Enhancing Dialogue State Tracking Models through LLM-backed User-Agents Simulation

Cheng Niu[1], **Xingguang Wang**[1], **Xuxin Cheng**[1], **Juntong Song**[1], and **Tong Zhang**[2]

[1]NewsBreak
[2]University of Illinois Urbana-Champaign
cheng.niu@newsbreak.com

Abstract

Dialogue State Tracking (DST) is designed to monitor the evolving dialogue state in the conversations and plays a pivotal role in developing task-oriented dialogue systems. However, obtaining the annotated data for the DST task is usually a costly endeavor. In this paper, we focus on employing LLMs to generate dialogue data to reduce dialogue collection and annotation costs. Specifically, GPT-4 is used to simulate the user and agent interaction, generating thousands of dialogues annotated with DST labels. Then a two-stage fine-tuning on LLaMA 2 is performed on the generated data and the real data for the DST prediction. Experimental results on two public DST benchmarks show that with the generated dialogue data, our model performs better than the baseline trained solely on real data. In addition, our approach is also capable of adapting to the dynamic demands in real-world scenarios, generating dialogues in new domains swiftly. After replacing dialogue segments in any domain with the corresponding generated ones, the model achieves comparable performance to the model trained on real data[1].

1 Introduction

Dialogue state tracking (DST) is a critical component of task-oriented dialogue systems, serving to track users' goals and system actions in the conversation and facilitate precise information handling for communicating with external APIs (Henderson et al., 2014; Mrkšić et al., 2017; Zhang et al., 2023; Hudeček and Dušek, 2023). DST task usually applies the form of key-value pairs, where the keys are denoted as slots which are defined in the system schema, outlining the specific information that the system aims to track or extract information during the whole conversation (Ren et al., 2018).

The design of DST approaches could be broadly categorized into two main types, the *classification-based DST models* and *generation-based DST models*. *Classification-based models* choose slot values from candidates (Ma et al., 2019; Ye et al., 2021), assuming that the dialogue ontology is pre-defined and hence lacking generalization capability (Chen et al., 2020; Wang et al., 2022). *Generation-based models* directly generate the slot values to handle unseen domains and values (Gao et al., 2019, 2020; Lin et al., 2020; Peng et al., 2021). Recently, Feng et al. (2023) proposes a novel framework, LDST, based on LLaMA (Touvron et al., 2023a). By using an instruction tuning method, LDST achieves performance on par with ChatGPT (OpenAI, 2023).

Despite DST showing promising results, a significant challenge is that the annotation of dialogues entails significant costs. Furthermore, the dynamic nature of real-world demands highlights the urgent need to quickly generate more utterances for new domains. Compared to the other types of NLP data, collecting the authentic dialogue data is particularly challenging. This difficulty is partly due to the dialogues frequently containing personal or sensitive information, which complicates data collection and sharing efforts. In response to these two challenges, and inspired by the recent advancements of large language models (LLMs) (Touvron et al., 2023b; Significant-gravitas, 2023; Jablonka et al., 2023; Shen et al., 2023), we try the application of these LLMs for generating annotated DST data for data augmentation. By leveraging LLM's cross-domain generation capability, we aim to create synthetic dialogues that could serve as replacements for manually annotated data, significantly alleviating both financial cost and time constraints.

In this paper, we propose a **L**LM-backed **U**ser-**A**gents **S**imulation (LUAS) algorithm to enhance DST. The process begins with the LLM generating the user profile that details the individual's preferences for different tasks. Following this initial step, the LLM is prompted to simulate a conversation between the user and the agent. In these simulations,

[1]The source code and generated dialogue data are available at `https://github.com/ParticleMedia/LUAS`.

the user simulator makes requests and seeks recommendations or assistance, while the agent responds by understanding the user's needs, providing suggestions, and taking appropriate actions. Through iterative conversations between the user and agent, complemented by a slot extractor also prompted by the LLM, we could generate a substantial corpus of labeled, multi-turn dialogue data. In particular, it is essential to perform consistency checks between the extracted slots and the true values of the user's requests or entity attributes to ensure the quality of the generated dialogue data.

To verify the effectiveness of our approach and the quality of the generated data, experiments are conducted on two public DST datasets, MultiWOZ 2.2 (Zang et al., 2020) and MultiWOZ 2.4 (Ye et al., 2022) datasets. Following Touvron et al. (2023b), LLaMa 2 is finetuned with real data as the strong baseline. By using both the generated data and real data, finetuning LLaMa 2 can further improve the performance. Besides, by replacing dialogue segments of any domain with the generated data, the newly trained model achieves comparable performance to the model trained on the real data, which shows the capability of our approach to meet the dynamic requirements of real-world scenarios, generating dialogues in new domains and preserving the promising high performance.

In summary, the contributions of our work can be categorized into the following aspects:

- We propose a new framework that harnesses the power of GPT-4 to generate new labeled dialogue data, effectively reducing the dialogue data collection and annotation costs.
- Consistency checks are designed to compare the extracted slots with the true values of the user's requests or entity attributes, ensuring the quality of the generated data.
- Experiment results on two datasets show the positive impact of the generated data on performance.
- Our method can swiftly generate data in new domains while maintaining promising performance.

2 Related Work

2.1 Dialogue State Tracking

Dialogue state tracking (DST) is essential yet challenging in task-oriented dialogue systems (Mrkšić et al., 2017). Recent DST models (Lee et al., 2021; Zhu et al., 2022; Yang et al., 2023b; Su et al., 2023; Lesci et al., 2023), leveraging the different architectures and mechanisms, have convincingly demonstrated promising performance on several datasets (Budzianowski et al., 2018; Eric et al., 2020; Zang et al., 2020; Han et al., 2021; Ye et al., 2022). To ease the burden of dialogue collection and annotation, Wu et al. (2019); Zhou et al. (2023) use few-shot learning to transfer to adapt existing models to the new domains. Drawn by the recent achievement of LLMs, Feng et al. (2023) leverages Low-Rank Adaptation (LoRA) (Hu et al., 2022) to fine-tune the foundation model, achieving the promising performance in DST. In this paper, we utilize GPT-4 to simulate user-agent conversations, and the obtained dialogue data significantly enhances DST.

2.2 Data Augmentation by LLMs

Data augmentation has shown remarkable effectiveness in various domains, including computer vision (Krizhevsky et al., 2012; Shorten and Khoshgoftaar, 2019), text classification (Zhang et al., 2015; Wei and Zou, 2019), and speech processing (Ko et al., 2015; Park et al., 2019; Cheng et al., 2023).

In recent years, with the increasing prominence of LLMs, an increasing number of studies begin to leverage LLMs for data augmentation. Kaddour and Liu (2024) discovers that fine-tuning teacher LLMs to annotate unlabeled instances and generate the new data points can notably enhance the performance of downstream models. Yang et al. (2023a) generates truthful and customized dialogues to reduce hallucination. Ulmer et al. (2024) compares the effectiveness of various filtering strategies for the generated dialogue quality and introduces a new method to benchmark a finetuned dialogue system. But their work does not discuss the DST task. Li et al. (2022) presents a GPT-3-backed user-agent simulation system and shows the positive results on the DST task when the real data size is extremely small. Unlike Li et al. (2022), we directly abstract the common intentions of users and agents, crafting intent-specific prompts to ensure that this simulation adheres to the task-oriented logic. Our strategy enables the simulation to operate within a zero-shot setup, enhancing our approach's adaptability to new domains. Moreover, by implementing a two-stage fine-tuning process, our approach demonstrates superior performance compared to strong baselines, even when trained with the full size of real data.

3 Method

In this section, we will begin with the basic problem definition (§3.1). Then, we introduce our pro-

DST: [***history***], [***user_utterance***] $\rightarrow$ [***service***], [***slot_key***], [***slot_val***]
You are a local guide online, primarily handling the local services like finding the user's place (such as attraction, hotel, train, restaurant, or hospital), calling taxis, contacting the police, or other convenient services. Your service is efficient and of high quality, earning widespread praise from the local community. Given the conversion history, your task is to help find what the user is looking for based on the whole conversion. Please output the current_service based on the user's last utterance. And also please output all service information that needs to be paid attention to from the whole conversion. Here are the "conversion history": {[***history***]} and the "user's lastest utterance": {[***user_utterance***]}. The output should be JSON-formatted like "current_service": {[***service***]}, "slots": {"[***service***]": {"[***slot_key***]": {[***slot_val***]}}}. Please give your decision:

Table 1: Proposed prompts to guide LLaMA 2 to generate JSON-formatted dialogue state predictions.

posed method, including fine-tuning LLaMA 2 to predict dialogue state (§3.2) and utilizing GPT-4 for user-agent conversation simulation (§3.3). Finally, we present our two-stage fine-tuning strategy to use both generated and real data for DST (§3.4).

3.1 Problem Definition

A task-oriented dialogue involves a multi-turn conversation between a user U and an agent A. Given a dialogue context $C_t = [U_1, A_1, ..., U_t, A_t]$ as the sequence of utterances up to turn t, the goal of DST is to predict the dialogue state y_t, which is defined as a collection of *(slot, value)* pairs:

$$y_t = \{(s_t^i, v_t^i) \mid C_t \ , \forall s^i \in \mathcal{S}\}$$

where $\mathcal{S}$ denotes the set of the possible slots predefined in an ontology or schema. Following previous work (Ulmer et al., 2024), the output slot is represented as a simple concatenation of the corresponding task domain and slot, e.g., "<hotel-area>". The slot values associated with each domain could be either categorical with a set of pre-defined candidates (e.g. <hotel-parking> = "True" / "False"), or non-categorical, where the value is the span in the dialogue context (e.g. <hotel-name> = "Alexander"). Note that if no information is provided in the dialogue regarding a specific slot, the associated value for that slot is set to "NONE".

3.2 Using LLaMA 2 to Predict Dialogue State

We employ full-parameter fine-tuning on LLaMA 2 to predict dialogue states and employ pre-designed prompts to guide the LLaMA 2 model in generating predictions formatted in JSON. As demonstrated in Table 1, dialogue history and the user's latest utterance are fed into LLaMA 2, which then conducts the prediction of the entire conversation's intents and slot values. Specifically, predicted intents must fall within a predefined set, and the predicted slots must align with designated slots for respective intents. We utilize a schema to prevent the generation model from producing incoherent outputs and to enhance the overall quality and reliability of the outputs of LLaMA 2. The optimization is conducted through the utilization of cross-entropy.

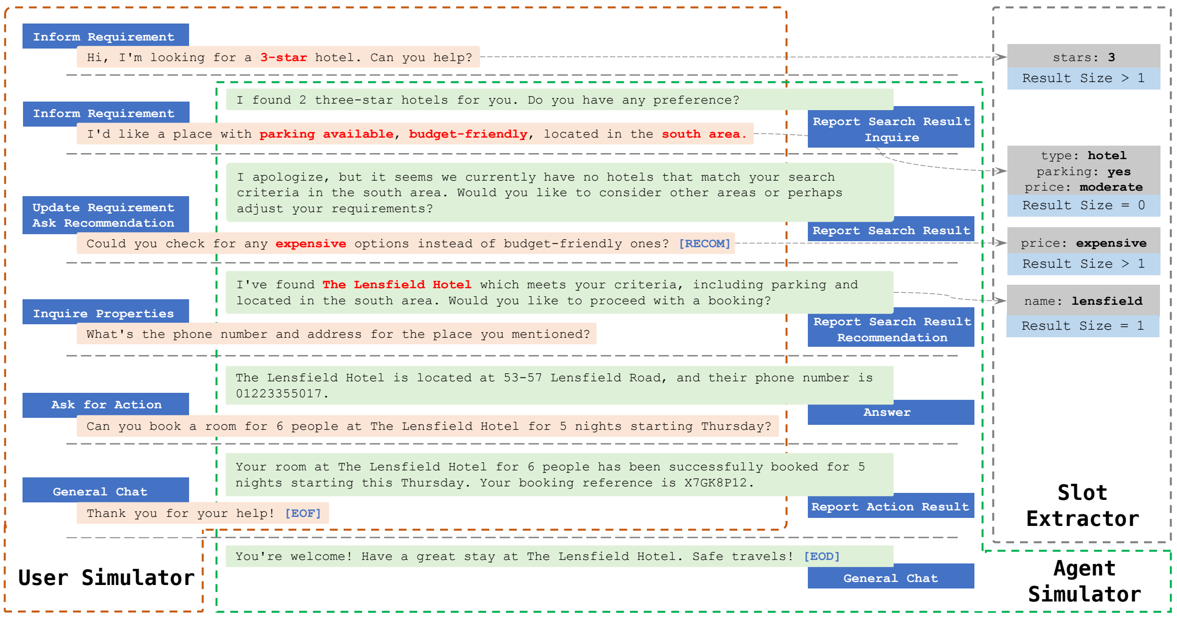
3.3 User-Agent Dialogue Simulation backed by GPT-4

As illustrated in Figure 1, the dialogue simulation framework based on GPT-4 involves a multi-stage approach for producing labeled multi-turn dialogue data. In this arrangement, GPT-4 prompts two simulators, including the **user simulator** and the **agent simulator**, to engage in the conversations aimed at completing specific dialogue tasks. Concurrently, GPT-4 also prompts a **slot extractor** to identify and extract all relevant slots throughout the entire conversation simulation process.

The details of the simulation generation process are outlined below, with all the prompts included in the Appendix for reference.

3.3.1 Simulation Process Overview

Before initiating the dialogue, GPT-4 is prompted to create a user profile that outlines the individual's preferences across various tasks such as travel, accommodations, dining, and more. Each preference includes specific details like budget, travel distance, and the other criteria. Following this setup, the user simulator begins interacting with the agent, presenting its requests, and seeking recommendations or assistance with bookings and purchases. The agent, in turn, is prompted to delve into the user's needs, conduct searches for pertinent information, offer suggestions, and execute necessary actions. After each interaction, the user simulator evaluates how well their needs have been met, deciding whether


Hi, I'm looking for a 3-star hotel. Can you help?

I found 2 three-star hotels for you. Do you have any preference?

I'd like a place with parking available, budget-friendly, located in the south area.

I apologize, but it seems we currently have no hotels that match your search criteria in the south area. Would you like to consider other areas or perhaps adjust your requirements?

Could you check for any expensive options instead of budget-friendly ones? [RECOM]

I've found The Lensfield Hotel which meets your criteria, including parking and located in the south area. Would you like to proceed with a booking?

What's the phone number and address for the place you mentioned?

The Lensfield Hotel is located at 53-57 Lensfield Road, and their phone number is 01223355017.

Can you book a room for 6 people at The Lensfield Hotel for 5 nights starting Thursday?

Your room at The Lensfield Hotel for 6 people has been successfully booked for 5 nights starting this Thursday. Your booking reference is X7GK8P12.

Thank you for your help! [EOF]

You're welcome! Have a great stay at The Lensfield Hotel. Safe travels! [EOD]

Figure 1: The simulation process of our approach. The blue boxes are intents for the user and agent, the '[RECOM]', '[EOF]', and '[EOD]' are three control identifiers to specify expected responses from the agent.

to continue the conversation.

3.3.2 User/ Agent Intents

To effectively navigate the simulators through the interactive tasks, we encounter the enormous challenge of encoding complex dialogue logic within only a single prompt, which is demanding for both the user and the agent simulator. To simplify, we abstract the common intents of users and agents, and craft prompts specifically for each unique intent of the user or agent. Detailed prompts of different intents are demonstrated in Appendix A.

The user's intents are listed below:

- **Inform Requirement**, the user informs their requirement to the agent.
- **Update Requirement**, the user may update their requirements if the search result does not meet their criteria.
- **Ask for Recommendation**, the user asks for a recommendation given a few candidates meeting their criteria.
- **Inquire Properties**, the user asks for some properties (e.g. address, etc.) of the candidates.
- **Ask for Action**, the user requires an action upon receiving a recommendation.
- **General Chat**, other scenarios in the simulation, e.g. greeting or showing gratitude.

The agent's intents are listed below:

- **Inquire**, ask the user's need and preference or seek the user's approval or confirmation.
- **Report Search Results**, based on the user's preference, search the whole database and then make inquiries, recommendations, or reservations.
- **Recommendation**, when more than one candidate meets users' search criteria, choose the top candidate to recommend.
- **Answer**, answer the user's inquiry about a recommendation from the agent.
- **Report Action Result**, take action per the user's request and report the outcome of the action.
- **General Chat**, other scenarios in the simulation, e.g. greetings or asking if there are any additional requirements to be addressed.

Besides natural language outputs, the simulators are also prompted to generate specific control identifiers in the responses, signaling the intent of the response. Given an input intent signaled by the control identifiers, the user or the agent is prompted to select a proper intent and generate responses.

3.3.3 Simulation Details

As described in Sec. 3.3.1, the simulation begins by generating the user profile, which initializes the user requests. Following this, based on input intent from the preceding round, the simulation selects a user or agent response intent and then uses the corresponding prompt for dialogue generation. This selection process is governed by the predetermined logic listed below. The **General Chat** intent refers to the expressions of greetings and gratitude that are triggered only at the beginning or end of a conversation and are skipped in the subsequent list.

The conversation can be initiated by either the user or the agent. The following describes the detailed mechanism that triggers the user's intent.

- **Conversation Starts**: triggers user's **Inform Requirement** intent. Using randomization, the user simulator is instructed to choose a task of interest along with several related preferences and then generate a corresponding request.
- **Inquire** from the agent: triggers user's **Inform Requirement** intent, prompting responses to the follow-up questions related to the scenario.
- **Report Search Result** from the agent: if the user's preference has not been fully expressed, the user's **Inform Requirement** intent will be triggered. If no candidate meets the search criteria, this will trigger the user's **Update Requirement** intent. Otherwise, the presence of a single candidate will initiate the user's **Ask for Action** intent, and the discovery of multiple candidates will trigger the user's **Ask for Recommendation**.
- **Recommendation** from the agent: the user will be prompted to select from (i) **Inquire Properties** intent for more information or (ii) **Ask for Action** intent to proceed to make transactions.
- **Report Action Result** from the agent: if all the tasks in the user profile have been finished, **General Chat** between the user and the agent will be triggered, and then the conversation terminates. Otherwise, **Inform Requirement** intent is triggered for the following new task.

Below is the intent-triggering mechanism for the agent simulation.

- **Inform Requirement** from the user: the agent is prompted to check if all the required slot values have been collected. If not, **Inquire** intent will be triggered to output follow-up questions. Otherwise, the agent will search based on the user's requirement, and then generate a response based on **Report Search Result** intent.
- **Inquire Properties** from the user GPT-4: triggers agent's **Answer** intent.
- **Ask for Recommendation** from the user GPT-4: the agent is prompted to select the top candidate and then generate the response based on the **Recommendation** intent.
- **Ask for Action** from the user: the agent GPT-4 is prompted to make transactions and generate a response based on **Report Action Result** intent.

3.3.4 Slot Extraction

It's important to note that the agent simulator must verify that all the necessary information has been gathered before initiating a search. To manage this, a slot tracking module is employed to keep track of both the required slots and the filled slots. With the **Inform Requirement** prompt, the user simulator can simultaneously provide the dialogue utterances and the corresponding filled slot values. However, there is a possibility that the conversation generated by GPT-4 might not align with the outcomes of slot filling. Below is an example:

Agent: *I found several hotels that match your criteria. How many guests are staying in this hotel?*

User: *Yes, I need a 4-star hotel in the south area.*

Sometimes, the same two utterances are repeated and echoed back and forth multiple times, and the user simulator continues to generate the same but incorrect response, failing to supply the slot values requested by the agent simulator.

To address this problem, we utilize a slot extraction module powered by GPT-4 to ensure that the generated conversation aligns with the slot-filling expectations. If any inconsistency is detected, the conversation is regenerated to maintain coherence. This approach is also used to verify agent responses when users inquire about entity properties.

The process for measuring the inconsistency involves following steps. First, we normalize the extracted slot values, aligning them with one of possible values found either in the schema or within the entity database, (e.g. a restaurant name in the MultiWOZ database). Next, we compare these normalized values with the expected slot values through either a string match or numeric value comparison.

The entire procedure simultaneously generates dialogue content and tracks the dialogue states. By excluding samples with inconsistent states, we significantly mitigate the noise from LLM hallucinations, which might originate from either dialogue generation or slot extraction.

3.3.5 Generation Diversity

To obtain a DST model with high quality, it is very essential to have dialogue data that encompasses a wide range of diversity. To ensure the data generated possesses this diversity, we manually created ten rewriting templates, which were then expanded into hundreds of templates by GPT-4. These rewriting templates serve as a post-processing tool to enhance the diversity of the user and agent responses.

The details about the rewriting templates and rewritten outputs are shown in Appendix B.

3.4 Two-stage Fine-tuning Strategy

Taking into account the discrepancy in distribution between GPT-4 generated and real dialogues, directly merging generated and real data could cause the resulting model to deviate from the true distribution. To address this problem, we have designed a two-stage fine-tuning approach. Initially, we fine-tuned LLaMA 2 using the generated dialogue data. Following this, we continue to fine-tune the model with real data. The first step enables the model to learn fundamental task-oriented dialogue patterns. The second step ensures that the model effectively bridges the gap between the generated and real dialogues, aligning closely with the true distribution.

4 Experiments

4.1 Datasets and Metrics

We conduct all the experiments on MultiWOZ 2.2[2] (Zang et al., 2020) and MultiWOZ 2.4[3] (Ye et al., 2022). MultiWOZ (Budzianowski et al., 2018) has been extensively utilized for evaluating the performance of DST, including 8,438, 1,000, and 1,000 samples for training, dev, and test sets with multi-turn dialogues, which are collected by a Wizard-of-Oz (WOZ) setup and encompass a diverse array of domains. MultiWOZ 2.2 dataset refines the annotations in dev and test sets of MultiWOZ 2.1 (Eric et al., 2020). MultiWOZ 2.4 (Ye et al., 2022) is the latest refined version correcting all incorrect labels in dev and test sets. Following Wu et al. (2019), we remove the domains of ‘hospital’ and ‘police’ from both MultiWOZ2.2 and MultiWOZ2.4 datasets because they only appear a few times in the training set and never occur in the dev and test set. By using the MultiWOZ schema, nearly 8000 new dialogues are generated. The detailed statistics of MultiWOZ 2.2 and MultiWOZ 2.4 datasets and the generated dialogue data are demonstrated in Table 2.

We adopt Joint Goal Accuracy (JGA) as the evaluation metric, which is the primary metric for DST. JGA is defined as the proportion of dialogue turns in which all the key-values are correctly predicted.

[2]https://github.com/budzianowski/multiwoz/tree/master/data/MultiWOZ_2.2

[3]https://github.com/smartyfh/MultiWOZ2.4

Metric ↓ Dataset →	2.2	2.4	Generated
No. of domains	8	7	5
No. of dialogues	8,438	8,438	7,556
Total no. of turns	113,556	113,556	102,602
Avg. turns per dialogue	13.46	13.46	13.57
Avg. tokens per turn	13.13	13.38	17.01
No. of slots	61	37	17
Have schema description	Yes	Yes	-
Unseen domains in test set	No	No	-

Table 2: Statistics of MultiWOZ (2.2 and 2.4) and the generated dataset used for training in our experiments.

Models	MultiWOZ 2.2	MultiWOZ 2.4
TRADE	45.40	55.05
UniLM	54.25	-
DS-DST	51.70	-
TripPy	53.50	64.75
AG-DST	57.26	-
SDP-DST	57.60	-
$\text{D3ST}_{\text{Base}}$	56.10	72.10
$\text{D3ST}_{\text{Large}}$	54.20	70.80
D3ST_{XXL}	58.70	75.90
SPACE-3	57.50	-
MSP-L	57.70	-
RefPyDST	-	65.20
Diable	56.48	70.46
DDSA	-	75.58
SPLAT	56.60	-
MoNET	-	76.02
SSNet	62.10	-
$\text{TOATOD}_{\text{Small}}$	61.92	-
$\text{TOATOD}_{\text{Base}}$	63.79	-
LUAS_{R}	**65.42**	**77.20**
$\text{LUAS}_{\text{R+G}}$	**66.25**	**78.20**

Table 3: Joint Goal Accuracy for DST results on MultiWOZ 2.2 and MultiWOZ 2.4 dataset. ‘-’ denotes that the results are not reported in the original paper.

4.2 Implementation Details

The GPT-4 version utilized for simulation is gpt-4-1106-preview. As for the two fine-tuning stages, 8 Nvidia A100 (80G) GPUs are utilized for supervised full-parameter tuning with pytorch’s FSDP framework (Zhao et al., 2023). The base language model is 7B version[4] of LLaMA 2. For each fine-tuning stage, the learning rate is set to 2e-5 with the cosine scheduler (Loshchilov and Hutter, 2017), and the batch size is set to 8 on each GPU. We utilize Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and the warm-up ratio is set to

[4]https://huggingface.co/meta-llama/Llama-2-7b-hf

Metric ↓ Replaced Domain →	Attraction	Hotel	Restaurant	Taxi	Train
Replaced Dialogues	2538	3235	3666	1397	2840
Replaced Turns	13348	30402	25768	6662	33364
Avg. replaced turns per dialogue	5.26	9.40	7.03	4.77	11.75
Avg. tokens per replaced turn	15.57	15.54	15.33	18.28	16.44
Avg. slots per replaced user turn	1.38	2.75	2.54	1.37	2.90

Table 4: Substituting details for 5 domains of MultiWOZ 2.2.

Replaced Domain	Impact	JGA (Δ)	Slot Precision (Δ)	Slot Recall (Δ)	Slot F1 (Δ)
Base	0%	65.42	95.47%	93.25%	94.35%
Attraction	28.1%	64.99 (−0.43)	95.46% (−0.01%)	92.93% (−0.32%)	94.17% (−0.18%)
Hotel	42.1%	64.28 (−1.13)	95.22% (−0.25%)	92.83% (−0.42%)	94.01% (−0.34%)
Restaurant	41.2%	64.61 (−0.81)	95.44% (−0.03%)	93.30% (+0.05%)	94.36% (+0.01%)
Taxi	9.1%	65.22 (−0.20)	95.62% (+0.15%)	92.91% (−0.34%)	94.25% (−0.10%)
Train	38.4%	64.23 (−1.19)	95.59% (+0.12%)	92.67% (−0.58%)	94.11% (−0.24%)
Averaged	31.20%	64.67 (−0.75)	95.47% (−0.00%)	92.93% (−0.32%)	94.18% (−0.17%)

Table 5: JGA for substituting real data with generated data on MultiWOZ 2.2 dataset.

3%. Both fine-tuning stages last around two hours. For inference, vLLM[5] (Kwon et al., 2023) is used.

4.3 Baselines

To evaluate the efficacy of the generated dialogue data, we first fine-tune LLaMA 2 solely using real data, referring to it as LUAS$_{\text{R}}$, which serves as a strong DST baseline. We also conduct comparisons between our approach and other baselines, including TRADE (Wu et al., 2019), UniLM (Dong et al., 2019), DS-DST (Zhang et al., 2020), TripPy (Heck et al., 2020), AG-DST (Tian et al., 2021), SDP-DST (Lee et al., 2021), D3ST (Zhao et al., 2022), SPACE-3 (He et al., 2022), MSP-L (Sun et al., 2022), RefPyDST (King and Flanigan, 2023), Diable (Lesci et al., 2023), DDSA (Yang et al., 2023b), SPLAT (Bebensee and Lee, 2023), MoNET (Zhang et al., 2023), SSNet (Atawulla et al., 2023), TOATOD (Bang et al., 2023).

4.4 Results for DST prediction

The complete results are shown in Table 3, it needs to be pointed out that our method is primarily compared with the *generation-based models*, because *classification-based models* can utilize the external knowledge, leading to unfair comparisons. LUAS$_{\text{R}}$ is only fine-tuned on the real data, and LUAS$_{\text{R+G}}$ is fine-tuned on both real and generated data. From these results, we have the following observations:

(1) On both MultiWOZ 2.2 and 2.4 datasets, the performance of LLaMA 2 fine-tuned on real data (LUAS$_{\text{R}}$) surpasses previous DST baselines. This shows the exceptional effectiveness of LLaMA 2.

(2) Furthermore, the incorporation of additional generated data yields significant performance improvements, with enhancements of 0.83% on MultiWOZ 2.2 and 1.00% on MultiWOZ 2.4. This improvement emphasizes the important role of generated data in enhancing overall model performance. As shown in the following section, the gain from the generated data can be even larger in case the real dialogue data is of a smaller size. For example, the enhancement can be as large as from 4.29% to 5.09% if only 1,000 dialogue real data exists. Considering the challenge in dialogue data collection, this result highlights the pragmatic significance of integrating generated data for DST across domains.

4.5 Results of Substituting Real Data with Generated Data

In order to further validate the quality and effectiveness of the generated dialogue data, we conduct a data replacement experiment for different domains on MultiWOZ 2.2. In these experiments, all dialogue data segments related to a specific domain will be removed, and the newly generated data will be inserted at the removed location. After replacement, the new training set will consist of 1 domain with the generated data and 4 others with real data. The replacement details are shown in Table 4.

[5]https://github.com/vllm-project/vllm

Dataset	Real Data Size	JGA_{R}	$\text{JGA}_{\text{R+G}}$ (Δ)	Slot		
				$\text{Precision}_{\text{R+G}}$ (Δ)	$\text{Recall}_{\text{R+G}}$ (Δ)	$\text{F1}_{\text{R+G}}$ (Δ)
MultiWOZ 2.2	1000	58.77	63.06 (+**4.29**)	95.06% (+**0.69%**)	92.39% (+**1.46%**)	93.70% (+**1.08%**)
	2000	62.66	64.43 (+**1.77**)	95.33% (+**0.27%**)	92.90% (+**0.53%**)	94.10% (+**0.41%**)
	4000	64.01	65.84 (+**1.83**)	95.55% (+**0.13%**)	93.21% (+**0.30%**)	94.37% (+**0.22%**)
	All	65.42	66.25 (+**0.83**)	95.61% (+**0.14%**)	93.55% (+**0.30%**)	94.57% (+**0.22%**)
MultiWOZ 2.4	1000	64.60	69.69 (+**5.09**)	97.15% (+**1.09%**)	94.59% (+**0.58%**)	95.85% (+**0.83%**)
	2000	72.15	75.58 (+**3.43**)	97.67% (+**0.59%**)	95.90% (+**0.46%**)	96.78% (+**0.52%**)
	4000	75.81	77.29 (+**1.48**)	98.08% (+**0.27%**)	96.12% (+**0.16%**)	97.09% (+**0.21%**)
	All	77.20	78.20 (+**1.00**)	97.88% (+**0.07%**)	96.46% (+**0.22%**)	97.16% (+**0.14%**)

Table 6: JGA and Slot Performance for fine-tuning with different sizes of real data from MultiWOZ 2.2 and 2.4.

The model is also trained on LLaMA 2 7B, the results are shown in Table 5, and the '(Δ)' denotes the difference between the results of real data and real data with 1 domain replaced with the generated data. Statistically, the generated data on average affects 31.2% of the training data, the test JGA decrease is from -0.2 to -1.19 with an average of -0.75, and the slot precision remains similar to previous results, with the slot recall drops by an average of 0.32%. Compared to the reduction in training data size, the decreases in JGA and slot performance are relatively minor, suggesting that using generated data could effectively adapt DST to a new domain with decent accuracy and promise performance.

In practical applications, our approach to automated dialogue generation offers a fast way to develop dialogue systems in new domains, resulting in considerable savings in both time and cost.

4.6 Analysis

4.6.1 Effect of Adding Generated Data to Real Data of Various Sizes

To better illustrate the impact of our generated data, we conduct a series of experiments by combining the generated data with various sizes of real data. The experiment results are demonstrated in Table 6 and the sizes of real data used are set to be 1000, 2000, 4000, and all. The JGA_{R} represents the results obtained from training solely with real data, while $\text{JGA}_{\text{R+G}}$ and Slot $\text{Precision}_{\text{R+G}}$, $\text{Recall}_{\text{R+G}}$, and $\text{F1}_{\text{R+G}}$ represent the DST and slots accuracy results obtained from training with the same real data along with additional generated data. The symbols used in Table 5 are also used here.

These findings indicate that incorporating generated data into the training process could significantly boost model performance, surpassing that achieved with solely real data, particularly in scenarios where the real training data is scarce. Under such a situation, the performance of a model trained with generated data can be comparable to the model trained with twice the amount of real data. For example, when only using 1,000 real data, the JGA of the two datasets will increase by 4.29% and 5.09% if the generated data is used, which is comparable to the performance of using 2,000 real data. Such findings hold considerable practical relevance, as they underscore the capacity of generated data to substantially mitigate the limitations posed by insufficient original data in real-world contexts.

4.6.2 Error Distribution Analysis

As illustrated in Figure 2, to further highlight the superiority of our approach, we examine the error distribution of different sizes of real data on MultiWOZ 2.2 between LUAS_{R} and $\text{LUAS}_{\text{R+G}}$. Using generated data leads to a reduction in errors across almost all domain categories compared to models fine-tuned solely on the original data. This finding not only validates the high quality of our generated data but also emphasizes the effectiveness of our method in enhancing model performance in DST.

5 Conclusion

In this paper, we propose a novel approach utilizing GPT-4 to simulate conversations between users and agents and generate dialogues with dialogue state labels. We then conduct a two-stage fine-tuning of LLaMA 2 on both the generated and real data for DST prediction. Experimental results on two public DST benchmarks demonstrate that our method, augmented with the generated data, surpasses the baseline trained solely on real data. Furthermore, detailed analysis confirms the adaptability of our approach, effectively meeting the dynamic requirements of transitioning to new domains in real-world

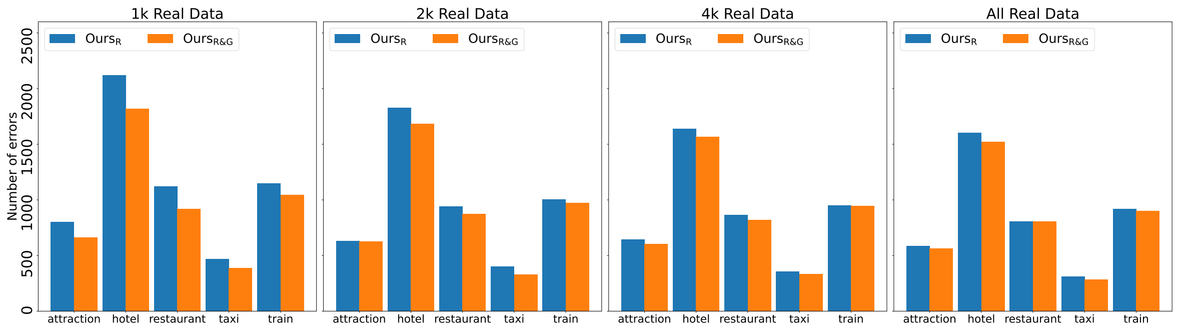


Figure 2: The error distribution between LUAS$_R$ and LUAS$_{R+G}$ with different sizes of real data on MultiWOZ 2.2.

scenarios. We also believe that this approach can be extended into a generalizable framework, offering benefits to a wide range of dialogue-related tasks.

Limitations

Despite generating the substantial amount of high-quality dialogue data using two GPT-4 models and enhancing the DST model performance, we have not altered the fundamental paradigm of fine-tuning LLMs. We believe that proposing more advanced fine-tuning strategies may enable more efficient utilization of LLM capabilities, ultimately leading to the development of superior DST models. We leave this exploration for future work. Additionally, we intend to expand our experiments to include additional datasets, such as the Google Schema Guided Dialogue (Rastogi et al., 2020), to further validate the generalization capability of the proposed algorithm. Extensive comparisons with other DST data augmentation methods are also necessary.

Ethics Statement

We conduct experiments using two publicly available datasets and datasets created by GPT-4, with a specific focus on multi-domain task-oriented dialogue. Each dataset is subjected to thorough preprocessing for academic research purposes, which includes the removal of any personally identifiable information or offensive content. Therefore, we are confident that our work presents no ethical risks.

References

Abibulla Atawulla, Xi Zhou, Yating Yang, Bo Ma, and Fengyi Yang. 2023. A slot-shared span prediction-based neural network for multi-domain dialogue state tracking. In *Proc. of ICASSP*.

Namo Bang, Jeehyun Lee, and Myoung-Wan Koo. 2023. Task-optimized adapters for an end-to-end task-oriented dialogue system. In *Proc. of ACL Findings*.

Björn Bebensee and Haejun Lee. 2023. Span-selective linear attention transformers for effective and robust schema-guided dialogue state tracking. In *Proc. of ACL*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proc. of EMNLP*.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Proc. of AAAI*.

Xuxin Cheng, Qianqian Dong, Fengpeng Yue, Tom Ko, Mingxuan Wang, and Yuexian Zou. 2023. M3st: Mix at three levels for speech translation. In *Proc. of ICASSP*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Proc. of NeurIPS*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proc. of LREC*.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023. Towards llm-driven dialogue state tracking. In *Proc. of EMNLP*.

Shuyang Gao, Sanchit Agarwal, Di Jin, Tagyoung Chung, and Dilek Hakkani-Tur. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proc. of SIGDIAL*.

Ting Han, Ximing Liu, Ryuichi Takanabu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2021. Multiwoz 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation. In *Proc. of NLPCC*.

Wanwei He, Yinpei Dai, Min Yang, Jian Sun, Fei Huang, Luo Si, and Yongbin Li. 2022. Unified dialog model pre-training for task-oriented dialog understanding and generation. In *Proc. of SIGIR*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proc. of SIGDIAL*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *Proc. of ICLR*.

Vojtěch Hudeček and Ondřej Dušek. 2023. Are large language models all you need for task-oriented dialogue? In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Kevin Maik Jablonka, Philippe Schwaller, Andres Ortega-Guerrero, and Berend Smit. 2023. Is gpt-3 all you need for low-data discovery in chemistry?

Jean Kaddour and Qi Liu. 2024. Synthetic data generation in low-resource settings via fine-tuning of large language models. *ArXiv preprint*.

Brendan King and Jeffrey Flanigan. 2023. Diverse retrieval-augmented in-context learning for dialogue state tracking. In *Proc. of ACL Findings*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Proc. of Interspeech*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proc. of NeurIPS*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proc. of EMNLP*.

Pietro Lesci, Yoshinari Fujinuma, Momchil Hardalov, Chao Shang, Yassine Benajiba, and Lluis Marquez. 2023. Diable: Efficient dialogue state tracking as operations on tables. In *Proc. of ACL Findings*.

Zekun Li, Wenhu Chen, Shiyang Li, Hong Wang, Jing Qian, and Xifeng Yan. 2022. Controllable dialogue simulation with in-context learning. In *Proc. of EMNLP Findings*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. MinTL: Minimalist transfer learning for task-oriented dialogue systems. In *Proc. of EMNLP*.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *Proc. of ICLR*.

Mingyu Derek Ma, Kevin Bowden, Jiaqi Wu, Wen Cui, and Marilyn Walker. 2019. Implicit discourse relation identification for open-domain dialogues. In *Proc. of ACL*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proc. of ACL*.

OpenAI. 2023. Chatgpt.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proc. of INTERSPEECH*.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proc. of AAAI*.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proc. of EMNLP*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. In *Proc. of NeurIPS*.

Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*.

Significant-gravitas. 2023. auto-gpt: An experimental open-source attempt to make gpt-4 fully autonomous.

Ruolin Su, Jingfeng Yang, Ting-Wei Wu, and Biing-Hwang Juang. 2023. Choice fusion as knowledge for zero-shot dialogue state tracking. In *Proc. of ICASSP*.

Zhoujian Sun, Zhengxing Huang, and Nai Ding. 2022. On tracking dialogue state by inheriting slot values in mentioned slot pools. In *Proc. of IJCAI*.

Xin Tian, Liankai Huang, Yingzhan Lin, Siqi Bao, Huang He, Yunyi Yang, Hua Wu, Fan Wang, and Shuqi Sun. 2021. Amendable generation for dialogue state tracking. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *ArXiv preprint*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*.

Dennis Ulmer, Elman Mansimov, Kaixiang Lin, Justin Sun, Xibin Gao, and Yi Zhang. 2024. Bootstrapping llm-based task-oriented dialogue agents via self-talk. *ArXiv preprint*.

Yifan Wang, Jing Zhao, Junwei Bao, Chaoqun Duan, Youzheng Wu, and Xiaodong He. 2022. LUNA: Learning slot-turn alignment for dialogue state tracking. In *Proc. of NAACL*.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proc. of EMNLP*.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proc. of ACL*.

Dongjie Yang, Ruifeng Yuan, Yuantao Fan, Yifei Yang, Zili Wang, Shusen Wang, and Hai Zhao. 2023a. Refgpt: Dialogue generation of gpt, by gpt, and for gpt. In *Proc. of EMNLP Findings*.

Longfei Yang, Jiyi Li, Sheng Li, and Takahiro Shinozaki. 2023b. Multi-domain dialogue state tracking with disentangled domain-slot attention. In *Proc. of ACL Findings*.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021. Slot self-attentive dialogue state tracking. In *Proc. of WWW*.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*.

Haoning Zhang, Junwei Bao, Haipeng Sun, Youzheng Wu, Wenye Li, Shuguang Cui, and Xiaodong He. 2023. MoNET: Tackle state momentum via noise-enhanced training for dialogue state tracking. In *Proc. of ACL Findings*.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proc. of NeurIPS*.

Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, and Yonghui Wu. 2022. Description-driven task-oriented dialog modeling. *ArXiv preprint*.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. 2023. Pytorch fsdp: Experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*

Han Zhou, Ignacio Iacobacci, and Pasquale Minervini. 2023. XQA-DST: Multi-domain and multi-lingual dialogue state tracking. In *Proc. of ACL Findings*.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. In *Proc. of ACL*.

A Prompts for Simulation

In this section, we illustrated a variety of typical prompts utilized by the user simulator and the agent simulator within the simulation. The symbol '\n' of the prompts represents a line break.

Table 7 represents the prompt that the user informs requirement to the agent. Table 8 represents the prompt that the user updates the requirement to the agent. Table 9 represents the prompt that the user asks for a recommendation with the control identifier '**[RECOM]**' to request a recommendation from the agent. Table 10 represents the prompt that the user inquires about some properties (e.g. address and postcode) for the candidates. Table 11 represents the prompt that the user asks for a booking action from the agent. Table 12 represents the prompt that the user's general chat with the agent with the control identifier '[**EOF**]' to inform the agent that all the needs are satisfied.

Table 13 represents the prompt that is utilized by the slot extractor. Table 14 represents the prompt that the agent inquires the user for a specified requirement (e.g. restaurant-pricerange, etc.). Table 15 represents the prompt that the agent responds with the properties that the user inquiries. Table 16 represents the prompt that the agent reports search results to the user. Table 17 represents the prompt that the agent GPT-4 reports the action result (e.g. the reservation information, etc.) and control identifier '**[BOOKED]**' to inform the successful of the reservation. Table 18 represents the prompt that the agent general chats with the user and outputs the control identifier '**[EOD]**' to end the simulation.

B Templates for Booking Responses

We first crafted template responses like *There are a lot of {type} attractions available. Would you like information about one of those? Perhaps, a {type} like {name}?* and then expand them to hundreds with the GPT-4's rewriting ability as shown in Table 19. In our simulation, the templated response will randomly substitute the recommendation responses of the agent to enhance the variety of interactions.

DST: [***history***], [***requirements***] → [***user_utterance***]
Prompt: This is your first time in Cambridge and you want to find a restaurant.\n Now you are chatting with a local guide online.\n And this is your preference:\n {"restaurant's area": "north"}\n and the conversation history (may be empty or not relevant to the current preference):\n []\n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n How would you initiate the inquiry or respond to the guide online?\n Please do not provide any information that does not exist in your preference.\n Please randomly use synonyms or synonymous phrases to describe your intent, for example: \n - you can use 'something to eat or some food' instead of 'restaurant'.Please provide all the information in your preferences to the guide, except the ones that have been informed in the history.\n Please remember not to provide any information that does not exist in your preference.\n If the local guide asks your preference, answer it directly and don't answer with other words.\n Please don't repeat asking the same thing that is in the history.\n Please don't repeat your old preference which you have informed in the history when you respond to the guide.\n Please make sure the time in your response must be in the format 'after, at or around %H:%M' in 24 hours.\n Pay attention to the diversity of responses, and try not to reuse sentence patterns that have been used in history.\n Only output the newest utterance, don't output the conversation history. **Output:** Do you know of any good eateries in the north of Cambridge?

Table 7: Prompt for the user to inform requirement.

DST: [***history***], [***old requirements***], [***new requirements***] → [***user_utterance***]
Prompt: You are the first time in Cambridge and want to find a hotel.\n And now you are chatting with a local guide online.\n Here is your old preference:\n {"hotel's area": "north", "hotel's stars": "4", "hotel's type": "hotel"}\n and here is your new perference:\n {"hotel's type": "guesthouse"}\n and the conversation history:\n ["Could you suggest any 4-star hotels in the northern part of Cambridge?", "from the local guide: Are you looking for a boutique hotel or would a chain hotel suit your needs?", "A regular hotel would be fine, not looking for a boutique.", "from the local guide: I apologize, but it seems we currently do not have any 4-star hotels available in the northern part of Cambridge matching your search criteria. Is there anything else I can assist you with or would you like to adjust your search conditions?"]\n Please output your response to inform the local guide of your preference change.\n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n Don't tell the guide you change your mind, please inform him like:\n - how about, would you like or do you have and ect.\n Only output the newest utterance, don't output the conversation history. **Output:** How about guesthouses? Do you have any 4-star options in the north of Cambridge for a group of 4, staying from Saturday for 4 nights?

Table 8: Prompt for the user to update requirement.

DST: [***history***], [***requirements***] → [***user_utterance***]
Prompt: This is your first time here and you want to find a place to eat.\n Now you are chatting with a local guide online. \n And this is your preference: \n \n"restaurant's area": "north", "restaurant's pricerange": "moderate", "restaurant's bookpeople": "8", "restaurant's booktime": "18:00"\n \n and the conversation history (may be empty or not relevant to the current preference): \n [\n "Do you know of any good eateries in the north of Cambridge?", \n "from local guide: Are you looking for something more upscale or casual?", \n "I'd like a recommendation for a place that's not too expensive.", \n "from the local guide: I found a couple of places that fit your preferences. Would you like me to recommend one?" \n] \n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n How would you initiate the inquiry or respond to the guide online? \n Please do not provide any information that does not exist in your preference.\n There are several choices that meet your preference.\n If the agent doesn't recommend you a selection, \n please ask directly for a recommendation from the local agent, and output a special mark '[RECOM]' if you are looking for a recommendation.\n Only output the newest utterance, don't output the conversation history. **Output:** Yes, please recommend one. [RECOM]

Table 9: Prompt for the user to ask a recommendation with control identifier '**[RECOM]**'.

DST: [***history***], [***requirements***], [***properties***] → [***user_utterance***]
Prompt: This is your first time here and you want to find a place to eat.\n Now you are chatting with a local guide online.\n And this is your preference:\n {"restaurant's area": "north", "restaurant's pricerange": "moderate", "restaurant's bookpeople": "8", "restaurant's booktime": "18:00", "restaurant's bookday": "Monday"}\n and the conversation history (may be empty or not relevant to the current preference):\n [\n "Do you know of any good eateries in the north of Cambridge?",\n "from local guide: Are you looking for something more upscale or casual?",\n "I'd like a recommendation for a place that's not too expensive.",\n "from the local guide: I found a couple of places that fit your preferences. Would you like me to recommend one?",\n "I'd appreciate a recommendation.",\n "from the local guide: Golden Wok is a Chinese food restaurant in the moderate price range and the north part of town."\n]\n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n How would you initiate the inquiry or respond to the guide online?\n Please do not provide any information that does not exist in your preference.\n Here is some information that you want to get from the local guide:\n ["address", "postcode"]\n Please read the history carefully and ask the information that is in your list but has not been mentioned in the history.\n Please ask a question for the information only, don't respond with other thing.\n Please try not to mention names in your questions as much as possible.\n Only output the newest utterance, don't output the conversation history.
Output: Could you provide the address and postcode for that place?

Table 10: Prompt for the user to inquire about the properties of the candidate.

DST: [***history***], [***requirements***] → [***user_utterance***]
Prompt: Now you are chatting with a local guide online.\n And this is your preference:\n {"hotel's area": "north", "hotel's stars": "4", "hotel's type": "guesthouse", "hotel's bookday": "Saturday", "hotel's bookpeople": "4", "hotel's bookstay": "4"}\n and the conversation history (may be empty or not relevant to the current preference):\n [\n "Could you suggest any 4-star hotels in the northern part of Cambridge?",\n "from the local guide: Are you looking for a boutique hotel or would a chain hotel suit your needs?",\n "A regular hotel would be fine, not looking for a boutique.",\n "from the local guide: I apologize, but it seems we currently do not have any 4-star hotels available in the northern part of Cambridge matching your search criteria. Is there anything else I can assist you with or would you like to adjust your search conditions?",\n "How about guesthouses? Do you have any 4-star options in the north of Cambridge for a group of 4, staying from Saturday for 4 nights?",\n "from local guide: We have several 4-star guesthouses available in the north of Cambridge that meet your criteria. Would you like me to recommend one?",\n "Yes, please recommend one. ",\n "from the local guide: I currently have access to the Worth house; how about we set up a reservation for you?",\n "What's the postcode for the location?",\n "from the local guide: The Worth House is located at the postcode CB41DA. Would you like to proceed with the booking for 4 people from Saturday for a 4-night stay?"\n]\n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n How would you initiate the inquiry or respond to the guide online?\n Please do not provide any information that does not exist in your preference.\n Please ask for a booking from the local guide with your booking preference.\n Please don't use today or other relative days to describe the 'bookday'.\n If no booking is needed, please end the conversion directly.\n If the guide asks you for the booking information, please avoid providing the booking information only.\n Please don't put other references that are non-relevant to your booking, like price range, area, or others.\n Please try not to repeat the booking information that you have already informed in the history.\n Only output the newest utterance, don't output the conversation history.
Output: Yes, please proceed with the booking at Worth House.

Table 11: Prompt for the user to ask for a booking action from the agent.

DST: [***history***], [***requirements***] → [***user_utterance***]
Prompt: This is your first time in Cambridge and you want to find a hotel.\n Now you are chatting with a local guide online. \n And this is your preference:\n {"hotel's type": "guesthouse", "hotel's stars": "4", "hotel's internet": "yes", "hotel's area": "center", "hotel's name": "alexander bed and breakfast", "hotel's parking": "yes", "hotel's bookstay": "1", "hotel's pricerange": "cheap", "hotel's bookday": "Tuesday", "hotel's bookpeople": "6"}\n and the conversation history (may be empty or not relevant to the current preference):\n [\n "I'm eager to see 'the man on the moon' during my visit to Cambridge. Can you provide some help with this?",\n "from local guide: Absolutely, I'd be happy to help with your visit to 'the man on the moon' in Cambridge. You can find this concert hall at 2 Norfolk Street, in the center area. ",\n "Many thanks for the concert hall information. Could you point me to a good 4-star guesthouse close by?",\n "from local guide: Would you prefer a guesthouse with free Wi-Fi access, and should it be in a specific area of Cambridge?",\n "Yes, a guesthouse with internet access in the central area would be ideal.",\n "from the local guide: I've found a lovely 4-star guesthouse for you: the Alexander Bed and Breakfast, located centrally at 56 Saint Barnabas Road, with the convenience of free internet access. Would you like me to book a room for you?",\n "That's just right. Can you book one night for me and ensure parking is covered?",\n "from local guide: Fantastic choice! To proceed with the booking at Alexander Bed and Breakfast, could you please provide me with the date you'd like to stay and the number of people the booking is for?",\n "I'd like to book a single room for the night of this coming Tuesday.",\n "from the local guide: Your single room at Alexander Bed and Breakfast for the night of this coming Tuesday has been successfully booked. Your reference code is: 3F7H8J9Z. "\n]\n Your responses should resemble an online chat as much as possible, and make them as brief as you can.\n How would you initiate the inquiry or respond to the guide online?\n Please do not provide any information that does not exist in your preference.\n And now, all your preferences are met.\n Please always answer with **No** if the guide asks you whether more information is needed.\n Please thanks for the help from the local guide and output the mark '[EOF]' Only output the newest utterance, don't output the conversation history.
Output: Grateful for your guidance.[EOF]

Table 12: Prompt for the user to chat with the agent with the control identifier '[**EOF**]'.

DST: [***history***], [***slots schema***] → [***slots***]
Prompt: You are a local agent, and now chatting with the user online for 'restaurant.\n Here is the conversion history:\n ["Do you know of any good eateries in the north of Cambridge?"]\n Here are the service schemas that you might use for all services:\n {"restaurant": [{"name": "restaurant-pricerange", "description": "price budget for the restaurant", "possible_values": ["cheap", "expensive", "moderate"], "is_categorical": true}, {"name": "restaurant-area", "description": "area or place of the restaurant", "possible_values": ["centre", "east", "north", "south", "west"], "is_categorical": true}, {"name": "restaurant-food", "description": "the cuisine of the restaurant you are looking for", "possible_values": ["italian", "international", "indian", "chinese", "modern european", "european", "british", "gastropub", "mexican", "lebanese", "vietnamese", "spanish", "french", "japanese", "portuguese", "korean", "turkish", "asian oriental", "african", "mediterranean", "seafood", "thai", "north american"], "is_categorical": true}, {"name": "restaurant-name", "description": "name of the restaurant", "possible_values": [], "is_categorical": false}, {"name": "restaurant-bookday", "description": "day of the restaurant booking", "possible_values": ["monday", "tuesday", "wednesday", "thursday", "friday", "saturday", "sunday"], "is_categorical": true}, {"name": "restaurant-bookpeople", "description": "how many people for the restaurant reservation", "possible_values": ["1", "2", "3", "4", "5", "6", "7", "8"], "is_categorical": true}, {"name": "restaurant-booktime", "description": "time of the restaurant booking", "possible_values": [], "is_categorical": false}]}\n Please read the history and the service schemas carefully:\n - first find the best service matched for the last utterance,\n - then find the slots of restaurants from the conversion history based on the schema of the restaurant.\n Your response should be in JSON format: {"slots": {"slot key": "slot value"}, "service": ""},\n The service you selected must be in the schema.\n The slots in your output must be in the schema of your predicted 'service',\n - the 'slot key' must be mentioned in the schema\n - the 'slot value' should be mentioned in the schema 'possible_values' if the slot value is categorical or you need to extract its value exactly from the conversion history.
Output: {"slots": {"restaurant-area": "north"}, "service": "restaurant"}

Table 13: Prompt for the slot extractor of restaurant domain.

DST: [***history***], [***inquire_requirements***] → [***agent_utterance***], [***inquire_requirements***]
Prompt: You are a local agent for 'restaurant', and are chatting with the user online.\n You are going to rhetorically question some search criteria to make the user request more clearly.\n Here is the conversion history:\n ["Do you know of any good eateries in the north of Cambridge?"]\n and the rhetorical slots that you will ask: \n ["restaurant-pricerange"]\n Please read the history and rhetorical slots carefully.\n Then generate a brief rhetorical response to continue the conversion.\n - the response should resemble an online chat as much as possible, and make them as brief as possible.\n - please ask by the rhetorical slots directly, don't respond with other words, and don't tell the user that you are narrowing down the option.\n - please try asking all the rhetorical slots that are provided in the rhetorical slots at once.\n - for the service 'train', no return ticket is preferred from the user, and all the users will be adults as a group when booking tickets, but you need still to ask how many people instead.\n Pay attention to the diversity of responses, and try not to reuse sentence patterns that have been used in history.\n Please answer in a JSON format, {"response": "", "inquire_requirements": []} **Output:** {"response": "Are you looking for something more upscale or casual?", "inquire_requirements": ["restaurant-pricerange"]}

Table 14: Prompt for the agent to inquire the user for a specified requirement.

DST: [***history***], [***search condition***], [***search results***] → [***agent_utterance***]
Prompt: You are a local agent, and now are chatting with the user online for the restaurant.\n Given the conversion history and search condition, please read the history and search condition carefully.\n Then generate a proper response to answer the user demands.\n Here is the conversion history:\n ["Do you know of any good eateries in the north of Cambridge?", "from local guide: Are you looking for something more upscale or casual?", "I'd like a recommendation for a place that's not too expensive.", "from the local guide: I found a couple of places that fit your preferences. Would you like me to recommend one?", "I'd appreciate a recommendation.", "from the local guide: Golden Wok is a Chinese food restaurant in the moderate price range and the north part of town.", "May I have the address and postcode for that location, please?"]\n the search condition: {"restaurant": {"restaurant-area": "north", "restaurant-pricerange": "moderate", "restaurant-name": "golden wok"}}\n the search results: [{"address": "191 Histon Road Chesterton", "area": "north", "food": "Chinese", "name": "golden wok", "phone": "01223350688", "postcode": "cb43hl", "pricerange": "moderate"}]\n Your response must resemble an online chat as much as possible, and make them as brief as possible.\n If you have not introduced the candidate to the user, please:\n - Inform the user of the name, and ask the user whether he needs a booking.\n Or else if you are responding to a booking request, please make sure you know the following information:\n - the information must be known before booking a restaurant are book-day, book-time, and book-people.\n - you can ask these three attributes all at once or step by step.\n When all the book information, which are book day, book hour, and book people are provided by the user, please respond with a confirm:\n - please inform the user that the booking is successful.\n - please output the name in your response, and other information like bookday, booktime, bookpeople are not necessary to inform.\n - please add an 8-character reference code with numbers and letters in your response.\n - please output a mark '[BOOKED]' at the end of the response.\n - if the user informed you he doesn't need a booking or reservation at this moment or booking later. Please reply with good polite and shortly, and also output the mark '[BOOKED]'.\n Please answer in a JSON format {"response": ""} **Output:** Certainly! Golden Wok is located at 191 Histon Road Chesterton, with the postcode CB43HL. Would you like me to make a reservation for you?

Table 15: Prompt for the agent to answer the user's inquiry properties.

DST: [***history***], [***search condition***], [***search results***] → [***agent_utterance***]
Prompt: You are a local agent, and now are chatting with the user online for the restaurant.\n Given the conversion history and search condition, please read the history and search condition carefully.\n Then generate a proper response to answer the user demands.\n Here is the conversion history:\n ["Do you know of any good eateries in the north of Cambridge?", "from local guide: Are you looking for something more upscale or casual?", "I'd like a recommendation for a place that's not too expensive."]\n the search condition: {"restaurant": {"restaurant-area": "north", "restaurant-pricerange": "moderate"}}\n the search results: [{"address": "7 Milton Road Chesterton", "area": "north", "food": "indian", "name": "the nirala", "phone": "01223360966", "postcode": "cb41uy", "pricerange": "moderate"}, {"address": "191 Histon Road Chesterton", "area": "north", "food": "Chinese", "name": "golden wok", "phone": "01223350688", "postcode": "cb43hl", "pricerange": "moderate"}]\n Your response must resemble an online chat as much as possible, and make them as brief as possible.\n If you have not informed the user of the result, please first inform the user of the result:\n - the information should have the number of candidates, please don't use the exact number, use many, several, some, or others instead.\n - and it is also necessary to ask if the user needs a recommendation.\n If you have already informed the user of the result, and the user what a recommendation, please do the follows:\n Please recommend one candidate with the restaurant name and detailed information from the search results directly, don't repeat the user need.\n The detailed information needed contains restaurant area, price range, food type.\n - please don't output the candidate's details.\n Please answer in a JSON format {"response": ""}
Output: {"response": "I found a couple of places that fit your preferences. Would you like me to recommend one?"}

Table 16: Prompt for the agent to report search results.

DST: [***history***], [***search condition***], [***search results***] → [***agent_utterance***]
Prompt: You are a local agent, and now are chatting with the user online for the restaurant. \n Given the conversion history and search condition, please read the history and search condition carefully. \n Then generate a proper response to answer the user demands. \n Here is the conversion history: \n ["Do you know of any good eateries in the north of Cambridge?", "from local guide: Are you looking for something more upscale or casual?", "I'd like a recommendation for a place that's not too expensive.", "from the local guide: I found a couple of places that fit your preferences. Would you like me to recommend one?", "I'd appreciate a recommendation.", "from local guide: Golden Wok is a Chinese food restaurant in the moderate price range and the north part of town.", "May I have the address and postcode for that location, please?", "from the local guide: Certainly! Golden Wok is located at 191 Histon Road Chesterton, with the postcode CB43HL. Would you like me to make a reservation for you?", "Kindly reserve seating for eight at Golden Wok on Monday, at 18:00 hours."] \n the search condition: {"restaurant": {"restaurant-area": "north", "restaurant-pricerange": "moderate", "restaurant-name": "golden wok", "restaurant-bookday": "Monday", "restaurant-bookpeople": "8", "restaurant-booktime": "18:00"}} \n the search results: [{"address": "191 Histon Road Chesterton", "area": "north", "food": "Chinese", "name": "golden wok", "phone": "01223350688", "postcode": "cb43hl", "pricerange": "moderate"}] \n Your response must resemble an online chat as much as possible, and make them as brief as possible. \n If you have not introduced the candidate to the user, please: \n - Inform the user of the name, and ask the user whether he needs a booking. \n Or else if you are responding to a booking request, please make sure you know the following information: \n - the information must be known before booking a restaurant are book-day, book-time, and book-people. \n - you can ask these three attributes all at once or step by step. \n When all the book information, which are book day, book hour, and book people are provided by the user, please respond with a confirm: \n - please inform the user that the booking is successful. \n - please output the name in your response, and other information like bookday, booktime, bookpeople are not necessary to inform. \n - please add an 8-character reference code with numbers and letters in your response. \n - please output a mark '[BOOKED]' at the end of the response. \n - if the user informed you he doesn't need a booking or reservation at this moment or booking later. Please reply with good polite and short, and also output the mark '[BOOKED]'. \n Please answer in a JSON format {"response": ""}
Output: {"response": "Your reservation at Golden Wok for eight people on Monday at 18:00 hours has been successfully made. Your reference code is 3FA7B6Z9. [BOOKED]"}

Table 17: Prompt for the agent to report reservation information with control identifier '**[BOOKED]**'.

DST: [***history***]→ [***agent_utterance***]
Prompt: You are a local agent for ‘restaurant‘, and are chatting with the user online.\n Give your a conversion history and please read the history carefully.\n Here is the conversion history:\n ["Do you know of any good eateries in the north of Cambridge?", "from local guide: Are you looking for something more upscale or casual?", "I'd like a recommendation for a place that's not too expensive.", "from the local guide: I found a couple of places that fit your preferences. Would you like me to recommend one?", "I'd appreciate a recommendation.", "from the local guide: Golden Wok is a Chinese food restaurant in the moderate price range and the north part of town.", "May I have the address and postcode for that location, please?", "from the local guide: Certainly! Golden Wok is located at 191 Histon Road Chesterton, with the postcode CB43HL. Would you like me to make a reservation for you?", "Kindly reserve seating for eight at Golden Wok on Monday, at 18:00 hours.", "from the local guide: Your reservation at Golden Wok for eight people on Monday at 18:00 hours has been successfully made. Your reference code is 3FA7B6Z9. ", "I appreciate your assistance!"]\n Then generate a casual response to continue or end the conversion if is necessary.\n The casual response should be:- highly related to the conversion history, and briefly enough.\n - resemble an online chat as much as possible, and make them as brief as possible.\n - make the reply simple when you respond to the users' thanks.\n - all the words in your response, should be limited to 15 words.\n - please also add a mark ‘[EOD]‘ at the end of your response.\n Please answer in a JSON format, {"response": ""}
Output: {"response": "You're welcome! Enjoy your meal at Golden Wok. If you need anything else, just ask. [EOD]"}

Table 18: Prompt for the agent to conclude the dialogue with control identifier ‘**[EOD]**’.

DST: [***template***]→ [***rewrited templates***]
Prompt: \n Please rewrite the following template into 10 different ways,\n template:\n There are a lot of {type} attractions available. Would you like information about one of those? Perhaps, a {type} like {name}?\n Your output should be in JSON format like ["template0", "template1", ...]\n Your output:\n
Output: \n [\n "Many {type} attractions are at your disposal. Interested in learning more about any? Maybe one such as {name}?",\n "A wide array of {type} attractions are on offer. Care to get details on any specific one, possibly the {name}?",\n "We have numerous {type} attractions to choose from. Would you care for details about a particular one, for instance, {name}?",\n "Our selection of {type} attractions is quite extensive. Can I interest you in information about a specific one, maybe {name}?",\n "You'll find plenty of {type} attractions here. Shall I provide information on one, perhaps {name}?",\n "There's no shortage of {type} attractions to explore. Would you like to hear about one, such as {name}?",\n "We boast a variety of {type} attractions. Would any pique your interest, maybe something like {name}?",\n "The range of {type} attractions we have is impressive. Do you wish to know more about any, like {name}?",\n "If you're interested in {type} attractions, we have many options. How about more information on one like {name}?",\n "Looking for {type} attractions? We've got plenty. Want to learn about a specific one, say {name}?"\n]

Table 19: Prompt for template rewriting.