outperforms the current SOTA systems in terms of various evaluation metrics.
References


Xiujun Li, Sarah Panda, JJ (Jingjing) Liu, and Jianfeng Gao. 2018. Microsoft dialogue challenge: Building...
end-to-end task-completion dialogue systems. In SLT 2018.


A Dataset Details

We elaborate the details of the dialogue datasets contained in the pre-training dialogue corpora.

- **MetaLWOZ** (Lee et al., 2019b) is designed for improving models’ ability in generating natural language responses in unseen domains. It contains annotations for natural language generation (NLG) spanning over 47 domains.

- **SNIPS** (Coucke et al., 2018) is designed to help developing models capable of understanding users’ intent (i.e., natural language understanding (NLU)). Its data consists of users’ utterances gathered by crowdsourcing with over 20 intent labels across 9 domains.

- **CLINC** (Larson et al., 2019) is built for improving model’s ability in detecting out-of-scope users’ intents. It contains data with NLU annotations for 150 intents across 10 different domains.

- **ATIS** (Amin, 2019) is used for building intent classification (NLU) model. It contains data with 22 user intents from the airline travel information domain.

- **KVRET** (Eric et al., 2017) is an in-car personal assistant dataset with dialogues from three domains: calendar scheduling, weather information retrieval, and point-of-interest navigation. It contains annotations for user belief state (DST) and system response (NLG).

- **WOZ** (Mrkšić et al., 2017) and **CamRest676** (Wen et al., 2017) are collected with Wizard-of-Oz procedure. They contain dialogues with DST and NLG annotations from the restaurant domain.

- **MSR-E2E** (Li et al., 2018) contains dialogues from three domains, including movie-ticket booking, restaurant reservation, and taxi booking. The data are annotated for three TOD-related tasks: DST, POL, and NLG.

- **Frames** (El Asri et al., 2017) contains dialogues from the trip booking domain. Its data are annotated for three TOD-related tasks, including DST, POL, and NLG.

- **TaskMaster** (Byrne et al., 2019) includes dialogues from six domains. Its data is collected with Wizard-of-Oz and self-dialogue approaches. The dataset is annotated with DST, POL, and NLG.

- **Schema-Guided** (Rastogi et al., 2020) is used for the DSTC8 (Kim et al., 2019) dialogue competition. It contains dialogues from 17 domains and it supports three TOD-related tasks, including DST, POL, and NLG.

B Low-Resource MultiWOZ Evaluation

In Table 10, we show the results of our model on MultiWOZ 2.0 under different low-resource settings. To get more confident results, for each setting, we train our model for five runs with different selection of training data and different random seeds. The complete results along with the mean and standard deviations are presented in Table 10.

C Human Evaluation Guidelines

Please evaluate the system’s response with respect to the following features: (1) Understanding; (2) Truthfulness; (3) Coherency; and (4) Fluency. In the following, we provide some guidelines regarding how to judge the quality of the system’s response in terms of different features.

C.1 Understanding

This metric measures whether the system’s response shows that the system is able to understand the goal and intent of the user. The definition of different scores are:

- 2: The system completely understands the user’s goal and intent.
- 1: The system partially understands the user’s goal and intent.
- 0: The system does not understand the user’s goal and intent at all.

C.2 Truthfulness

This metric measures whether the system’s response is factually supported by the reference response. The definition of different scores are:

- 2: The facts in the system’s response are all supported by or can be inferred from the reference response.
- 1: The facts in the system’s response are partially supported by the reference response.
- 0: The system’s response is contradicted to the facts contained in the reference response.
Table 10: Low-Resource Experiments on MultiWOZ: The average and std rows show the mean and standard deviation of results from five different runs. The Succ. and Comb. denote Success and Combined Score, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>1% of training data</th>
<th>5% of training data</th>
<th>10% of training data</th>
<th>20% of training data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inform Succ BLEU Comb</td>
<td>Inform Succ BLEU Comb</td>
<td>Inform Succ BLEU Comb</td>
<td>Inform Succ BLEU Comb</td>
</tr>
<tr>
<td>run-1</td>
<td>68.50 54.90 13.98 75.68</td>
<td>78.40 61.30 14.78 84.73</td>
<td>79.70 68.20 17.10 91.30</td>
<td>83.40 71.10 17.05 94.30</td>
</tr>
<tr>
<td>run-2</td>
<td>64.70 50.20 12.19 69.64</td>
<td>75.20 61.30 15.85 84.10</td>
<td>87.00 67.30 13.89 91.04</td>
<td>82.80 69.80 17.03 92.88</td>
</tr>
<tr>
<td>run-3</td>
<td>65.30 46.10 10.79 66.49</td>
<td>75.40 60.80 15.99 84.09</td>
<td>84.30 68.10 15.33 91.50</td>
<td>83.70 70.00 17.01 93.61</td>
</tr>
<tr>
<td>run-4</td>
<td>64.80 51.00 12.43 70.33</td>
<td>77.20 59.70 15.75 84.20</td>
<td>84.50 71.90 14.51 92.71</td>
<td>82.40 69.40 17.93 93.83</td>
</tr>
<tr>
<td>run-5</td>
<td>71.50 52.30 13.14 75.04</td>
<td>76.70 64.70 14.37 85.07</td>
<td>78.00 64.90 16.99 88.44</td>
<td>83.00 70.10 16.10 92.65</td>
</tr>
<tr>
<td>average</td>
<td>66.96 50.90 12.51 71.44</td>
<td>76.58 61.60 15.35 84.44</td>
<td>83.50 68.18 15.86 91.01</td>
<td>82.96 69.90 17.02 93.45</td>
</tr>
<tr>
<td>std 2.67 2.88 1.06 3.46</td>
<td>1.18 1.67 0.65 0.39</td>
<td>3.33 2.26 1.29 1.40</td>
<td>0.34 0.74 0.58 0.61</td>
<td></td>
</tr>
</tbody>
</table>

C.3 Coherence

This metric measures whether the system’s response is logically coherent with the dialogue context. The definition of different scores are:

- 2: The system’s response is logically coherent with the dialogue context.

- 1: The system’s response contains minor information that is off the topic of the dialogue context.

- 0: The system’s response is completely irrelevant to the dialogue context.

C.4 Fluency

The metrics measures the fluency of the system’s response. The definition of different scores are:

- 2: The system’s response is grammatically correct and easy to understand.

- 1: The system’s response contains minor errors but they do not affect your understanding.

- 0: The system’s response does not make sense and it is unreadable.

D Case Study

Table 11 presents a generated dialogue example from the PPTODbase model. The user starts the conversation by asking for an expensive restaurant that serves Indian food for dinner. PPTOD finds 14 restaurants that satisfy the user’s goal and asks the user for a preferred location. We can see that, when the user states no preference on the restaurant location, PPTPD correctly updates the dialogue state by adding the area information which is missed by the oracle information. Then the user switches the dialogue topic for booking a hotel. Through the dialogue trajectory, we see that PPTOD completes the dialogue by successfully providing the user the necessary information such as number of hotel choices (at turn 3) and the booking reference number (at turn 6). When finding the user’s booking request cannot be fulfilled (at turn 5), the models ask the user for an alternative option. Moreover, this example also demonstrates that PPTOD is able to deal with some NLU challenges displayed in the conversations. For example, at turn 4, the user already provides the information about the Gonville Hotel. But only after the user describes the intention of booking the hotel at turn 5, the model updates the name of hotel in the dialogue state based on the co-referenced information from the previous turn. Interestingly, the hotel name is ignored by the oracle dialogue state but our model correctly detects it. The dialogue understanding ability of PPTOD can also be observed in turn 6, in which it updates the hotel stay in the belief state from 2 days to 1 day after the user provides the corresponding information.
<table>
<thead>
<tr>
<th>Turn</th>
<th>User Utterance</th>
<th>Generated Belief State</th>
<th>Oracle Belief State</th>
<th>Delexicalized Response</th>
<th>Lexicalized Response</th>
<th>Oracle Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>i am in need of an expensive indian restaurant for dinner .</td>
<td>[restaurant] {food = indian, pricerange = expensive}</td>
<td>[restaurant] {food = indian, pricerange = expensive}</td>
<td>there are [value_choice] [value_price] [value_food] restaurants in cambridge . what area would you like to dine in ?</td>
<td>there are 14 expensive indian restaurants in cambridge . what area would you like to dine in ?</td>
<td>there are 14 such restaurants . do you want to be in a specific part of town ?</td>
</tr>
<tr>
<td>2</td>
<td>i don't care what part of town it is in , i would like a phone number please .</td>
<td>[restaurant] {food = indian, pricerange = expensive, area = don't care}</td>
<td>[restaurant] {food = indian, pricerange = expensive}</td>
<td>[value_name] is a [value_food] restaurant in the [value_area] part of town . their phone number is [value_phone] .</td>
<td>curry garden is a indian restaurant in the center part of town . their phone number is 01223302330 . would you like me to reserve a table for you ?</td>
<td>sure , i am sorry . i have curry garden their phone number is 01223302330 . would you like me to reserve a table for you ?</td>
</tr>
<tr>
<td>3</td>
<td>not at this time . i am looking for a 3-star hotel , though . no guest house .</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>i have [value_choice] [value_stars] star [value_type] in the [value_area] . [value_name] is a [value_stars] star [value_type] . would you like to book a room ?</td>
<td>i have 2 3-star hotel in the center . gonville hotel is a 3-star hotel . would you like to book a room ?</td>
<td>i have found the gonville hotel . it is a 3 star hotel . would you like me to book it for you ?</td>
</tr>
<tr>
<td>4</td>
<td>can you book a room for 2 people for 2 nights starting on sunday ?</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>yes , it is in the [value_price] price range . would you like me to book it for you ?</td>
<td>yes , it is in the expensive price range . would you like me to book it for you ?</td>
<td>yes , it is an expensive hotel .</td>
</tr>
<tr>
<td>5</td>
<td>could you try it for 1 night ? if that works , please book it and get the reference number .</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>i am sorry , but the [value_name] is not available for that stay . would you like to try a shorter stay ?</td>
<td>i am sorry , but the gonville hotel is not available for that stay . would you like to try a shorter stay ?</td>
<td>the booking was unsuccessful . would you like to try for another day or a shorter stay ?</td>
</tr>
<tr>
<td>6</td>
<td>you are all set for 1 night ( sunday ) , reference number is 6wvv053q . would you like to try another location for monday night ?</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>i was able to successfully book your room . your reference number is [value_reference] . is there anything else i can help you with ?</td>
<td>i was able to successfully book your room . your reference number is 7rzme01z . is there anything else i can help you with ?</td>
<td>you are all set for 1 night ( sunday ) , reference number is 6wvv053q . would you like to try another location for monday night ?</td>
</tr>
<tr>
<td>7</td>
<td>you are welcome . have a great day ! thank you , and goodbye .</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>[hotel] {stars = 3, type = hotel}; [restaurant] {food = indian, pricerange = expensive}</td>
<td>you are welcome . have a great day !</td>
<td>you are welcome . have a great day !</td>
<td>thank you , and goodbye .</td>
</tr>
</tbody>
</table>

Table 11: An generated dialogue example from the PPTOD\textsubscript{base} model. (best viewed in color)