Plan, Generate and Complicate: Improving Low-resource Dialogue State Tracking via Easy-to-Difficult Zero-shot Data Augmentation

Ming Gu and **Yan Yang***
School of Computer Science and Technology, East China Normal University
51215901012@stu.ecnu.edu.cn, yanyang@cs.ecnu.edu.cn

Abstract

Data augmentation methods have been a promising direction to improve the performance of small models for low-resource dialogue state tracking. However, traditional methods rely on pre-defined user goals and neglect the importance of data complexity in this task. In this paper, we propose **EDZ-DA**, an **E**asy-to-**D**ifficult **Z**ero-shot **D**ata **A**ugmentation framework for low-resource dialogue state tracking that utilizes large language models to automatically catch the relationships of different domains and then generate the dialogue data. We also complicate the dialogues based on the domain relation to enhance the model's capability for co-reference slot tracking. Furthermore, we permute slot values to mitigate the influence of output orders and the problem of incomplete value generation. Experimental results illustrate the superiority of our proposed method compared to previous strong data augmentation baselines on MultiWOZ.[1]

1 Introduction

The data scarcity challenge in dialogue state tracking (DST) is significant due to the incessant emergence of new domains in task-oriented dialogue systems (ToDs) and the high costs associated with data annotation. Currently, large language models (LLMs) like ChatGPT have shown promising results in zero-shot DST(Heck et al., 2023). However, although these models achieve superb performance, they have significant limitations such as closed source, request limitations, and deployment difficulties(Feng et al., 2023). Therefore, a smaller, fine-tuned model is a more practical and cost-efficient choice for DST. Nevertheless, developing such a powerful small model faces a big challenge in the absence of training data.

Recently, strategies of data augmentation with the help of LLM's strong capability of instruction following and generation is a promising direction for enhancing a task-specific model. However, how to augment the DST data with LLMs is still underexplored. For training a powerful DST model, the logicality and naturalness of the dialogue are important, as well as the data complexity.

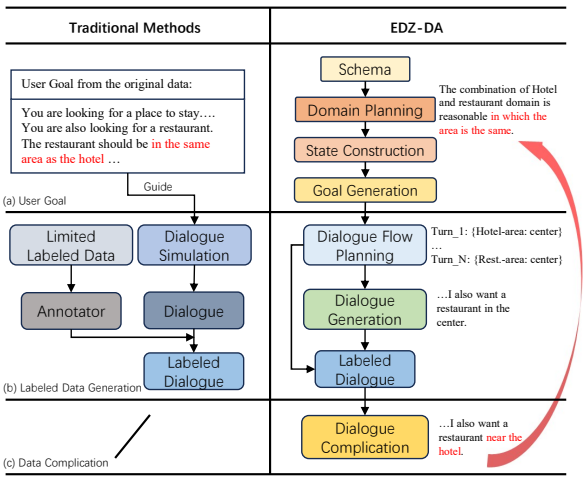

Figure 1: Differences between traditional methods and our method.

We investigate the process of data collection and find that constructing such an annotated dialogue dataset has three major issues: (i) while constructing a dialogue, the user goal is the most important since it guides the whole dialogue construction. Traditional data augmentation methods (Kim et al., 2021; Mohapatra et al., 2021; Wan et al., 2022) directly employ user goals from the original datasets or template-based goals. However, constructing diverse user goals is not that easy. As shown in Figure 1(a), the user goal not only contains all the domain-slots information but also the logical relationship among different domains within a dialogue. We propose to first plan the possible domain combination and then generate the user goal based on the synthetic dialogue state. (ii) annotation accuracy plays an important role in training a DST model.

*Corresponding Author
[1] https://github.com/SLEEPWALKERG/EDZ-DA

Findings of the Association for Computational Linguistics: ACL 2024, pages 6988–7005

Traditional methods train a model with limited data to annotate the synthesized dialogue. However, due to the limitation of data, the annotator is not satisfying. We propose to plan the dialogue flow first, where the dialogue flow is in the form of the turn state, and then generate the turn utterances based on the turn states. What's more, we propose to first instruct the LLM to generate dialogues with all slot values explicitly appearing in the utterance to alleviate the risk of hallucinations. (iii) complex data is inherently challenging for small models. In multi-domain dialogue state tracking, there are crucial challenges of co-reference, where slot values are sometimes expressed indirectly and should be inferred from the dialogue history. Greater attention should be devoted to this kind of data in order to further enhance the model's ability to handle these challenging samples. However, traditional methods neglect the data complexity problem. We find that the co-reference information like "restaurant-area" shares the same value as "hotel-area" is the direct expression of the domain relationship as shown in Figure 1(c). So, we propose to complicate the dialogue based on the generated logical relationship among domains.

Moreover, for generative information extraction models, the order of the output can exert influence on the model's training(Ye et al., 2021; Glazkova and Morozov, 2023; Luo et al., 2021). For example in value-based DST(Gu et al., 2024), which concatenates multiple slot values in a pre-defined order as the target output during training. Imposing a pre-defined order will result in wrong bias in the training process. And this problem becomes more serious under low-resource settings(Vinyals et al., 2016). We propose to permute slot values to mitigate the influence of the output order. Additionally, samples containing several slot values are inherently difficult samples for value generation. The augmentation with permutation can also enhance the model's capability to generate complete slot values within a dialogue turn.

In this paper, we propose **EDZ-DA**, an **E**asy-to-**D**ifficult **Z**ero-shot **D**ata **A**ugmentation framework for low-resource DST, which leverages the LLM's powerful reasoning ability on dialogue planning and then generate and complicate the dialogue. Specifically, we first propose to automatically catch the logical relationship among different domains with the help of the strong reasoning ability of LLMs and then generate the user goal. Second, we propose to first prompt an LLM to plan the dialogue flow, which contains the turn state annotation, and then generate the corresponding dialogue contents based on the flow, aiming at accurate dialogue generation with annotation matched. Third, we devise to complicate the synthetic dialogues based on the co-reference information to make conversations closer to real scenes and further improve the state tracker's capability of catching co-reference slots' values. Finally, we also propose to permute slot values to not only mitigate the influence of output orders but also reduce the incomplete generation phenomenon in value generation. Experimental results show that our method outperforms previous data augmentation methods and significantly improves the model's ability for co-reference slots tracking, demonstrating the superiority of our proposed method.

The contributions of this paper are summarized as the following:

- We propose EDZ-DA, an effective and generalizable LLM-based easy-to-difficult zero-shot data augmentation framework for low-resource DST.
- We propose to plan both the domain relationships and the dialogue flow for natural and accurate labeled dialogue construction. We also devise to complicate dialogues to further enhance model's capability to track co-reference slots.
- We propose to permute slot values to mitigate both the influence of the output order and the risk of incomplete generation in the value-based DST model.
- Experimental results show that our method achieves new SOTA performance.

2 Methodology

Figure 2 illustrates the process of our data augmentation. First, we prompt the LLM to judge whether it is reasonable for different combinations of domains to appear in one dialogue, and then generate the seed state, where the seed state describes the domains and co-reference information within a dialogue. Second, we synthesize diverse dialogue states based on the seed states. Finally, a series of tasks are proposed to generate the labeled dialogue for each synthetic dialogue state and then complicate them based on the co-reference information.

All prompt templates used in our framework are described in appendix A. Tables 7, 8, 9, 10, 11,

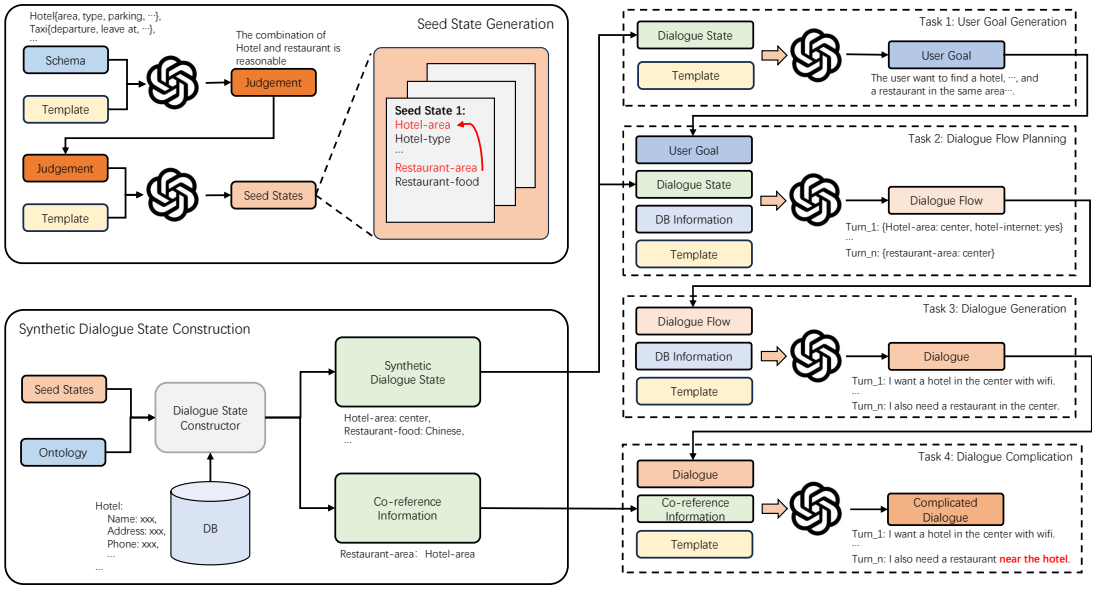

Figure 2: The overview of our proposed data augmentation method.

and 12 give an example of our dialogue generation method.

2.1 Dialogue State Construction

In this section, we introduce how we construct the synthetic dialogue state.

2.1.1 Seed State Generation

For multi-domain task-oriented dialogue, the logicality of the combination of domains is very important, we divide the seed state generation process into two steps: (1) domain judgment and (2) seed state generation. We carefully construct a manual prompt to instruct the LLM to judge whether the combination of domains is logical and reasonable and give some explanation. The MultiWOZ dataset includes five domains, and an analysis of the limited training set reveals that the majority of dialogues encompass one, two, or three domains. Consequently, we have extracted all combinations of two and three domains from the set of five and prompt the LLM to judge the possibility of these combinations of domains within a dialogue.

Second, we prompt the model to generate several seed states based on the explanation of the judgment and give the co-reference information. The seed state is in forms of a set of domain-slot, value pairs, and the co-reference information is included in it. As shown in figure 2, GPT determines that it is possible for hotel and restaurant domains to appear in the same dialogue and "restaurant-area" to share the same value with "hotel-area", which means that the user wants to find a restaurant in the same area as the booked hotel. Since the hallucination problem in LLMs, we carefully construct some rules based on logicality to filter out noisy seed states. For example in Table 8, "taxi-leaveat: restaurant-book time" in the first seed state is impossible in practice, so we remove it.

2.1.2 Synthetic Dialogue State Construction

After obtaining the seed states, we should fill in the blank values in them. For each seed state, we first adopt topological sort to gain the order of domains and then randomly select some places such as restaurants and hotels from the database (DB) according to the domain order. Some of slot values may share the same value with the former domain. So, when encountering these domains, we add constraints while searching the database. The process ends until all values are filled in the seed state. Repeat the aforementioned processes, and we will get several corresponding synthetic dialogue states based on one seed state.

2.2 Labeled Dialogue Generation

In this section, we describe how we construct the DST data based on the synthetic dialogue states.

2.2.1 User Goal Generation

Although the synthetic dialogue state summarizes the whole dialogue, generating the dialogue based only on the dialogue state suffers from role confusion in that the LLM will assume that the agent already known the requests of the user. Therefore,

before generating the dialogue, we first prompt the model to generate the user goal based on the synthetic dialogue state to guide the consequent dialogue flow generation. As shown in Table 9, we add constraints to ensure the user goal contains only the needs and the named entities like restaurant-names should be recommended by the agent.

2.2.2 Dialogue Flow Planning

To correctly generate the dialogue and the corresponding dialogue state annotation, we first prompt the model to plan the flow of a dialogue based on the user goal, the synthetic dialogue state, and the corresponding DB information. The flow consists of a list of {'description': <description for the user/agent's utterance>, 'turn state': <turn state mentioned in the utterance>} as shown in Table 10, where the turn state is in the form of a set of domain-slot, value pairs that constraints the content of the current turn. Moreover, we add further information about a certain place from the database and prompt the model to plan some turns to ask for additional information like phone numbers for more natural dialogue generation.

2.2.3 Easy-to-Difficult Dialogue Generation

Based on the dialogue flow and the dialogue state, we start to generate the dialogue. The most important thing in dialogue generation is consistency with the dialogue flow because the annotation is the turn label in the dialogue flow. Moreover, we also want the model to generate diverse utterances, especially when encountering co-reference slots. It is difficult to meet the above two needs simultaneously. So, we propose to first generate the dialogue strictly following the dialogue flow and express all slot values in an explicit way. Then we complicate the dialogue turns containing co-reference slots based on co-reference information in the seed state. Table 11 and Table 12 show an example of dialogue generation and dialogue complication, respectively.

2.3 Slot Value Permutation

We employ a permutation-based approach to mitigate the influence of sequence order on slot value generation. Table 13 shows an example. Specifically, we permute the set of slot values within each training example, including every permutation as a distinct training sample. For instance, if the current set of state values is {"A", "B"}. the output for the original training example would be "A | B". After permutation, two training samples are generated, one being "A | B" and the other "B | A". This method not only alleviates the impact of output order on the model but also serves as a form of data augmentation. The concurrent generation of multiple state values is one of the inherent challenges in state value generation. The permutation approach significantly amplifies the proportion of such samples within the dataset.

3 Experiments

3.1 Datasets and Metrics

Datasets We conduct our experiments on the MultiWOZ 2.1 dataset (Eric et al., 2020). It is a multi-domain task-oriented dialogue dataset which contains 8438 dialogues for training, 1000 dialogues for validating, and 1000 dialogues for testing. Following existing work (Wu et al., 2019), only five domains (restaurant, hotel, attraction, taxi, train) are used in our experiments because the other two domains have very few dialogues and only appear in the training set. We also test our results in MultiWOZ 2.3(Han et al., 2021) and MultiWOZ 2.4(Ye et al., 2022). MultiWOZ 2.3 provides the co-reference annotation and MultiWOZ 2.4 is an updated version upon MultiWOZ 2.1, which is the cleanest version of MultiWOZ for testing at the time of writing[2].

Metrics The standard metric (Wu et al., 2019), joint goal accuracy (JGA) is used in our experiments. This metric compares the whole predicted belief state to the gold one at each dialogue turn. If and only if all the predicted states match the ground truth states exactly for all domains, the prediction is treated as correct. In addition, we adopt co-reference slot accuracy to evaluate the model's capability for tracking co-reference slots.

3.2 Experimental Settings

We employ the GPT-4 Turbo model available in OpenAI API[3] to synthesize all the data. In terms of parameter configuration, a temperature of 0.7 has been set for dialogue generation, aiming at generating more diverse outputs. While for other modules, the temperature has been set to 0. The top-p parameter was uniformly set to 1 for all experiments.

For the dialogue state tracking model, we use SVAG (Gu et al., 2024), a SOTA small model for low-resource DST, which first generates all slot

[2]We do not use the validation set of MultiWOZ 2.4 for validating

[3]https://openai.com

values in the turn utterances and then generates the corresponding domain-slot type for each generated value. We exclude the self-training strategy in SVAG and directly adopt the experimental settings from Gu et al. (2024). The base models for both slot value generation and domain-slot generation are T5(Raffel et al., 2020), which contains about 770M parameters. Following Wu et al. (2019), we randomly sampled 1% and 5% of the data to simulate the low-resource scenarios with different seeds. We use the same data selection seeds as provided in (Gu et al., 2024), which are 10, 20, and 48.

3.3 Baseline Models

We compare our proposed method with several strong baseline data augmentation methods for low-resource DST.

NeuralWOZ(Kim et al., 2021) synthesizes annotated dialogues with a collector and a labeler. The collector generates a dialogue by using the given goal instruction and candidate relevant API call results from the KB. The labeler annotated the generated dialogue by reformulating it as a multi-choice problem. The augmented data of NeuralWOZ is publicly available and we sample the same number of dialogues from it for training.

Simulated Chats(Mohapatra et al., 2021) proposes to generate dialogues by simulating the interaction between crowd workers with a user bot and an agent bot. To generate the belief state, they also train a belief state generator. The authors did not provide the augmented data but the code. We reproduce their method and also sample the same number of dialogues from the generated data for training.

3.4 Main Results

We randomly select 1% and 5% data from the training set to simulate the low-resource scenarios with three different seeds to conduct our experiments and we report the averaged JGA score and co-reference slot accuracy over three runs. Table 1 shows the joint goal accuracy of the SVAG model on the MultiWOZ 2.1 and 2.4 test set when subjected to our data augmentation method and other baselines under different data ratio settings. Our method achieves SOTA performance compared to previous augmentation approaches.

Since the permutation of slot values is a general enhancement for generative extraction approaches like SVAG, we also present the results using only the dialogue data generated by our approach ("Ours w/o P"). We observe that our method engenders a more pronounced improvement under the data ratio setting of 1%, where it surges ahead of the NeuralWOZ augmentation approach by 4.15 in joint goal accuracy on the MultiWOZ 2.4 test set. Moreover, similar performance has been achieved by the two methods under the data ratio setting of 5%. And under both data ratio settings, EDZ-DA achieves better performance than simulated chats. Different from these two baselines, our approach does not rely on pre-defined user goals from the original dataset or manually constructed goal templates. Instead, our method automatically identifies the relationships among different domains and then generate user goals, providing a more general solution for constructing ToD data. And the better performance reveals both the logicality and accuracy of our proposed planning process and the effectiveness of our proposed labeled dialogue generation method. In particular, we find that Simulated Chats do harm to the model's performance when 5% data is available. Simulated Chats relies on the fine-tuning process on the limited data. So, the performance of their methods is limited in low-resource scenarios. Our method first plans the dialogue flow which contains the annotation and then generates the dialogue based on it, leading to more accurate labeled data generation.

Table 2 shows the co-reference slot accuracy by the SVAG model when enhanced with our data augmentation technique and other baselines under different data ratio settings. Our method achieves the highest increase among the two baseline models. Note that our method brings a 200% improvement in co-reference slot accuracy under the data ratio setting of 1%, which demonstrates the efficiency of our proposed easy-to-difficult dialogue generation for enhancing the model's capability to track co-reference slots. Under the data ratio setting of both 1% and 5%, our method achieves a measurable improvement in co-reference slot accuracy than NeuralWOZ. NeuralWOZ also brings benefits for co-reference slot tracking when only 1% of original data is available, but the improvement is very limited. For fine-tuning a powerful small model, complex data is very important since these data are even scarcer in extremely low-resource scenarios. It can be observed that both NeuralWOZ and simulated chats engender adverse effects on SVAG when dealing with co-reference slots under the data ratio setting of 5%, which not only demonstrates the importance of data complexity for DST but

Augmentation method	Pre-defined user goal	MultiWOZ 2.1		MultiWOZ 2.4	
		1	5	1	5
None	-	31.94	43.54	35.53	50.12
NeuralWOZ	Manual templates	34.43	43.51	37.64	51.04
Simulated Chats	Original data	-	41.09	-	47.29
Ours w/o P	-	36.49	43.3	41.79	50.95
Ours	-	**37.26**	**44.98**	**43.82**	**54.09**

Table 1: Joint goal accuracy of the SVAG models trained on different augmented data. "None" means that only limited original dataset is used for training. P refers to slot value permutation. The average results of 3 runs are reported and best results are marked bold.

Augmentation method	Data ratio	
	1	5
None	30.35	65.63
NeuralWOZ	35.54	61.87
Simulated Chats	-	54.73
Ours w/o P	64.07	71.08
Ours	**64.72**	**73.41**

Table 2: Co-reference slot accuracy on MultiWOZ 2.3 of the SVAG models trained on different augmented data.

Model	Param. size	Data ratio	
		1	5
DS2	<1B	36.76	49.89
DS2 + Ours		38.99	51.54
SVAG		35.53	50.12
SVAG + Ours		**43.82**	**54.09**
SM2-3B	<100B	37.59	49.22
SM2-11B		40.03	51.14
LDST*		46.77	56.48
IC-DST	>100B	48.35	55.43

Table 3: Joint goal accuracy compared with several strong models for low-resource DST on MultiWOZ 2.4. Bolded numbers indicate best performance on models under 1 billion parameters. *: LDST is trained on MultiWOZ 2.2.

also proves that our method can identify logical relations among domains and generate correct complex data to simulate real conversation and further enhance model performance.

Furthermore, we compare two small models enhanced by our method with other strong DST models containing more than 1 billion parameters. Table 3 summarizes the results. We conduct experiments on DS2(Shin et al., 2022) using the same data selection seeds provided in the original paper and observe that our augmentation data can further improve its performance under the data ratio settings of 1% and 5%, which demonstrates the quality of our annotated dialogue data. Compared to models with more than 1 billion parameters, it can be observed that SVAG enhanced by our augmented data surpasses SM2(Chen et al., 2023) by a margin of 3.79 and 2.95 in JGA under the data ratio settings of 1% and 5%, respectively. LDST(Feng et al., 2023) shows better performance than the enhanced SVAG. However, they use MultiWOZ 2.2 for training and evaluate the results on MultiWOZ 2.4. MultiWOZ 2.2 is a cleaner version than MultiWOZ 2.1. Over 17% of the annotation in MultiWOZ 2.1 is corrected in 2.2. So, it is not comparable. What's more, the enhanced SVAG model achieves competitive performance with IC-DST(Hu et al., 2022). IC-DST is based on CodeX, which contains more than 100 billion parameters. In summary, our proposed data augmentation method can significantly improve small models' performance in low-resource DST, reaching even better performance than the models ten times larger. Furthermore, since SVAG achieves better performance with the augmented data, it will make the consequent self-training more effective and further improve the performance.

3.5 Ablation Study

We conduct an ablation study to identify the contribution of different components from our proposed

Aug.	MultiWOZ 2.1		MultiWOZ 2.4	
method	1	5	1	5
EDZ-DA	37.26	**44.98**	**43.82**	**54.09**
-DG	34.39	43.76	39.11	52.15
-P	36.49	43.3	41.79	50.95
-P-Comp.	**37.45**	43.87	43.71	50.89

Table 4: Ablation study on joint goal accuracy. Comp. refers to data complication and P refers to slot value permutation.

Aug. method	Data ratio	
	1	5
EDZ-DA	**64.72**	**73.41**
-DG	47.21	70.56
-P	64.07	71.08
-P-Comp.	45.65	63.16

Table 5: Ablation study on co-reference slot accuracy on MultiWOZ 2.3.

augmentation method. Table 4 shows the joint goal accuracy score tested on MultiWOZ 2.1 & 2.4 when trained with different versions of our augmented data and Table 5 shows the co-reference slot accuracy tested on MultiWOZ 2.3.

First, we eliminate the generated dialogue data by the LLM, denoted as "-DG". We observe that both the joint goal accuracy and co-reference slot accuracy drop a lot under all data ratio settings. Notably, under the data ratio setting of 1%, removing the generated dialogue data dramatically harms the model performance, leading to a 4.71 decrease in JGA tested on MultiWOZ 2.4 and a 17.51 decrease in co-reference slot accuracy. The results indicate the effectiveness of our proposed method for DST data generation.

Second, we examine the use of slot value permutation. The results without slot value permutation ("-P") show that removing slot value permutation leads to a decrease in both JGA and co-reference slot accuracy under all data ratio settings, demonstrating the effectiveness of slot value permutation. Slot value permutation can not only mitigate the influence of output orders but also reduce the risk of incomplete generation, leading to better dialogue state tracking performance. Notably, as depicted in Table 5, the co-reference slot accuracy decreases more while removing the generated data, compared to the impact of removing slot value permutation, which further proves the significance of our proposed easy-to-difficult dialogue generation method for co-reference slot tracking.

Dialogue history: ... [user] I would like to go to thanh binh. [sys] Excellent. For how many would you like a reservation and at which preferred date and time? [user] I would like the reservation to be for 4 people on thursday at 18:30, please.
Current Turn Utterances: [sys] you're all set! Your reference number is: xxx. [user] I would also like a taxi to get me to the restaurant by that time.
Without augmentation: None
Augmented with NeuralWOZ: taxi-destination: thanh binh, restaurant-book time: 18:30
Augmented with ours: taxi-destination: thanh binh, **taxi-arriveby: 18:30**
Ground truth: taxi-destination: thanh binh, taxi-arriveby: 18:30

Dialogue history: [user] Hello, I am looking for a train that arrives by 16:00 and leaves on Monday. ...[sys] The tr0796 train leaves cambridge at 05:01 and arrives in broxbourne 06:01. the cost in 17.90 pounds. Is this a good train for you? [user] Yes please. Please book a ticket for 1 person.
Current Turn Utterances: [sys] The booking was successful your reference number is xxx. Is there anything I can help with today? [user] Yes. I would like a restaurant for the same day. the name is travellers rest.
Without augmentation: restaurant-name: travellers rest, restaurant-book day: Monday
Augmented with NeuralWOZ: restaurant-name: travellers rest
Augmented with ours: restaurant-name: travellers rest, restaurant-book day: Monday
Ground truth: restaurant-name: travellers rest, restaurant-book day: Monday

Table 6: Example dialogue state outputs from SVAG ehanced by EDZ-DA and NeuralWOZ under the data ratio setting of 1%.

Third, to evaluate the effectiveness of the dialogue complication strategy, we conduct an experiment with only the generated dialogue data without complication ("-P-Comp."). Under the data ratio setting of 1%, we observe that the joint goal accuracy score is improved a little on the MultiWOZ 2.1 test set when only dialogue data without complication is used. However, as shown in Table 5, the co-reference slot accuracy drops a lot without dialogue complication. Dialogue complication can significantly improve the model's capability to track co-reference slots. Under the data ratio setting of 5%, the generated augmentation data without dialogue complication even does harm to the co-reference slots training, which further illustrates the data complexity's importance for DST.

3.6 Case Study

In this section, we give some example output of the SVAG model enhanced by EDZ-DA and NeuralWOZ. Table 6 shows two examples. In the first example, The user express that he/she want a taxi to

get to the restaurant by the booking time. Both the models enhanced by NeuralWOZ and our method can capture the slot values for the two shared slots during the value generation stage. However, the model enhanced by NeuralWOZ fails to generate the correct "domain-slot" for the time "18:30", indicating that our data augmentation method can better enhance the model's ability to track co-reference slots. Moreover, it also demonstrates the effectiveness of our method in capturing dialogue logic with more accurate annotations.

In the second example, the user want to book "travellers rest" for the same day with the hotel booking. We find that the model enhanced by NeuralWOZ fails to capture the information "restaurant-book day: Monday," while both the models without augmentation and with our data augmentation can accurately capture the information of the shared slot. This indicates that our proposed complexity-aware method can generate complex data with accurate annotations, thereby ensuring that the augmentation does not weaken the model's reasoning ability but enhances it. In contrast, traditional methods like NeuralWOZ do not pay attention to the importance of data complexity and even weaken the model's reasoning ability by adding too much simple data. This is also shown in Table 2, where the 1% original data can only provide limited reasoning ability to the model, while the 5% data can already provide some reasoning ability for co-reference slots. Adding too much simple data will weaken the model's ability to track difficult data like co-reference slots.

3.7 Constraint Following Analysis

We do an additional evaluation of the LLM's adherence to the imposed constraints during data generation. 95.8% of the generated dialogues are retained, and the rest are deleted because they cannot match the planned dialogue flow in the process of dialogue generation. Furthermore, we sample some generated dialogue goals for further evaluation and find that all generated goals are in the form of the predefined form in the prompt. Additionally, we also sample 50 dialogue turns that contain co-reference slots to manually check the correctness of dialogue complication. 96% of these turns express the co-reference slots implicitly and correctly after the complication process. In summary, the LLM can follow our instructions well to generate correct and natural dialogues.

4 Related Work

4.1 Low-resource DST

Low-resource dialogue state tracking has received increasing attention in academia and industry. Most previous work have attempted to tackle the challenge in three ways(Jacqmin et al., 2022): (1) cross-domain transfer learning(Wu et al., 2019; Dingliwal et al., 2021); (2) cross-task transfer learning(Gao et al., 2020; Lin et al., 2021); and (3) pre-trained language model adaption(Wu et al., 2020; Su et al., 2022; Mi et al., 2022; Peng et al., 2021; Hosseini-Asl et al., 2020). Recently, more and more data augmentation based approaches have been proposed for low-resource DST. (Kim et al., 2021) proposed a collector to synthesize dialogues and a labeler to annotate the generated dialogues.(Mohapatra et al., 2021) proposed to generate dialogue data by mimicking the data collection process employed by crowd workers.Wan et al. (2022) proposed to first pre-train the user simulation model on several publicly available datasets and then tune it on target domains with few-shot data. However, all of these methods rely on the usage of the user goal from the original dataset.

4.2 Data Augmentation via LLMs

Recently, more and more studies have tended to prompt LLMs to generate synthetic training data with the purpose of augmenting data in low-resource scenarios. (Ubani et al., 2023; Zheng et al., 2023; Dai et al., 2023; Piedboeuf and Langlais, 2023) used GPT-3.5 and GPT-4 as the base generative model for data augmentation. Ubani et al. (2023) evaluated the effectiveness of zero-shot prompting for data augmentation under low-resource settings. Dai et al. (2023) used GPT to generate paraphrases of existing texts for augmentation. Both studies report better results using LLMs for data augmentation compared to previous SOTA data augmentation approaches. Piedboeuf and Langlais (2023) further compared different data augmentation methods and revealed that the performance of ChatGPT is highly dependent on the dataset. The more relevant work to ours is (Zheng et al., 2023), which used LLM to generate open-domain dialogue data for emotional support conversation. In this paper, we prompt the LLM to generate DST data, which is more challenging due to the difficulty in domain planning, the demand for accurate annotation, and the co-reference data.

5 Conclusions and Future Work

In this paper, we propose EDZ-DA, an easy-to-difficult zero-shot data augmentation framework for low-resource DST. We reveal three issues in constructing DST data and propose to first determine the logical relationship among domains and generate the user goal with the help of the LLM's strong reasoning ability. In order to enhance the DST model's performance in tracking co-reference slots, we propose to complicate the dialogue content based on the domain relationship. Moreover, we propose to permute slot values to mitigate the influence of output order and the incomplete generation problem. Experimental results on the MultiWOZ dataset illustrate the superiority of EDZ-DA over previous data augmentation approaches for low-resource DST.

In future work, we will further study how to generate diverse natural dialogue flows.

Limitations

In this section, we discuss several limitations of our proposed framework. First, although our generated augmentation data can significantly improve low-resource DST, the naturalness of the dialogue process can be further improved. In practice, there may be situations such as booking failure and requalification. Future work can look into studying how to prompt the LLM to plan such dialogues. Second, the prompts in our method are manually constructed. How to explore a more systematic method for prompt engineering leaves future direction for our work. Finally, it could be interesting to investigate the performance of other LLMs such as LLaMA(Touvron et al., 2023) in this task.

Ethics Statement

In our paper, we propose an LLM-based data augmentation method for low-resource DST. We choose GPT-4 for generating all the augmentation data and use T5 as the backbone model of our DST model. We carefully check all outputs in our experiments and we do not observe any ethical issues. Moreover, we conduct our experiments on the MultiWOZ dataset which is a publicly-available benchmark, and in our view, it does not have any attached privacy or ethical issues. In summary, there are no direct ethical concerns in our study.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments. This research is funded by the Science and Technology Commission of Shanghai Municipality Grant (No. 22511105901).

References

Derek Chen, Kun Qian, and Zhou Yu. 2023. Stabilized in-context learning with pre-trained language models for few shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 1506–1519. Association for Computational Linguistics.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation.

Saket Dingliwal, Shuyang Gao, Sanchit Agarwal, Chien-Wei Lin, Tagyoung Chung, and Dilek Hakkani-Tür. 2021. Few shot dialogue state tracking using meta-learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 1730–1739. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428. European Language Resources Association.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023. Towards llm-driven dialogue state tracking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 739–755. Association for Computational Linguistics.

Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Hakkani-Tür. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. *CoRR*, abs/2004.05827.

AV Glazkova and DA Morozov. 2023. Applying transformer-based text summarization for keyphrase generation. *Lobachevskii Journal of Mathematics*, 44(1):123–136.

Ming Gu, Yan Yang, Chengcai Chen, and Zhou Yu. 2024. State value generation with prompt learning and self-training for low-resource dialogue state tracking. In *Proceedings of the 15th Asian Conference on Machine Learning*, volume 222 of *Proceedings of Machine Learning Research*, pages 390–405. PMLR.

Ting Han, Ximing Liu, Ryuichi Takanobu, 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 *Natural Language Processing and Chinese Computing - 10th CCF International Conference, NLPCC 2021, Qingdao, China, October 13-17, 2021, Proceedings, Part II*, volume 13029 of *Lecture Notes in Computer Science*, pages 206–218. Springer.

Michael Heck, Nurul Lubis, Benjamin Matthias Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, and Milica Gasic. 2023. Chatgpt for zero-shot dialogue state tracking: A solution or an opportunity? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 936–950. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2627–2643. Association for Computational Linguistics.

Léo Jacqmin, Lina Maria Rojas-Barahona, and Benoît Favre. 2022. "do you follow me?": A survey of recent approaches in dialogue state tracking. *CoRR*, abs/2207.14627.

Sungdong Kim, Minsuk Chang, and Sang-Woo Lee. 2021. NeuralWOZ: Learning to collect task-oriented dialogue via model-based simulation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3704–3717, Online. Association for Computational Linguistics.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Zhenpeng Zhou, Paul A. Crook, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021. Zero-shot dialogue state tracking via cross-task transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7890–7900. Association for Computational Linguistics.

Yichao Luo, Yige Xu, Jiacheng Ye, Xipeng Qiu, and Qi Zhang. 2021. Keyphrase generation with fine-grained evaluation-guided reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 497–507, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Fei Mi, Yasheng Wang, and Yitong Li. 2022. CINS: comprehensive instruction for few-shot learning in task-oriented dialog systems. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11076–11084. AAAI Press.

Biswesh Mohapatra, Gaurav Pandey, Danish Contractor, and Sachindra Joshi. 2021. Simulated chats for building dialog systems: Learning to generate conversations from instructions. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1190–1203, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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. *Trans. Assoc. Comput. Linguistics*, 9:907–824.

Frédéric Piedboeuf and Philippe Langlais. 2023. Is ChatGPT the ultimate data augmentation algorithm? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15606–15615, Singapore. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Jamin Shin, Hangyeol Yu, Hyeongdon Moon, Andrea Madotto, and Juneyoung Park. 2022. Dialogue summaries as dialogue states (ds2), template-guided summarization for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3824–3846. Association for Computational Linguistics.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4661–4676. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Solomon Ubani, Suleyman Olcay Polat, and Rodney Nielsen. 2023. Zeroshotdataaug: Generating and augmenting training data with chatgpt.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2016. Order matters: Sequence to sequence for sets. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Dazhen Wan, Zheng Zhang, Qi Zhu, Lizi Liao, and Minlie Huang. 2022. A unified dialogue user simulator for few-shot data augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3788–3799, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chien-Sheng Wu, Steven C. H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 917–929. Association for Computational Linguistics.

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 *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 808–819. Association for Computational Linguistics.

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, SIGDIAL 2022, Edinburgh, UK, 07-09 September 2022*, pages 351–360. Association for Computational Linguistics.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2Set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608, Online. Association for Computational Linguistics.

Chujie Zheng, Sahand Sabour, Jiaxin Wen, Zheng Zhang, and Minlie Huang. 2023. AugESC: Dialogue augmentation with large language models for emotional support conversation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1552–1568, Toronto, Canada. Association for Computational Linguistics.

A Appendix

Prompt:
There are five domains that the AI agent supported, and their slots are listed in the following:
1. Hotel: {area, type, internet, parking, name, book day, price range, star, book stay, book people}
2. Restaurant: {area, book day, book people, book time, food, name, price range}
3. Attraction: {area, name, type}
4. Taxi: {arrive by, departure, destination, leave at}
5. Train: {book people, day, departure, destination, leave at}
6. Flight: {arrive by, departure, destination, leave at}
Categorical slots and their possible values:
1. Area: centre, east, south, west, north
2. Internet: yes, no
3. Parking: yes, no
4. Price range: cheap, moderate, expensive
5. Star: 1, 2, 3, 4, 5
6. Day: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday
7. Hotel-type: hotel, guest house
8. Book people: 1, 2, 3, 4, 5, 6, 7, 8
9. Book stay: 1, 2, 3, 4, 5, 6, 7, 8
10. Arrive by & leave at: time in forms of "xx:xx" such as "13:00"
I will give you some of these domains. Your task is to determine whether these domains can be fused together in a dialogue reasonably. The order of these domains does not matter, and you can determine the order of these domains yourself. Please output it in a json format like {"is_reasonable": <1 for reasonable and 0 the opposite>, "explanation": <give some explanation about your judgement>}.

(Example-1)# domains: restaurant, taxi.

(Example-2)# domains: flight, train

GPT output of example-1:
{

"is_reasonable": 1,
"explanation": "The domains of restaurant and taxi can be reasonably fused together in a dialogue. A common scenario is when a user wants to book a restaurant and then needs a taxi to get to the restaurant. The user might want to book a table at a specific time and then arrange for a taxi to arrive or depart accordingly. The slots for restaurant bookings (such as area, book day, book time, and book people) can be discussed first, followed by arranging a taxi (with departure, destination, leave at, and arrive by) to ensure the user arrives on time for their reservation."
}

GPT output of example-2:
{

"is_reasonable": 0,
"explanation": "Fusing the flight and train domains in a single dialogue could be confusing and impractical because they both serve the purpose of transporting a person from one location to another, but they are typically used for different types of journeys. Flights are generally used for longer distances or international travel, while trains are often preferred for regional or shorter distance travel. Including both in a single dialogue might suggest an unrealistic scenario or a misunderstanding of the user's travel needs. Additionally, coordinating between the two modes of transport would involve complex logistics, such as timing and transfer points, which are beyond the scope of a simple dialogue system."
}

Table 7: Two examples of domain relation judgment. GPT determines that it is reasonable for the restaurant domain and taxi domain to appear in one dialogue but fusing the flight and train domains is confusing and impractical.

Prompt:
So, based on your judgement, please help me generate several dialogue states with different inferrable information. Each dialogue state is in forms of a dictionary where the key is the domain-slot and the value is the corresponding value. The dialogue state summarizes a user's goal in his/her dialogue with an AI agent. You should determine the number of states yourself based on your judgement and output your answer with a list of dialogue states.
You can determine the values of the categorical slots yourself. Values of other non-categorical slots should be left blank (an empty string), and I will fill them. Note that values should be logical among different domains. What's more, some the values may be the value of another domain. Please express this information by "domain1-slot": "domain2-slot". But note that you cannot refer to domains that are not mentioned in the current domains. And there are some restrictions in my system:
1) slots 'departure', 'destination' cannot be inferred from slot 'area'.
2) slots 'arrive by', 'leave at', 'book time' cannot be inferred from slots 'day', 'book day' and vice versa.
3) area can only be inferred from area.
4) train-destination and train-departure cannot be inferred from other domain.
5) inferable slots must be different domains.
6) Please carefully refer to the schema I have given in the former turn. Only legal slots in a certain domain can be inferred.
7) Infer from a blank slot is allowed since I will fill in all the blank slots later.

GPT output:
Based on your instructions, here are several dialogue states with inferable information, formatted as dictionaries. Each state represents a user's goal in a conversation with an AI agent, with some values left blank for you to fill in later:
[
{"taxi-departure": "", "taxi-destination": "restaurant-area", "taxi-leave at": "restaurant-book time", "restaurant-book time": "", "restaurant-area": "", "restaurant-food": "Italian"},

{"taxi-departure": "", "taxi-destination": "restaurant-name", "taxi-arriveby": "restaurant-book time", "restaurant-name": "", "restaurant-book time": "", "restaurant-pricerange": "moderate"},
{"taxi-departure": "hotel-name", "taxi-destination": "restaurant-name", "taxi-arriveby": "", "hotel-name": "", "restaurant-name": "", "restaurant-book time": "taxi-arriveby"},
{"taxi-departure": "hotel-area", "taxi-destination": "", "taxi-leaveat": "", "hotel-area": "", "restaurant-name": "taxi-destination", "restaurant-booktime": "taxi-leaveat"}
]
In these dialogue states, I've adhered to the restrictions you've provided. The values for categorical slots have been determined, and the non-categorical slots have been left blank. The inferable information is based on the schema you mentioned, ensuring that the slots are inferred from different domains and that the restrictions on which slots can be inferred from which are respected.

Table 8: An example of seed state generation. Words in green are one of the selected seed states.

Prompt:

Given the following dialogue state with various domain-slot pairings that include specific details:

{
"restaurant-name": "midsummer house restaurant",
"restaurant-book time": "12:30",
"restaurant-area": "centre",
"restaurant-book people": "1",
"restaurant-food": "british",
"restaurant-book day": "wednesday",
"restaurant-pricerange": "expensive",
"taxi-destination": "midsummer house restaurant",
"taxi-arriveby": "12:30",
"taxi-departure": "peterborough train station"
}.

Construct the user's overall goal for the ensuing conversation based solely on the information provided in the dialogue state. The user goal should be as specific as possible while adhering to the following guidelines:

- Do not include any specific names of restaurants, hotels and attractions that are present in the dialogue state. These details are meant to be suggested by the agent during the conversation.
- Do not infer or add any information beyond what is present in the dialogue state. Use only the details provided in the slots.
- The order in which the domain-slot pairings are presented is not indicative of importance or the sequence of addressing them in the dialogue. The language model should determine the logical flow and prioritization independently.
- Ensure that the user goal lays a foundation for a focused yet dynamic interaction, where the agent has the opportunity to make recommendations and the user can respond or confirm these suggestions. The objective is to clarify what the user intends to achieve or learn during the interaction without presuming or preempting the agent's role in guiding the conversation.

Example of a properly constructed user goal:

"User aims to find [general need based on dialogue state], with preferences matching [specific preferences], and is looking for recommendations on [broad categories without naming specific entities]."

Please proceed by formulating a user goal that aligns with these criteria.

GPT output:

The user aims to book a table for dining at a high-end restaurant that serves British cuisine in the city center, with the reservation set for one person on a Wednesday at 12:30 PM. Additionally, the user requires transportation arrangements to be made from a train station to the restaurant, ensuring arrival by the reserved time. The user is looking for recommendations on suitable dining establishments that fit these criteria and for a taxi service that can accommodate the specified departure and arrival details.

Table 9: An example of user goal generation. This prompt template is further improved by ChatGPT after we manually construct it.

Prompt:
Dialogue State: <synthetic dialogue state>
User Goal: <generated user goal>
Information for the Agent: <additional information in the database for a certain place>
Instruction:
The dialogue state summarize the dialogue between the user and the agent. Your task is to generate a dialogue flow. Each element in the dialogue flow determine the information of a dialogue turn. Each element should contain a description for the dialogue turn and the corresponding turn state. The description is used to guide the subsequent turn utterance generation. The state is a subset of the dialogue state that describe the information the user mentioned or confirmed in the dialogue turn. Note that the dialogue is started with the agent and ended with the user and must follow an alternating pattern between the agent and the user. After a successful booking service, the model should provide a synthetic reference number, which consists of 8 random characters. Sometimes there are a lot of restriction for a domain, so you should construct more dialogue turns to make sure that the user express no more than six new restrictions of domain-slots in one dialogue turn. The value 'dontcare' means that the user does not care about the restriction of that domain-slot and this information must be expressed by the user.
Remember that all the states in the dialogue state should be used and no additional information can be added.
Remember that the agent does not have prior knowledge of the user's goals.
Ensure that the user express no more than six new restriction of domain-slots in one dialogue turn. Be meticulous to confirm that the user does not introduce more than six new restrictions of domain-slots within a single turn. To comply with this, divide constraints across multiple turns when necessary.

Parsed GPT output:
{ "description": "The agent greets the user and asks what they can assist them with today.",
"turn state": {} },
{ "description": "The user requests to book a table at a high-end restaurant that serves British cuisine in the city center for one person.",
"turn state":{ "restaurant-area": "centre", "restaurant-food": "british", "restaurant-book people": "1", "restaurant-pricerange": "expensive" } },
... ,
{ "description": "The user confirms the booking at Midsummer House Restaurant and requests a taxi to get there, specifying they need to arrive by 12:30 PM.",
"turn state": { "taxi-destination": "midsummer house restaurant", "taxi-arriveby": "12:30" } },...

Table 10: An example of dialogue flow generation.

Prompt:
Information for the Agent: <additional information in the database for a certain place>
Dialogue Flow: <generated dialogue flow>
Instruction:
Based on the dialogue flow, please generate a dialogue between a agent and a user. You should include information provided in the description and the turn state while construct a dialogue turn. The synthetic reference number should be a string consists of 8 random character. Remember no turn in the dialogue flow can be omitted and no additional dialogue turn can be added. The generated dialogue turns must correspond one-to-one with the provided dialogue flow of, neither more nor less. Remember no turns in the dialogue flow can be omitted and no additional dialogue turn can be added.
Output Format: <output format>

Parsed GPT output:

{
"role": "agent",
"description": "The agent greets the user and asks what they can assist them with today.",
"content": "Good day! How may I assist you today?"
},
{
"role": "user",
"description": "The user requests to book a table at a high-end restaurant that serves British cuisine in the city center for one person.",
"content": "I'd like to book a table for one at a high-end British cuisine restaurant in the city center."
},
...
{
"role": "agent",
"description": "The agent confirms the reservation details and informs the user that they have a table available at Midsummer House Restaurant, which fits all the criteria provided.",
"content": "I have a table available for you at Midsummer House Restaurant, which offers British cuisine and is located in the city center. It's quite an exquisite place, and it fits your criteria perfectly."
},
{
"role": "user",
"description": "The user confirms the booking at Midsummer House Restaurant and requests a taxi to get there, specifying they need to arrive by 12:30 PM.",
"content": "That sounds wonderful. Please confirm my booking at Midsummer House Restaurant. Also, I need a taxi to get there and arrive by 12:30 PM."
}
...

Table 11: An example of dialogue generation.

Prompt:
Dialogue History: <dialogue history>
Current Turn Utterances:
system: i have a table available for you at midsummer house restaurant , which offers british cuisine and is located in the city center . it s quite an exquisite place , and it fits your criteria perfectly.
user: that sounds wonderful . please confirm my booking at midsummer house restaurant . also , i need a taxi to get there and arrive by 12:30 pm.
Turn State: <turn state>
Co-reference:
{"taxi-destination": "restaurant-name", "taxi-arriveby": "restaurant-book time"}
Instruction:
Your task is to modify the current turn utterances to express the co-reference information implicitly rather than explicitly stating the full name of the value. Note that the rest of the values in the current turn state should still be expressed explicitly. The modified utterance should be the paraphrase of the original utterance in the current turn for both the system and the user. Only to modify the information for these co-reference value.
Remember to maintain the meaning of the modified utterance with the original one.
If the expression of the co-reference value in the original utterance is already in a implicit way, please do not modify the utterances and just copy it.
Output Format:
{
“description”: <description of the co-reference information in natural language>,
“system”: <modified system utterance of the current turn>,
“user”: <modified user utterance of the current turn>
}

Parsed GPT output:
{
"description": "The ‘taxi-destination’ is the same as the ‘restaurant-name’, which is Midsummer House Restaurant, and the ‘taxi-arriveby’ time is the same as the ‘restaurant-book time’, which is 12:30 pm.",
"system": "I have a table available for you at Midsummer House Restaurant, which offers British cuisine and is located in the city center. It’s quite an exquisite place, and it fits your criteria perfectly.",
"user": Ṫhat sounds wonderful. Please confirm my booking at Midsummer House Restaurant. Also, I need a taxi to get there in time for my reservation.
}

Table 12: An example of dialogue complication. As described in "# Co-reference", taxi-arriveby should be inferred from restaurant-book time. GPT modify the expression of that information. Words in red are the expression before modification and words in green are the expression after modification.

The original training sample:
Dialogue History: <dialogue history>
Current Turn Utterances:
system: What can I help you today? user: I want to have dinner in a restaurant that serves Chinese food, and it should be in the city centre.
Target Output: "Chinese | centre"

After permutation:
Training sample-1:
Dialogue History: <dialogue history>
Current Turn Utterances:
system: What can I help you today? user: I want to have dinner in a restaurant that serves Chinese food, and it should be in the city centre.
Target Output: "Chinese | centre"
Training sample-2:
Dialogue History: <dialogue history>
Current Turn Utterances:
system: What can I help you today? user: I want to have dinner in a restaurant that serves Chinese food, and it should be in the city centre.
Target Output: "centre | Chinese"

Table 13: An example of slot value permutation. For each training sample in the dataset, we will permute the slot values. Therefore, the sample containing 2 values will be augmented to 2 samples, the sample containing 3 values will be augmented to 6 samples. And so on, for each sample.