Controllable Dialogue Simulation with In-Context Learning

Zekun Li[1], Wenhu Chen[2], Shiyang Li[1], Hong Wang[1], Jing Qian[1], Xifeng Yan[1]
[1]University of California, Santa Barbara
[2]University of Waterloo, Vector Institute
{zekunli, shiyangli, hongwang600, jing_qian, xyan}@cs.ucsb.edu
wenhuchen@uwaterloo.ca

Abstract

Building dialogue systems requires a large corpus of annotated dialogues. Such datasets are usually created via crowdsourcing, which is expensive and time-consuming. In this paper, we propose DIALOGIC, a novel dialogue simulation method based on large language model in-context learning to automate dataset creation. Seeded with a few annotated dialogues, DIALOGIC automatically selects in-context examples for demonstration and prompts GPT-3 to generate new dialogues and annotations in a controllable way. Our method can rapidly expand a small set of dialogue data with minimum or zero *human involvement* and *parameter update* and is thus much more cost-efficient and time-saving than crowdsourcing. Experimental results on the MultiWOZ dataset demonstrate that training a model on the simulated dialogues leads to even better performance than using the same amount of human-generated dialogues under the challenging low-resource settings, with as few as 85 dialogues as a seed. Human evaluation results also show that our simulated dialogues have near-human fluency and annotation accuracy. The code and data are available at https://github.com/Leezekun/dialogic.

1 Introduction

Task-oriented dialogue (TOD) systems can assist users in completing tasks such as booking a restaurant or making an appointment. Building such a dialogue system requires a large corpus of annotated dialogues to train the models (Wu et al., 2020), which is costly to obtain in terms of money and time.

One popular approach to collecting and annotating task-oriented dialogues is crowdsourcing via a Wizard-of-Oz setup (Mrksic et al., 2017; Eric et al., 2017; Budzianowski et al., 2018), where crowdworkers produce conversations. Significant annotation efforts are further needed to label intent, entities, etc. Prior work has been proposed to minimize the cost and effort in data collection by hiring crowdworkers or leveraging user simulators to interact with existing dialogue systems (Williams et al., 2013; Shah et al., 2018b,a; Papangelis et al., 2019; Zhao et al., 2019; Rastogi et al., 2020; Tseng et al., 2021). Typically they utilize pre-defined rules and hand-crafted templates, which can not guarantee the naturalness and diversity of generated dialogues and require considerable engineering efforts and human involvement. To solve the problem, Mohapatra et al. (2020) proposed a method that uses GPT-2 (Radford et al., 2019) to simulate the user, system, and their interactions. However, it requires human-written instructions for generation and a considerable amount of seed dialogues for simulator training to ensure acceptable generation quality.

In recent years, large language models (LLMs) (Brown et al., 2020; Lieber et al., 2021; Rae et al., 2021; Thoppilan et al., 2022; Smith et al., 2022) demonstrate strong in-context learning capability. Provided with a few in-context examples, the LLMs, such as GPT-3 (Brown et al., 2020), can generate text with similar patterns without fine-tuning. This capability has been leveraged to synthesize training data in a few NLP tasks (Wang et al., 2021b; Liu et al., 2022). Although there have been methods that generate training data for a single component in the TOD systems (Li et al., 2022), there hasn't been a plausible solution to generate whole dialogues with annotations for end-to-end training due to its complex nature of involving multi-turn interactions, multiple possible logic flows, and multiple types of annotations.

To address the challenge, we introduce a controllable dialogue simulation method DIALOGIC for dialogue dataset creation. Seeded with a few seed dialogues, DIALOGIC automatically selects in-context examples for demonstration and prompts LLMs such as GPT-3 to generate annotated dialogues in a

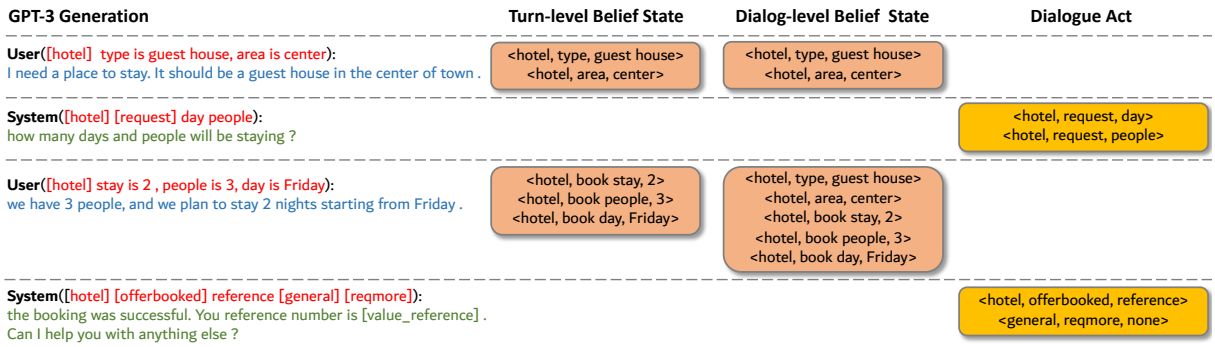

Figure 1: Illustration of a partial annotated dialogue generated by our method. **Left**: the conversations and annotations are generated simultaneously by GPT-3, where the user utterances are in blue, the system responses are in green, and the annotations are in red. **Right**: the parsed structured annotation from the GPT-3 generation. Best viewed in color. A complete generated dialogue is shown in Appendix G as Table 9.

controllable way. Figure 1 illustrates a partial example. For the user side, GPT-3 is prompted first to generate the turn-level user goal (belief state), conditioned on which the user utterance that expresses the goal will be generated. Likewise, we prompt GPT-3 to generate the dialog act for the system side and then the corresponding system response. We also propose automatic verification and revision methods to mitigate the annotation errors.

This paper has two key insights. First, leveraging the in-context learning ability of LLMs, our method, DIALOGIC, can generate annotated dialogues by learning from a few example dialogues. Except for the minimum efforts in collecting the small seed dataset and training an auxiliary model on that, the simulation process is free of *human involvement* and *parameter update*, making our method much cheaper and faster than crowdsourcing in dataset creation. Specifically, a large-scale and high-quality dataset such as MultiWOZ (Budzianowski et al., 2018) can be created using our method within only several hours. Second, we design controllable dialogue generation strategies to overcome the deficiency of GPT-3 in lack of reliability and interpretability. We also investigate effective representations and selection strategies of in-context dialogue examples for LLMs to better leverage their in-context learning capabilities.

We conduct experiments on MultiWOZ2.3 (Han et al., 2021) dataset. Remarkably, in the challenging low resource settings where as low as only 85 seed dialogues (1% of the whole training dataset) are given, the dialogues simulated by our method lead to even better model performance than the same amount of human-generated dialogues. DIALOGIC can also serve as an effective data augmentation method when enough training data is provided. Human evaluations indicate that our simulated dialogues have near-human fluency and annotation accuracy. Our results demonstrate the promise of leveraging large language model to automate dialogue dataset creation. We have released the code and simulated data to facilitate future studies.[1]

2 Related Work

2.1 Dialogue Collection and Simulation

Building end-to-end dialogue systems heavily relies on annotated training data. Wizard-of-Oz (Kelley, 1984), as a popular approach, is able to produce high-quality conversations but totally relies on human efforts (Mrksic et al., 2017; Eric et al., 2017; Asri et al., 2017; Budzianowski et al., 2018). There are also dialogue corpora of interactions between humans and existing dialogue systems or APIs (Williams et al., 2013, 2014; Raux et al., 2005). To further reduce human efforts, user simulators are leveraged to interact with the system via reinforcement learning or self-play (Shah et al., 2018b,a; Papangelis et al., 2019; Zhao et al., 2019; Rastogi et al., 2020; Tseng et al., 2021). However, existing dialogue systems or APIs are still needed, which restricts these solutions to existing domains. To this end, Mohapatra et al. (2020) proposed a method that utilizes GPT-2 (Radford et al., 2019) to simulate both the user and system side. However, this method still needs many dialogues to train the simulators and cannot guarantee the simulation quality in low-resource settings.

2.2 Task-oriented Dialogue

A task-oriented dialogue system usually consists of three components: natural language understanding

[1] https://github.com/Leezekun/dialogic

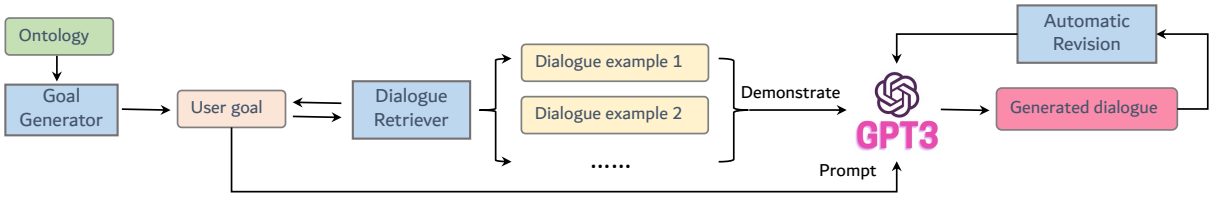

Figure 2: Overview of the proposed method.

(NLU) for dialogue state tracking, dialogue management (DM) for predicting the dialog act based on the dialogue states, and natural language generation (NLG) for mapping dialog act to natural language response. The annotated data of belief states, dialog acts, and system responses are needed to train these components whether in a separate way (Wu et al., 2019; Lee et al., 2019; Heck et al., 2020), or an end-to-end fashion (Peng et al., 2021; Hosseini-Asl et al., 2020; Lin et al., 2020; Yang et al., 2021; Su et al., 2021). In this paper, we aim to generate dialogues and their complete set of annotations.

2.3 In-Context Learning

As an alternative to finetuning, in-context learning with LLMs, such as GPT-3 (Brown et al., 2020), can perform a new task by learning from in-context examples. Due to the superior few-shot performance and scalability, in-context learning has been applied to a wide range of NLP tasks. As for dialogue tasks, in-context learning has been increasingly deployed in tasks such as intent classification (Yu et al., 2021), semantic parsing (Shin and Van Durme, 2021), and dialogue state tracking (Hu et al., 2022). Madotto et al. (2021) built an end-to-end dialogue system solely based on in-context learning. Despite its success, GPT-3 requires a large number of resources to be deployed. And its public API is charged based on the length of input text. What's worse, the limitation of input length restricts the number of in-context examples and thus the generation performance. Consequently, a few methods have been proposed to leverage GPT-3 to synthesize training data to train smaller models for inference (Wang et al., 2021a,b; Liu et al., 2022). Although it is especially desirable for dialogue tasks as the input prompt of dialogues is usually lengthy, there hasn't been a plausible solution to generating a full set of annotated dialogue data for TOD systems due to its complex nature of involving multi-turn interactions and multiple types of annotations.

3 Method

In this paper, we introduce a novel method DIALOGIC to simulate annotated dialogues for building task-oriented dialogue systems based on language model in-context learning. The only requirements are a small seed dataset $\mathcal{D}_s$ consisting of a few annotated dialogues and an ontology $\mathcal{O}$ that includes all slots and possible slot values for each domain. An auxiliary TOD model $\mathcal{M}$ such as SimpleTOD (Hosseini-Asl et al., 2020) and PPTOD (Su et al., 2021) trained on $\mathcal{D}_s$ will be used to verify and revise generated annotations. Our goal is to expand $\mathcal{D}_s$ by generating new dialogues. For each turn of the dialogues, we need to generate the user utterance U, belief state B, database (DB) query result Q, dialog act A, and system responses S (we omit the turn index for brevity).

We will elaborate the design of our method using a well-studied task-oriented dialogue dataset MultiWOZ (Budzianowski et al., 2018; Eric et al., 2020; Han et al., 2021), which covers 7 domains such as *hotel* and *restaurant*, and 24 slots such as *hotel-area* and *restaurant-food* (see Appendix A for more details). To simulate the low-resource environment, we use 1%, 5%, 10% of the training dataset as the seed dataset $\mathcal{D}_s$.

3.1 Overview

A partial example of a simulated dialogue is shown in Figure 1. The pipeline of our method is illustrated in Figure 2. For a domain, the goal generator will take the ontology $\mathcal{O}$ as input to generate a new user goal $\mathcal{G}_i$. Then we select a few seed dialogues with similar user goals from $\mathcal{D}_s$ as the in-context example for GPT-3. Given the user goal $\mathcal{G}_i$ and the seleted in-context examples, we leverage GPT-3 to generate a new dialogue $\mathcal{C}_i$. As the generated data may fail to satisfy our requirement, we design methods for automatic verification and revision.

3.2 In-context Example

User Goal A task-oriented dialogue is a conversation where the dialogue system helps accomplish

the user's goal. For a new dialogue $\mathcal{C}_i$, we first generate its user goal $\mathcal{G}_i$ based on the ontology. The user goal and belief state are a set of domain-slot-value triplets: (*domain, slot_name, slot_value*). For example, when a user wants to book a 4-star hotel for 2 nights, and a cheap restaurant that serves Chinese food, his user goal will be {(*hotel, stars, 4*), (*hotel, book stay, 2*), (*restaurant, pricerange, cheap*), (*restaurant, food, chinese*)}. We investigate several ways to generate the user goal, i.e., determining the domains, slots, and slot values to be selected, which will be discussed as follows.

Example Selection Given the target user goal $\mathcal{G}_t$, we select a few seed dialogues as in-context examples, from which GPT-3 can learn to generate the target dialogue $\mathcal{C}_i$. To achieve that, the selected dialogue examples should contain as much ontology information needed in the target dialogue (i.e., mentioned slots) as possible so that GPT-3 can mimic the "in-domain" generation. To measure how two dialogue goals $\mathcal{G}_i$ and $\mathcal{G}_j$ overlap, we calculate their similarity as:

$$w_{ij} = \left| \frac{D(\mathcal{G}_i) \bigcap D(\mathcal{G}_j)}{D(\mathcal{G}_i) \bigcup D(\mathcal{G}_j)} \right| \cdot \left| \frac{S(\mathcal{G}_i) \bigcap S(\mathcal{G}_j)}{S(\mathcal{G}_i) \bigcup S(\mathcal{G}_j)} \right|, \quad (1)$$

where $D(\mathcal{G}_i)$ and $S(\mathcal{G}_i)$ denote the set of domains and slots in the user goal $\mathcal{G}_i$, respectively. The first part is the Jaccard similarity (Niwattanakul et al., 2013) of the domain set, while the second part is that of the slot set. The probability of a dialogue $\mathcal{C}_j$ from the seed dataset $\mathcal{D}_s$ being sampled as in-context examples for the target dialogue $\mathcal{C}_i$ is:

$$p_j = \frac{e^{w_{ij}/\tau}}{\sum_{k \in \mathcal{D}_s} e^{w_{ik}/\tau}}, \quad (2)$$

where τ is the temperature. A higher temperature will introduce more randomness and diversity in example selection.

We investigate several ways to generate user goals and select in-context examples:

- *Random Sampling*: we randomly select domains, slots, and slot values to form a user goal and sample in-context examples as described in Equation 2. In this way, we can generate any unseen user goal and thus the corresponding dialogues. However, as the number of seed dialogues is limited, it is hard to guarantee that the sampled dialogue examples can cover all the information required for generating the target dialogue.

Instruction: You are going to book a hotel, and your requirements for the hotel are ([hotel] type is hotel , pricerange is cheap , parking is yes , stay is 2 , day is tuesday , people is 6). Make sure you get the booking information once booked.

Conversation:
User ([hotel] type is hotel , pricerange is cheap): I am looking for a place to to stay that has cheap price range it should be in a type of hotel .
Assistant ([hotel] [request] area): okay , do you have a specific area you want to stay in ?
User ([hotel] parking is yes , pricerange is cheap): no , i just need to make sure it is cheap . oh , and i need parking .
Assistant ([hotel] [inform] price choice parking type [offerbook]): i found [value_choice] [value_price] [value_type] for you that include -s parking . do you like me to book it ?
User ([hotel] stay is 3 , day is tuesday , people is 6): yes , please . 6 people 3 nights starting on tuesday .
Assistant ([hotel] [nobook] day [request] stay day): i am sorry but i was n't able to book that for you for [value_day] . is there another day you would like to stay or perhaps a shorter stay ?
User ([hotel] stay is 2): how about only 2 nights .
Assistant ([hotel] [offerbooked] reference [general] [reqmore]): booking was successful . reference number is : [value_reference] . anything else i can do for you ?
User ([general]): no , that will be all . goodbye .
Assistant ([general] [bye]): thank you for using our services .

Figure 3: Illustration of an in-context example from the MultiWOZ dataset. The user goal, belief states, and dialog acts are in red. User utterances are in blue, while system responses are in green. Best viewed in color.

- *Value Substitution*: we only substitute the slot values of the seed dialogues' user goal to form a new user goal. This method can ensure that all the required slots are mentioned in the in-context examples. However, GPT-3 will tend to replicate the in-context examples, and thus few diversity can be introduced.
- *Combination*: we first select a few dialogues from the seed dataset and then combine their user goals to create a new goal. As the new user goal might involve too many domains and slots, we randomly drop some slots. This method can ensure that all the mentioned slots in the target user goal are covered in the examples and encourage the GPT-3 to generate diverse data.

We experimentally found the *Combination* method yields the best performance. More details, comparison, and discussion of different goal generation methods can be found in Appendix D.

Demonstration To better demonstrate the desired pattern of generated data for a dialogue to GPT-3, we design the format for the example dialogues as shown in Figure 3. The user goal and belief state are converted from a sequence of triplets to the natural language via a template. For example, the user goal of {(*hotel, stars, 4*), (*hotel, book stay, 2*), (*restaurant, pricerange, cheap*), (*restaurant, food, chinese*)} will be converted to *[hotel] star is 4 ,*

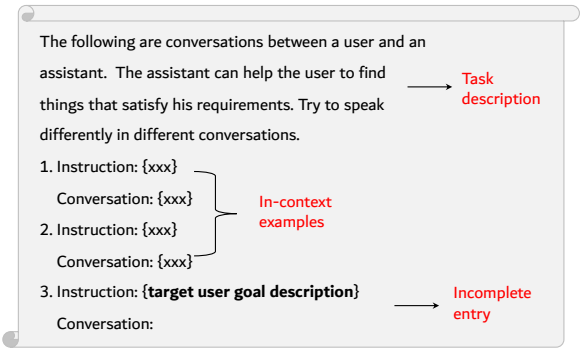

Figure 4: Template for the prompt of GPT-3 to generate new dialogues. An actual example of the complete prompt is shown in Appendix F as Table 8.

book stay is 2 [restaurant] pricerange is cheap , food is chinese, where *[domain]* separates domains and the comma separates slots in each domain.

As for the *conversation* part, the desired annotations are incorporated with the utterances for each turn. For the user side, GPT-3 will generate the user utterance and the turn-level belief state, i.e., the user goal mentioned in this turn. Dialog acts and their corresponding system response are needed for the system side. Similarly, the dialog act is also a set of triplets (*domain, action_type, slot_name*) and is converted to natural language similarly to belief states. An actual example of the demonstration can be seen in Table 8.

3.3 Prompt Design

Given a prompt consisting of the task description, a few in-context examples, and an incomplete entry, we instruct GPT-3 to generate text to complete the entry. A template of our prompt is shown in Figure 4. The format of in-context examples is described in Section 3.2, which consists of an *instruction* (user goal), and the *conversation*. As for the incomplete entry, the target user goal description is used as the *instruction*, and GPT-3 will generate the corresponding *conversation* in a controllable way, which will be described in the next section.

3.4 Controllable Dialogue Generation

Considering GPT-3's known deficiencies in lack of reliability and interpretability, we propose methods to control GPT-3 when it generates dialogue data. In addition, we design automatic revision methods to minimize potential annotation errors. Figure 5 illustrates the controllable generation process of a dialogue turn.

For the user side, GPT-3 will generate the belief state $\hat{B}$ and the corresponding user utterance U.

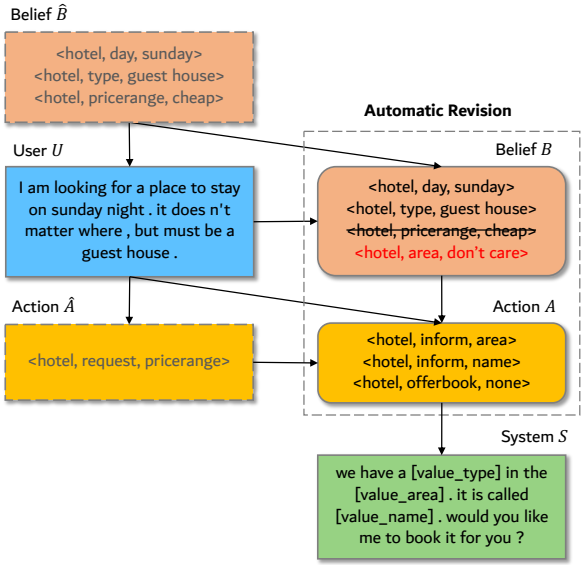

Figure 5: Illustration of the controllable generation process of a dialogue turn. An example of the generation process of a complete dialogue is shown in Appendix F as Table 9.

The belief state is expected to be consistent with the the user utterance. We keep U unchanged as the final user utterance and check the annotation errors in the generated belief state $\hat{B}$, which can be categorized into two types. Taking the example in Figure 5 for illustration, *(hotel, stay, 1)*, as a part of the original generated belief $\hat{B}$, doesn't appear in the user utterance, which is called *over-generation*. On the contrary, the value *don't care* for slot *hotel-area* is mentioned by the user but not included in U, which is called *de-generation* (Li et al., 2020). We utilize an *auxiliary generator* and *slot-value match filter* to mitigate *de-generation* and *over-generation* issues, respectively.

Auxiliary Generator To tackle the de-generation issue, we try to detect as many mentioned slots in the user utterances as possible. To this end, we utilize an auxiliary TOD model $\mathcal{M}$ trained on the seed dataset $\mathcal{D}_s$ to generate its predicted belief state $\bar{B}$, conditioned on the dialogue context of all previous turns and user utterance U of the current turn. $\bar{B}$ could be complementary to $\hat{B}$ generated by GPT-3. We found that GPT-3 sometimes forgets to generate all or even any belief state. If not corrected, GPT-3 will continue the errors in the following turns. Therefore, it is nontrivial to utilize the auxiliary generator, though not well trained when the seed data is limited, to mitigate the de-generation issue. If enough seed data is given, the auxiliary generator can better detect the belief state slots forgotten by GPT-3 and ensure the annotation quality.

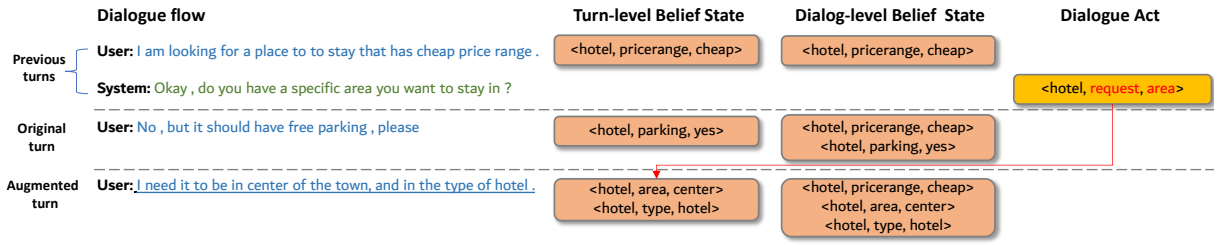

Figure 6: Illustration of the turn-level generation for DST augmentation. The turn-level belief state is decided on the dialog act of the last turn. The user utterance, which is underlined, is generated conditioned on the turn-level belief state. The newly generated user turn will be concatenated with previous turns of the original dialogue to form a new dialogue. An example of the generated user turn is shown in Appendix as Table 11.

Slot-value Match Filter $\hat{B}$ and $\bar{B}$ contain the belief states detected by GPT-3 and auxiliary generator and are complementary. We thus combine them and filter out the *over-generated* slots whose values couldn't be matched in the user utterance, resulting in the final belief state B.

The auxiliary generator and slot-value match filter are used jointly to automatically detect and correct annotation errors, mitigating the *de-generation* and *over-generation* issues. Taking the example in Figure 5 for illustration, the auxiliary model detects the correct belief state (*hotel, area, don't care*), which is missing in $\hat{B}$. On the contrary, the slot value *cheap* of the slot *hotel-pricerange* couldn't be detected in the user utterance, and thus is removed from the belief state. The resulting final belief state B is used to automatically retrieve the DB entry Q from a pre-defined database.

As for the system side, GPT-3 can generate the dialog act $\hat{A}$ by concatenating the generated user utterance U and belief state B with the prompt. We also utilize the auxiliary TOD model $\mathcal{M}$ to generate its prediction $\bar{A}$ conditioned on the dialogue context X and the user utterance U, revised belief states B, and DB query result Q. To ensure some dialogue logic is followed and the database queried result is taken into account, we write some rules to filter out invalid dialog acts and decide the final dialog act annotation A, which is then concatenated with the prompt to continue generating the system response S.

Note that GPT-3's generation is acceptable in most cases without requiring revision and we cannot guarantee that all the errors can be detected and corrected (we list the frequency in Appendix E). However, the automatic revision on the fly is still important, as GPT-3 tends to imitate the errors in following turns. Therefore, each revision can not only correct the current error, but also avoid numerous potential errors in the following turns.

3.5 Turn-level Generation for DST

For the dialogue state tracking (DST) task, the belief states at each step are an accumulation of previous steps. Any errors from the earlier steps will be propagated to the later steps. In addition, when focusing on the DST task and the belief state annotation, it is not efficient to generate them along with the whole dialogue and other annotations. To avoid error accumulation and unnecessary cost, we propose a method that only generates the user utterance and the corresponding belief states annotation at a turn level.

As shown in Figure 6, for each turn of the seed dialogue, we will generate a new user turn with turn-level belief states and user utterances as an alternative to the original turn to form a new dialogue. To preserve the consistency of dialogue flow, we generate turn-level belief states according to the dialog act of the last turn:

- *Request*: this dialog act means that the system is requesting user's requirements on some attributes (slots), and the user is expected to answer the question by mentioning values for requested slots. We thus generated new turn-level belief states by selecting some or all of the requested slots in the dialog act and adding some other unmentioned slots.
- *Reqmore*: this dialog act means the system asks the user whether he wants service for other domains. Under this circumstance, we select an unmentioned domain and randomly select several slots in this domain to form the new turn-level belief state.

For the other dialog acts, we can randomly select unmentioned slots in the current domain. Some examples of these three kinds of dialog acts are provided in Appendix as Table 10. Given the new belief state, we prompt GPT-3 to generate the corresponding user utterance, which is then verified

and revised as described in Section 3.4. As for in-context examples, we only need to sample user turns instead of whole dialogues, which largely reduces the length of prompts and, thus, the generation cost.

Similarly, Li et al. (2020) proposed a method that can only substitute the slot values of turn-level belief states to form new belief states (*Value Substitution*), and trains GPT-2 to generate the corresponding user utterance. Compared with our work, it cannot generate more diverse belief states, i.e., slot combinations. In addition, it requires a large set of data to train the utterance generator, which is not available in low-resource settings.

4 Experiments

4.1 Experimental Setup

Seed Dataset We implemented our method on the MultiWOZ (Budzianowski et al., 2018) dataset, which consists of 8,438 training, 1,000 validation, and 1,000 test dialogues across 7 domains. As annotation errors exist in the original dataset, we conduct experiments on a cleaner version MultiWOZ2.3 (Han et al., 2021). To simulate the challenging low-resource scenarios, we use 1% (85/8438), 5% (422/8438), and 10% (843/8438) as the seed training dataset and adopt the standard val/test set for evaluation. We also simulate 422 and 843 dialogues given the full training dataset to evaluate its effectiveness as a data augmentation method.

Simulated Dataset We select the largest version of GPT-3 API *text-davinci-002* and use the top-p decoding, where $p = 0.7$. When generating the user goal, we limit the maximum number of requested domains in a dialogue to 4 and the maximum of slots in each domain to 6. We stop generating a dialogue if the number of turns exceeds 12. We use two in-context examples for all generations. More details are provided in Appendix B. PPTOD trained on the seed dataset is utilized as the auxiliary model for automatic verification and revision.

Cost Comparison MultiWOZ dataset creation required 1,249 workers and cost $30k (and around another $20k for postprocessing) (Budzianowski et al., 2018). Assuming a minimum hourly wage of $8, the whole process would take up to 3,750 work hours. In comparison, our approach doesn't require either *human involvement* or *parameter update*, except for the minimal efforts in collecting the small seed dataset and training an auxiliary model on it. The cost and time are thus mainly derived from GPT-3 API call.[2] Averagely, generating a dialogue using GPT-3 only cost **$0.52**. Using other open-resourced LLMs such as OPT-175B (Zhang et al., 2022) can avoid the cost and make our method almost free. Each dialogue can be generated within a few seconds, meaning we can create a large-scale dataset such as MultiWOZ within several hours, which largely shortens the time for dataset creation.

Evaluation Metric To assess the quality of the simulated dialogues, we evaluate the performance of models trained on these simulated dialogues on two benchmark TOD tasks: (1) end-to-end dialogue modeling (E2E) and (2) dialogue state tracking (DST). For E2E evaluation, we use the metrics defined in MultiWOZ (Budzianowski et al., 2018): *Inform*, *Success*, *BLEU*, and an overall metric *Combined Score*: $BLEU + 0.5 \times (Inform + Success)$. For DST evaluation, we report the joint accuracy.

Baselines We select the following three recent end-to-end TOD models as baseline models: SimpleTOD (Hosseini-Asl et al., 2020), MinTL (Lin et al., 2020), and PPTOD (Su et al., 2021). These three models are all based on pre-trained transformers. SimpleTOD is initialized with GPT-2_{small}, while MinTL and PPTOD are initialized with $T5_{\text{small}}$. PPTOD has also been pretrained on heterogeneous dialogue corpus, making it more powerful in low-resource settings. These three models are all capable of performing end-to-end dialogue modeling tasks and DST tasks. We also experiment on a classic DST model TRADE (Wu et al., 2019). For a fair comparison, we use the delexicalized system response in the same format and the evaluation script as in (Zhang et al., 2020; Lin et al., 2020; Su et al., 2021) for E2E evaluation on all these models. During inference, we didn't use any oracle information. For DST evaluation, we use lexicalized utterances. We use the default hyperparameters in their original implementations.

4.2 Experimental Results

4.2.1 End-to-end Dialogue Modeling

We here investigate a realistic question when building a TOD system for a new task or domain: *would our method be a good alternative to crowdsourcing in expanding a small corpus of dialogue data?* To answer this question, we combine the seed dataset with (1) simulated dialogues with our method from the seed dataset (Sim.); (2) human-generated dia-

[2] https://openai.com/api/pricing/

Seed data	Augmented data	SimpleTOD				MinTL-T5				PPTOD				Avg. Improv.
		I	S	B	C	I	S	B	C	I	S	B	C	
1% (85)	Base	37.94	25.53	6.40	38.13	56.81	40.38	12.16	60.76	55.36	38.44	12.25	59.14	-
	+Orig.(85)	46.45	33.73	**6.96**	47.05	64.93	50.20	12.37	70.13	60.26	**44.94**	**12.85**	65.45	16.50%
	+Sim.(85)	**49.65**	**36.74**	6.90	**50.09**	**69.44**	**50.30**	**12.46**	**72.33**	**68.67**	44.54	12.82	**69.43**	**22.60%**
5% (422)	Base	55.96	42.44	7.66	56.86	74.05	60.42	14.71	82.70	68.17	53.55	13.58	74.44	-
	+Orig.(422)	57.96	46.85	**8.04**	60.44	72.24	60.42	**14.91**	81.24	72.07	**61.26**	**15.94**	**82.61**	5.17%
	+Sim.(422)	**58.96**	**47.35**	7.86	**61.02**	**77.45**	**64.93**	13.98	**85.17**	**74.67**	60.06	14.20	81.56	**6.62%**
10% (843)	Base	57.96	46.85	8.04	60.44	72.24	60.42	14.91	81.24	72.07	61.26	15.94	82.61	-
	+Orig.(843)	59.26	47.45	**8.30**	61.65	78.76	68.74	**15.92**	89.67	**78.48**	**64.46**	**15.22**	**86.69**	5.77%
	+Sim.(843)	**62.86**	**51.75**	8.16	**65.47**	**79.96**	**69.84**	15.41	**90.31**	77.98	63.36	14.39	85.06	**7.48%**
100% (8438)	Base	68.67	61.05	10.21	75.08	80.06	72.85	17.87	94.33	**83.88**	69.47	16.33	93.01	-
	+Sim.(422)	68.38	61.82	**10.32**	75.42	79.46	73.45	18.52	94.98	82.88	70.97	**19.22**	**96.15**	1.57%
	+Sim.(843)	**68.97**	**62.46**	10.21	**75.93**	**80.76**	**74.15**	**18.72**	**96.18**	83.08	**71.77**	18.44	95.87	**2.12%**

Table 1: End-to-end dialogue modeling evaluation on MultiWOZ2.3 (Han et al., 2021), where I, S, B, C stand for the Inform, Success, BLEU, and Combined Score metrics, respectively. Sim. and Orig. stand for simulated and original dialogues. The highest scores are bolded. The average improvements are w.r.t. the combined scores.

Statistics	1%		5%		10%	
	Sim.	Orig.	Sim.	Orig.	Sim.	Orig.
Total dialogs	85	85	422	422	843	843
Total turns	616	599	3510	2778	6634	5617
Total domains	229	147	1076	738	2203	1471
Avg. turns	7.25	7.05	8.32	6.58	7.87	6.66
Avg. domains	2.69	1.73	2.55	1.75	2.61	1.74
Uniq. tokens	550	561	1095	1084	1336	1373
Uniq. 3-grams	3413	4300	10662	13815	15063	22261

Table 2: Comparison of the statistics of simulated dialogues (Sim.) with human generated dialogues from the original MultiWOZ dataset (Orig.).

Model	I	S	B	C
SimpleTOD	49.65	36.74	6.90	50.09
-w/o revision	43.92	29.13	6.21	42.74
MinTL-T5	69.44	50.30	12.46	72.33
-w/o revision	61.68	46.34	11.83	65.84
PPTOD	68.67	44.54	12.82	69.43
-w/o revision	62.41	40.27	11.97	63.31

Table 3: Ablation study on the automatic revision under 1% low-resource setting.

logues from the original dataset, excluding the seed data (Orig.). We then train several representative TOD models with these two datasets and compare their performance. For a fair comparison, we use the same amount of simulated dialogues and original dialogues and the same set of seed data across all setups.

As seen in Table 1, the models trained on simulated dialogues along with the seed dataset perform much better than those only trained on the seed dataset. The performance improvement is especially significant with in the most challenging 1% setting, which can be attributed to the powerful few-shot learning ability of GPT-3. Remarkably, compared with human-generated dialogues, the same amount of our simulated dialogues can lead to even more significant performance improvement in most cases. Our simulated dialogues can still improve performance when all the training data are provided. One can also generate more dialogues for better performance. The results suggest the effectiveness of our method as a data augmentation method. This is expected when we have enough data to select from as in-context examples for GPT-3 and train the auxiliary revision model. As a result, more diverse dialogues and accurate annotations can be simulated.

To understand the observations, we further analyze the statistics of simulated and human-generated dialogues and present them in Table 2. We found that our simulated dialogues have more requested domains and dialogue turns compared with human-generated dialogues, which are controlled by the generated user goals. The much more sub-tasks and sub-conversations in the simulated dialogues improve the model's ability to deal with more complex and challenging multi-domain tasks, thus leading to a higher Inform rate and Success rate. On the contrary, the system responses generated by our method have a comparable amount of unique tokens but fewer 3-grams than the original ones, which explains why the BLEU score is slightly lower than the human-generated dialogues.

Effect of Revision Table 3 shows performances of PPTOD trained with dialogues generated in the presence and absence of the automatic verification and revision under the 1% setting. Without verification and revision, the generated dialogues lead to lower performance improvement in all metrics. This is reasonable as GPT-3 sometimes generates

Model	Base	Augmentation size			
		x1	x2	x3	x4
TRADE	12.98	14.41	15.96	16.60	17.17
SimpleTOD	13.69	16.43	17.81	18.51	19.15
MinTL	23.25	27.30	27.69	28.71	29.27
PPTOD	34.48	36.71	37.74	38.12	38.51

Table 4: DST evaluation with 1% of training data and different sizes of augmented data.

incorrect belief states and dialog acts, declining the model performance. What's worse, GPT-3 will imitate these errors in consecutive turns if not corrected, leading to more significant problems.

4.2.2 Dialogue State Tracking

Next, we investigate the effectiveness of belief state annotations generated by our method as augmented data for DST training in low resources settings. We simulate the low-resource setting using 1% of the MultiWOZ training set as seed data. Table 4 shows the performance of various models trained on different sizes of augmented data. The augmented data can consistently improve the model performance across different models. With the increase of augmented data, the accuracy keeps increasing, though the upward trend is gradually slowing down.

4.3 Human Evaluation

To get a more comprehensive measure of the quality of the simulated data compared with the original human-generated data, we conducted a blind human evaluation study. Three participants with NLP backgrounds are given 50 dialogues simulated from 1% of the training data and another 50 dialogues from the original dataset without knowing their source (simulated or original). Following (Mohapatra et al., 2020), we ask the participants to check the quality of the conversation and annotations of each dialogue turn by answering the following questions: (1) "*Are the utterances grammatically correct?*" (2) "*Is the user utterance fluent and natural?*" (3) "*Is the system response fluent and natural?*" (4) "*Is the belief state annotation consistent with the user utterance?*" (5) "*Is the dialog act consistent with the system response?*". For each question, the participants should answer "yes" or "no". We adopt a majority vote approach to decide the final answer with at least two votes. Table 5 shows the percentage of dialogue turns that satisfy each quality measure (with the answer of "yes").

We find that our method can generate even more grammatically correct conversations than human

Measure	Simulated	Original
Grammar	99.00	96.32
User fluency	94.55	95.22
System fluency	88.61	94.48
User consistency	84.65	96.69
System consistency	73.76	94.12

Table 5: Human evaluation on the simulated dialogues and original dialogues w.r.t. the percentage of dialogue turns that satisfy each quality measure.

crowdworkers leveraging the strong generation ability of GPT-3. The generated user utterances are comparably fluent and natural compared with original human-generated ones. In contrast, the generated system responses are not that fluent. We suspect this is because the system responses are delexicalized, which is more challenging for GPT-3 to understand and imitate. As for the annotation quality, although humans generate the original dialogues, annotation errors still exist, which suggests the difficulty of creating a clean task-oriented dialogue dataset. As expected, our generated dialogue data has more annotation errors than human-generated ones. The gap between simulated dialogues and the original dialogues on the user consistency is not as large as that on the system consistency because the belief states are easier for GPT-3 to understand and generate compared with the more complex dialog acts. Overall, the conversations generated by our method have comparable quality to human-generated ones. However, the generated annotations are not as accurate as human annotations. Satisfyingly, the noisy annotation still introduces considerable performance improvement.

5 Conclusion

In this paper, we propose a dialogue simulation method based on large language model in-context learning to automate dataset creation. Our proposed method can generate dialogues and annotations given only a few seed dialogues. The simulation process requires zero or minimum *human involvement* and *model training*, making our method much more cost-efficient and time-saving than crowdsourcing. Human and automatic evaluations demonstrate that the simulated dialogues have comparable quality to human-generated ones, which shows the potential of our method as an alternative to crowdsourcing in dialogue dataset creation.

Limitations

In this paper, we investigate ways to leverage in-context learning with GPT-3 to automatically generate high-quality task-oriented dialogues for building dialogue systems. Although our method can already generate high-quality dialogues without requiring human involvement, there are still some limitations in real-world applications. GPT-3 has a deficiency in lack of reliability and is inevitable to generate some unexpected data even with automatic revision. Human review and revision are still necessary to ensure the annotation is completely correct. However, it is challenging as the revision at each step will influence the latter steps. Therefore, an effective and efficient human and machine collaboration approach is our future direction. In addition, as the dialogues along with annotations are very lengthy, it is essential to reduce their length to lower the generation cost and enable the use of more in-context examples.

Ethics Statement

Our proposed method instructs LLMs to generate dialogues for building dialogue systems. However, LLMs such as GPT-3 (Brown et al., 2020) are observed to generate toxic or biased text (Brown et al., 2020; Lucy and Bamman, 2021; Chan, 2022). Although a new version of GPT-3 called InstructGPT has been released, trying to reduce these toxic languages, the issue hasn't been sufficiently addressed. Thus, automatic filtering or human review methods is necessary to exclude some parts of training data to avoid the models generating undesirable responses containing toxicity and bias from the simulated dialogues.

Acknowledgements

The authors would like to thank the anonymous reviewers for their thoughtful comments. This research was partly sponsored by the DARPA PTG program (HR001122C0009) and Alexa Prize Taskbot Challenge. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of funding agencies.

References

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In SIGDIAL Conference.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. ArXiv, abs/2005.14165.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. arXiv preprint arXiv:1810.00278.

Anastasia Chan. 2022. Gpt-3 and instructgpt: technological dystopianism, utopianism, and "contextual" perspectives in ai ethics and industry. AI and Ethics, pages 1–12.

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

Mihail Eric, Lakshmi. Krishnan, François Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In SIGDIAL Conference.

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 NLPCC.

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, SIGdial 2020, 1st virtual meeting, July 1-3, 2020, pages 35–44. 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. arXiv preprint arXiv:2203.08568.

John F Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. ACM Transactions on Information Systems (TOIS), 2(1):26–41.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. ArXiv, abs/1907.07421.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers. In International Conference on Learning Representations.

Zekun Li, Hong Wang, Alon Albalak, Yingrui Yang, Jing Qian, Shiyang Li, and Xifeng Yan. 2022. Making something out of nothing: Building robust task-oriented dialogue systems from scratch.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. 2021. Jurassic-1: Technical details and evaluation. White Paper. AI21 Labs.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. ArXiv, abs/2009.12005.

Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2022. Wanli: Worker and ai collaboration for natural language inference dataset creation. arXiv preprint arXiv:2201.05955.

Li Lucy and David Bamman. 2021. Gender and representation bias in GPT-3 generated stories. In Proceedings of the Third Workshop on Narrative Understanding, pages 48–55, Virtual. Association for Computational Linguistics.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. arXiv preprint arXiv:2110.08118.

Biswesh Mohapatra, Gaurav Pandey, Danish Contractor, and Sachindra Joshi. 2020. Simulated chats for task-oriented dialog: Learning to generate conversations from instructions. ArXiv, abs/2010.10216.

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

Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. 2013. Using of jaccard coefficient for keywords similarity. In Proceedings of the international multiconference of engineers and computer scientists, volume 1, pages 380–384.

Alexandros Papangelis, Yi-Chia Wang, Piero Molino, and Gökhan Tür. 2019. Collaborative multi-agent dialogue model training via reinforcement learning. In SIGdial.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Lidén, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. Transactions of the Association for Computational Linguistics, 9:807–824.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint arXiv:2112.11446.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. ArXiv, abs/1909.05855.

Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. 2005. Let's go public! taking a spoken dialog system to the real world. In in Proc. of Interspeech 2005. Citeseer.

Pararth Shah, Dilek Hakkani-Tur, Bing Liu, and Gökhan Tür. 2018a. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers), pages 41–51.

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018b. Building a conversational agent overnight with dialogue self-play. arXiv preprint arXiv:1801.04871.

Richard Shin and Benjamin Van Durme. 2021. Few-shot semantic parsing with language models trained on code. arXiv preprint arXiv:2112.08696.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. arXiv preprint arXiv:2201.11990.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. arXiv preprint arXiv:2109.14739.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam M. Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, Yaguang Li, Hongrae Lee, Huaixiu Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, I. A. Krivokon, Willard James Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Hartz Søraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravindran Rajakumar, Alena Butryna, Matthew Lamm, V. O. Kuzmina, Joseph Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. 2022. Lamda: Language models for dialog applications. ArXiv, abs/2201.08239.

Bo-Hsiang Tseng, Yinpei Dai, Florian Kreyssig, and Bill Byrne. 2021. Transferable dialogue systems and user simulators. arXiv preprint arXiv:2107.11904.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021a. Want to reduce labeling cost? gpt-3 can help. arXiv preprint arXiv:2108.13487.

Zirui Wang, Adams Wei Yu, Orhan Firat, and Yuan Cao. 2021b. Towards zero-label language learning. arXiv preprint arXiv:2109.09193.

J. Williams, Antoine Raux, Deepak Ramachandran, and Alan W. Black. 2013. The dialog state tracking challenge. In SIGDIAL Conference.

Jason D Williams, Matthew Henderson, Antoine Raux, Blaise Thomson, Alan Black, and Deepak Ramachandran. 2014. The dialog state tracking challenge series. AI Magazine, 35(4):121–124.

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 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 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.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. Ubar: Towards fully end-to-end task-oriented dialog systems with gpt-2. In AAAI.

Dian Yu, Luheng He, Yuan Zhang, Xinya Du, Panupong Pasupat, and Qi Li. 2021. Few-shot intent classification and slot filling with retrieved examples. arXiv preprint arXiv:2104.05763.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In AAAI.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. arXiv preprint arXiv:1902.08858.

Appendix

A Implementation Details

We conduct experiments on MultiWOZ2.3[3], which corrects annotations in dialogue acts from dialogue states and adds co-reference annotation. The ontology information of MultiWOZ2.3 dataset is presented in Table 6. There are 12 dialog acts and 24 slots in total. We use the released source codes and default hyperparameters of the baseline models SimpleTOD[4], MinTL[5], PPTOD[6], and TRADE[7]. For a fair comparison, we use the same set of training data and dev/test data across all these models. In addition, we use the same evaluation script provided in PPTOD for all these models. The delexicalized utterances are used for end-to-end evaluations, while lexicalized utterances are used for DST evaluation. All the experiments are run on a server with 8 NVIDIA RTX A6000 GPUs.

B Dialogue Generation Details

We select the most powerful version of GPT-3 API *text-davinci-002*[8]. The temperature is set as 0.7 and the frequency penalty as 1.0 to encourage the generation of more diverse data and avoid the repetition of the in-context examples. When generating the dialogues, we set the maximum number of turns as 12 to avoid endless generations.

We test three strategies to generate the user goals and prompt: (1) *Random Sampling*; (2) *Value Substitution*; and (3) *Combination*. For *Random Sampling* strategy, the sampling distribution of the number of domains, the minimal number of slots in each domain, and the maximum number of slots in each domain are presented in Table 7. For the *Random Sampling* strategy, we keep all the slots and substitute the slots values of the original user goal to create new goals. For the *Combination* strategy, we set the temperature in Equation 2 as 0.2 when sampling similar dialogues from the seed dataset.

C User Turn Generation

We generate user turns for DST augmentation based on the human-generated seed dialogues. For the turns from the original dialogues, we generate augmented user turns and concatenate them with the previous turns to create a new training sample. We generate the user turns based on the dialog acts of the last system turn in the context. Suppose the last system turn contains the *request* act. In that case, we will randomly select at least one slot from the requestable slots and another at least two slots from the unmentioned slots in the current domain to form the belief state of the augmented user turn. If the last system turn contains the *reqmore* act, we will randomly select one unmentioned domain and at least one, at most four slots in the domain. For the other cases, we will randomly drop at least one slot from the original belief state and add at least one unmentioned slot. Having selected the slots in the belief state, we will randomly select a possible slot value for each slot to create the concrete belief state for the augmented user turn.

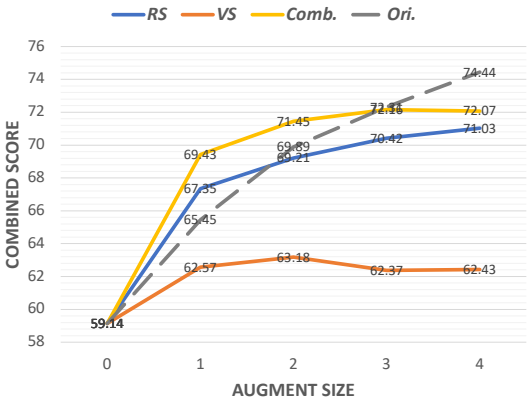

Figure 7: The combined score of PPTOD trained with dialogues simulated with different methods and augmentation size with 1% training data as the seed data.

D Goal Generation Methods

As mentioned in Section 3.2, we investigate the following methods to generate user goals and retrieve dialogues as in-context examples: (1) *Random Sampling (RS)*, (2) *Value Substitution (VS)*, and (3) *Combination (Comb.)*. We show the performance of PPTOD trained on dialogues simulated with different goal generation methods and varying augmentation sizes in Figure 7. We also show the model performance trained on the same amount of original dialogues in the dashed line as a baseline (Ori.).

We can see that the *VS* method performs worst, as it can only change the slot values. Thus, GPT-3 tends to simply replicate the in-context examples. As the *Combination* method can cover most of the information needed in the target dialogues and en-

[3]https://github.com/lexmen318/MultiWOZ-coref
[4]https://github.com/salesforce/simpletod
[5]https://github.com/zlinao/MinTL
[6]https://github.com/awslabs/pptod
[7]https://github.com/jasonwu0731/trade-dst
[8]https://openai.com/api/pricing/

Table 6: Full ontology for all domains in MultiWOZ2.3 (Han et al., 2021). The upper script indicates which domains it belongs to. *: universal, 1: restaurant, 2: hotel, 3: attraction, 4: taxi, 5: train, 6: hospital, 7: police. The table is adapted from (Budzianowski et al., 2018).

dialog acts	inform* / request* / select^{1235} / recommend/^{123} / nooffer^{1235} / offerbook^{125} / offerbooked^{125} / nobook^{12} / welcome* / greet* / bye* / reqmore*
slots	address^{12367} / postcode^{1367} / phone^{123467} / name^{123} / area^{123} / pricerange^{12} / type^{23} / internet^{2} / parking^{2} / stars^{2} / departure^{45} / destination^{45} / leave^{45} / arrive^{45} / people^{125} / reference^{1235} / id^{5} / price^{5} / time^{15} / department^{6} / day^{125} / stay^{2} / car^{4} / food^{1}

#domains	Min #slots	Max #slots	Probability
1	4	6	0.3
2	3	5	0.6
3	2	5	0.1

Table 7: The sampling distribution of user goals w.r.t. the number of domains, the minimal and maximum number of slots in each domain for the *Random Sampling* strategy.

courage GPT-3 to generate diverse data, it achieves the best performance. However, it is harder to improve further with the increased augmentation size as there are limited seed dialogues and, thus, their combinations. On the contrary, as the *RS* method can generate any user goals, the model performance keeps increasing. However, there is still an upper bound of performance, which depends on the number of provided seed dialogues.

E Automatic Revision Frequency

To investigate how often the data generated by GPT-3 need to be revised and how many errors can be corrected by our method, we randomly selected 20 dialogues simulated under the 1% low-resource setting and manually checked the annotations. In a total of 170 turns, GPT-3 generates incorrect belief state annotations in 31 turns (18 de-generation / 13 over-generation). The auxiliary generator corrects GPT-3 in 13 out of 18 de-generated turns, while the slot-value match filter corrects in 10 out of 13 over-generated turns. Finally, the revised belief states in only 11 turns are still incorrect (6.47%), while that for dialog act is 19 (11.18%).

F Dialogue Generation Example

An example of a complete prompt is shown in Table 8, given which the controllable generation process of the dialogue is presented in Table 9. The process is fully automated.

G User Turn Generation Example

We provide the illustration of three different types of augmented turns for DST in Table 10, and the prompt and generation process of it in Table 11.

Task Description	The following are conversations between a user and an assistant. The assistant can help the user to find things that satisfy his requirements. Try to speak differently in different conversations.
Example 1 (PMUL1576)	**Instruction1:** You are going to book a train, and your requirements for the train are ([train] destination is leicester , departure is cambridge , leave is 08:45 , day is saturday , arrive is dontcare). You also want to book a hotel, and your requirements for the hotel are ([hotel] name is cityroomz , stay is 4 , day is tuesday , people is 8). Make sure you get the booking information once booked. **Conversation1:** **User**([train] destination is leicester , departure is cambridge , leave is 08:45): i really need to get out of cambridge ! can you find me a train to leicester ? sometime after 08:45 because i like to sleep in . **Assistant**([train] [request] day): i would be happy to help with your request , what day will you be leaving ? **User**([train] day is saturday): i 'll be leaving this place on saturday . **Assistant**([train] [request] arrive): what time would you like to arrive by ? **User**([train] arrive is dontcare): it does not matter . **Assistant**([train] [inform] destination arrive leave [offerbook]): there is a train that leaves at [value_leave] and arrive in leiester at [value_arrive] . would you like me to book it for you ? **User**([train]): no thank you . what is the cost of the ticket ? **Assistant**([train] [inform] price [general] [reqmore]): the ticket price is [value_price] . can i be of further assistance ? **User**([train]): i also need the travel time and arrival time please . **Assistant**([train] [inform] arrive time): arrival time is [value_arrive] and travel time is [value_time] . **User**([hotel] name is cityroomz): i ' m also looking for a particular hotel . its name is called cityroomz . **Assistant**(): sure , what kind of information do you need ? **User**([hotel] stay is 4 , day is tuesday , people is 8): i would like to to book it for 8 people and 4 nights starting from tuesday **Assistant**([hotel] [offerbooked] day people reference stay [general] [reqmore]): i have booked a room for [value_people] for [value_stay] nights beginning on [value_day] . your reference number is [value_reference] . is there anything else i can help you with today ? **User**([general]): not at this time . thank you . **Assistant**([general] [bye]): have a fantastic day , goodbye .
Example 2 (SNG0955)	**Instruction2:** You are going to book a hotel, and your requirements for the hotel are (hotel] pricerange is expensive , area is east , parking is yes). Make sure you get the booking information once booked. **Conversation2:** **User**(([hotel] pricerange is expensive): i need a place to stay that does n't have to have internet and is in the expensive price range please . **Assistant**(([hotel] [inform] choice [request] area): i have [value_choice] different ones all around town . did you prefer to stay in a certain area ? **User**(([hotel] area is east): yes , on the east side please . **Assistant**([hotel] [inform] name [offerbook]): [value_name] meets your needs , would you like to book it ? **User**([hotel] parking is yes): does it have free parking ? **Assistant**([hotel] [inform] parking): yes , it does . **User**([hotel]): i ' m not ready to book . can you just tell me what the address is ? oh , and how many stars is it ? **Assistant**([hotel] [inform] type stars address [general] [reqmore]): sure . it is a [value_stars] star [value_type] and the address is [value_address] . anything else ? **User**([general]): no , that is all . thanks . **Assistant**([general][bye]): you are welcome ! please contact us if you would like to make a reservation in the future .
Target Dialogue Prompt	**Instruction3:** You are going to book a hotel, and your requirements for the hotel are (([hotel] area is south , stay is 5 , people is 4). You also want to book a train, and your requirements for the train are ([train] destination is birmingham new street , arrive is 13:06). Make sure you get the booking information once booked. **Conversation3:**

Table 8: Example of a complete prompt which contains a task description, two in-context examples (PMUL1576 and SNG0955 are from the MultiWOZ dataset), and the user goal (*instruction*) of the target dialogue. The user goal, belief states, and dialog acts are in red color. The user utterances are in blue, while the system responses are in green. Given the prompt, we can generate the *conversation* of the target dialogue as shown in Table 9.

Turn 1	Original user turn generation	**User**([hotel] area is south , stay is 5 , people is 4): i need a hotel in the south side please .
	GPT-3 generated belief state	<hotel, area, south> , <hotel, book stay, 5> , <hotel, book people, 4>
	Revised belief state	<hotel, area, south> , ~~<hotel, book stay, 5>~~ , ~~<hotel, book people, 4>~~, <hotel, type, hotel>
	Revised user turn generation	**User**([hotel] area is south , type is hotel): i need a hotel in the south side please .
	DB query result	[db_1]
	Revised dialog act	<hotel, inform, area>, <hotel, inform, name>, <hotel, inform, internet>, <hotel, inform, parking>, <hotel, inform, type>, <hotel, inform, stars>, <hotel, offerbook, none>
	Revised system turn generation	**Assistant**([hotel] [inform] area name internet price parking type stars [offerbook]): the [value_name] hotel is in the south side and it has [value_internet] . the price is [value_price] per night and it has [value_parking] . it is a [value_stars] star hotel .
Turn 2	Original user turn generation	**User**([hotel] stay is 5 , people is 4): i would like to to book it for 4 people and 5 nights.
	GPT-3 generated belief state	<hotel, book stay, 5> , <hotel, book people, 4>
	Revised belief state	<hotel, book stay, 5> , <hotel, book people, 4>
	Revised user turn generation	**User**([hotel] stay is 5 , people is 4): i would like to to book it for 4 people and 5 nights.
	DB query result	[db_1]
	Revised dialog act	<hotel, offerbooked, reference>, <general, reqmore, none>
	Revised system turn generation	**Assistant**([hotel] [offerbooked] reference [general] [reqmore]): your booking reference number is [value_reference] .
Turn 3	Original user turn generation	**User**([train] destination is birmingham new street , arrive is 13:06): i need a train to birmingham new street station that arrives by 13:06 please .
	GPT-3 generated belief state	<train, destination, birmingham new street> , <train, arrive, 13:06>
	Revised belief state	<train, destination, birmingham new street> , <train, arrive, 13:06>
	Revised user turn generation	**User**([train] destination is birmingham new street , arrive is 13:06): i need a train to birmingham new street station that arrives by 13:06 please .
	DB query result	[db_3]
	Revised dialog act	<train, request, day>, <train, request, departure>
	Revised system turn generation	**Assistant**([train] [request] day departure): what day will you be leaving and what is your departure station ?
Turn 4	Original user turn generation	**User**([train] day is saturday , departure is cambridge): i will be leaving this saturday from cambridge station .
	GPT-3 generated belief state	<train, day, saturday> , <train, departure, cambridge>
	Revised belief state	<train, day, saturday> , <train, departure, cambridge>
	Revised user turn generation	**User**([train] day is saturday , departure is cambridge): i will be leaving this saturday from cambridge station .
	DB query result	[db_3]
	Revised dialog act	<train, inform, arrive>, <train, inform, id>, <train, inform, leave>, <train, offerbook, none>
	Revised system turn generation	**Assistant**([train] [inform] arrive id leave [offerbook]): the train arrives at [value_arrive] and the id is [value_id] . would you like me to book it for you ?
Turn 5	Original user turn generation	**User**([train]): no thank you . what is the cost of the ticket ?
	GPT-3 generated belief state	<train, none, none>
	Revised belief state	<train, none, none>
	Revised user turn generation	**User**([train]): no thank you . what is the cost of the ticket ?
	DB query result	[db_3]
	Revised dialog act	<train, inform, price>, <general, reqmore, none>
	Revised system turn generation	**Assistant**([train] [inform] price [general] [reqmore]): the ticket price is [value_price] . can i be of further assistance ?
Turn 6	Original user turn generation	**User**([general]): that is all for now . thanks
	GPT-3 generated belief state	<general, none, none>
	Revised belief state	<general, none, none>
	Revised user turn generation	**User**([general]): that is all for now . thanks
	DB query result	[db_nores]
	Revised dialog act	<general, bye, none>
	Revised system turn generation	**Assistant**([general] [bye]): you are welcome , please contact us if you need anything else .

Table 9: The controllable dialogue generation process of a dialogue given the prompt in Table 8. For each turn, GPT-3 will first generate the belief state and the user utterance. We parse the belief state, which is then verified and revised automatically. We replace the original generated belief state with the revised belief state in the user turn generation. Then the revised user turn generation will be used to query the database and concatenated with the dialogue context to continue the generation. As for the system turn, we use the revised dialog act, conditioned on which we prompt GPT-3 to generate the system response. Note that the user goal of the target dialogue is allowed to change during the generation. We will keep the updated user goal instead of the original one in the prompt (*instruction3*), which is only used to initiate the generation of the target dialogue.

Example 1 (*request*)	
Context (SNG01856, turn 1)	**User**: am looking for a place to to stay that has cheap price range it should be in a type of hotel **Assistant**: i would be happy to help with your request , what day will you be leaving ?
Last dialog act (turn 1)	<hotel, request, area>
Original user turn (turn 2)	**Belief state**: <hotel, parking, yes>, <hotel, pricerange, cheap> **User**: no , i just need to make sure it is cheap . oh , and i need parking
Augmented user turn (turn 2)	**Belief state**: <hotel, star, 1>, <hotel, area, west> **User**: a 1 star hotel in the west .
Example 2 (*others*)	
Context (SNG01856, turn 1-2)	**User**: am looking for a place to to stay that has cheap price range it should be in a type of hotel **Assistant**: i would be happy to help with your request , what day will you be leaving ? **User**: no , i just need to make sure it is cheap . oh , and i need parking **Assistant**: i found [value_choice] [value_price] [value_type] for you that include -s parking . do you like me to book it ?
Last dialog act (turn 2)	<hotel, inform, price>, <hotel, inform, choice>, <hotel, inform, parking>, <hotel, inform, type>, <hotel, offerbook, none>
Original user turn (turn 3)	**Belief state**: <hotel, book stay, 3>, <hotel, book day, tuesday>, <hotel, book people, 6> **User**: yes , please . 6 people 3 nights starting on tuesday .
Augmented user turn (turn 3)	**Belief state**: <hote, book people, 8>, <hotel, stars, 3>, <hotel, book stay, 2>, <hotel, book day, saturday> **User**: please book me a room for 8 people on saturday . we will be staying for 2 nights and would like a 3-star hotel .
Example 3 (*reqmore*)	
Context (SNG01856, turn 1-3)	**User**: am looking for a place to to stay that has cheap price range it should be in a type of hotel **Assistant**: i would be happy to help with your request , what day will you be leaving ? **User**: no , i just need to make sure it is cheap . oh , and i need parking **Assistant**: i found [value_choice] [value_price] [value_type] for you that include -s parking . do you like me to book it ? **User**: how about only 2 nights . **Assistant**: booking was successful . reference number is : [value_reference] . anything else i can do for you ?
Last dialog act (turn 3)	<hotel, offerbooked, reference>, <general, reqmore, none>
Original user turn (turn 4)	**Belief state**: <general, none, none> **User**: no , that will be all . goodbye .
Augmented user turn (turn 4)	**Belief state**: <taxi, destination, avalon> **User**: i need a taxi to avalon .

Table 10: Example of three augmented user turns for DST in **SNG01856** (MultiWOZ). These three turns are augmented according to different types of dialog acts in the last turn, which are *request*, *others*, and *reqmore*, respectively. We show the prompt and generation of Example 2 in Table 11.

Task Description	Answer the assistant's question on each feature you require when booking a train. Also mention no preference on a feature when your requirement on it is "dontcare". **Features**: people: number of people for the hotel booking; type: what is the type of the hotel, guesthouse, guest house, or hotel; stay: length of stay at the hotel; name: name of the hotel; day: day of the hotel booking; stars: star rating of the hotel;
Example 1 (PMUL1576, turn 3)	**Assistant**: what is your requirement on day? **User**([hotel] day is friday): yes , please book me a room for friday .
Example 2 (SNG1006, turn 2)	**Assistant**: what is your requirement on type, name, stay, day, people? **User**([hotel] type is hotel , name is gonville hotel , stay is 4 , day is saturday , people is 6): okay , i would like to book a room at the gonville hotel for 4 nights . there will be 6 people and we will be arriving on saturday .
Target Turn Prompt	**User**([hotel] people is 8 , stars is 3 , stay is 2 , day is tuesday):
Target Turn Completion	**User**([hotel] people is 8 , stars is 3 , stay is 2 , day is tuesday): please book me a room for 8 people on tuesday . we will be staying for 2 nights and would like a 3-star hotel .

Table 11: Example of a complete prompt for the generation of Example 2 in Table 10. The task description contains the description of the mentioned slots to help GPT-3 better understand. We sample some turns from seed dialogues instead of the whole dialogues. Given the target turn-level belief state, GPT-3 is able to generate the user utterance that expresses the user goal.