A Generative User Simulator with GPT-based Architecture and Goal State Tracking for Reinforced Multi-Domain Dialog Systems

Hong Liu[1,3], **Yucheng Cai**[1,3], **Zhijian Ou**[1,3]*, **Yi Huang**[2,3], **Junlan Feng**[2,3]
[1]Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University, Beijing, China
[2]China Mobile Research Institute, Beijing, China
[3]Tsinghua University-China Mobile Communications Group Co., Ltd. Joint Institute, Beijing, China
{liuhong21,cyc22}@mails.tsinghua.edu.cn,
ozj@tsinghua.edu.cn,
{huangyi,fengjunlan}@chinamobile.com

Abstract

Building user simulators (USs) for reinforcement learning (RL) of task-oriented dialog systems (DSs) has gained more and more attention, which, however, still faces several fundamental challenges. First, it is unclear whether we can leverage pretrained language models to design, for example, GPT-2 based USs, to catch up and interact with the recently advanced GPT-2 based DSs. Second, an important ingredient in a US is that the user goal can be effectively incorporated and tracked; but how to flexibly integrate goal state tracking and develop an end-to-end trainable US for multi-domains has remained to be a challenge. In this work, we propose a generative user simulator (GUS) with GPT-2 based architecture and goal state tracking towards addressing the above two challenges. Extensive experiments are conducted on MultiWOZ2.1. Different DSs are trained via RL with GUS, the classic agenda-based user simulator (ABUS) and other ablation simulators respectively, and are compared for cross-model evaluation, corpus-based evaluation and human evaluation. The GUS achieves superior results in all three evaluation tasks.

1 Introduction

Task-oriented dialog (TOD) systems are mainly designed to help users accomplish specific goals, such as finding restaurants or booking flights. The dialog system (DS) usually consists of several modules - dialog state tracking (DST), database querying (DB), dialog policy (DP) and natural language generation (NLG). Recent studies recast these modules all as conditional generation of tokens and build on some pretrained language model (PLM) such as GPT-2 (Radford et al., 2019) as the backbone. Fine-tuning PLM over annotated dialog datasets via supervised learning (SL) has shown state-of-the-art results (Hosseini-Asl et al., 2020; Li et al., 2020; Kulhánek et al., 2021; Yang et al., 2021; Lee, 2021), thanks to the powerful generation ability of PLMs.

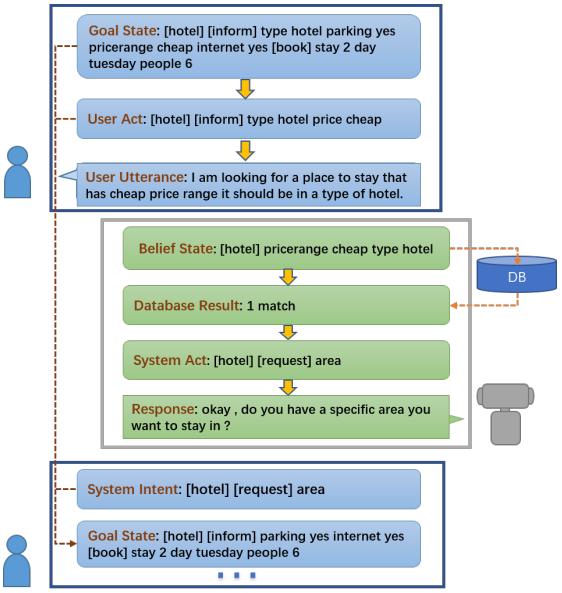


Figure 1: The information flow in a task-oriented dialog. Domains and intents are enclosed by square brackets.

However, supervised trained agents could become biased by the annotations, and it has long been recognized that reinforcement learning (RL) could be applied to policy learning for the agent (Young et al., 2013), which aims at goal-directed learning from interaction between the dialog agent and the user. Interaction with human users is expensive and time-consuming in practice. Therefore, an alternative approach, building user simulators (USs), has gained more and more attention, which, however, still faces several fundamental challenges.

First, note that the recent research on building dialog agents has been significantly advanced with the end-to-end trainable generative approach based on PLMs such as GPT-2. However, prior work on user simulators are mostly LSTM-based, not utilizing any PLMs, as reviewed in Table 1. It is unclear whether we can leverage PLMs to design, for ex-

*Corresponding author. The code is released at https://github.com/thu-spmi/GUS

Proceedings of the Towards Semi-Supervised and Reinforced Task-Oriented Dialog Systems (SereTOD), pages 85 - 97

ample, GPT-2 based[1] user simulators, to catch up and interact with the GPT-2 based dialog agents. This has not ever been systematically examined, to the best of our knowledge. We leave detailed discussion to Related Work section, where we review prior work on USs from a number of important features in building USs.

Second, an important ingredient in a US is that the user goal can be incorporated and tracked. Task-oriented dialog systems are characterized by a user goal, which is composed of user constraints and requests. The user goal ensures that the user behaves in a consistent, goal-directed manner, and the system agent is considered successful if it is able to fulfill the user goal by the end of a dialog session. Thus, it is desirable for the US to track the completion process of the goal explicitly (which we call goal state tracking in this paper), as did in the classic agenda-based user simulator (ABUS) (Schatzmann et al., 2007). However, the goal state tracking process is overlooked in later data-driven USs (Asri et al., 2016; Gür et al., 2018; Papangelis et al., 2019), or realized by binary vectors (Kreyssig et al., 2018; Lin et al., 2021; Tseng et al., 2021), or only works at the semantic level (Takanobu et al., 2020). How to flexibly integrate goal state tracking and develop an end-to-end trainable US for multi-domains has remained to be a challenge.

In this work, we propose a generative user simulator (GUS) with GPT-2 based architecture and goal state tracking towards addressing the above two challenges in building end-to-end trainable USs for reinforced multi-domain dialog systems. Basically, a US, interacting with a DS in natural languages, needs several modules - natural language understanding (NLU) of system responses, goal state tracking (GST) to refresh the remained constrains and requests that need to send subsequently, user policy (UP), and natural language generation (NLG). The information flow in a task-oriented dialog between a US and a DS is illustrated in Figure 1. In generative user simulator (GUS), we recast these modules in US all as conditional generation of tokens, similar to the recent approach of fine-tuning PLMs such as GPT-2 to build end-to-end trainable generative DSs.

To be specific in this paper, we use the GPT-2 based architecture for GUS to generate user acts and user utterances, and constantly track the goal state according to the user acts and system acts of the previous turn, which is shown in Figure 2. In this work, the definition of goal state is similar to the agenda in ABUS (Schatzmann et al., 2007), which represents a collection of pending user acts that are needed to elicit the information specified in the goal. The maintenance of goal state includes not only removing the completed user acts, but also changing the user goal when the system cannot find a requested entity.

Extensive experiments are conducted on MultiWOZ2.1 (Eric et al., 2020). Different DSs are trained via RL with GUS, ABUS and other ablation simulators respectively, and are compared for cross-model evaluation, corpus-based evaluation and human evaluation. The GUS achieves superior results in all three evaluation tasks.

US	PLM	Goal State Tracking	Cross-model Evaluation	Compared with DS-SL	Natural Lang. Interaction	Multi-Domain
Schatzmann et al. (2007)	N	Y	N	N	N	N
Asri et al. (2016)	N	N	N	N	N	N
Liu and Lane (2017)	N	N	N	Y	Y	N
Gür et al. (2018)	N	N	N	N	N	N
Kreyssig et al. (2018)	N	Y	Y	N	Y	N
Papangelis et al. (2019)	N	N	N	Y	Y	N
Shi et al. (2019)	N	N	Y	N	Y	N
Takanobu et al. (2020)	N	Y	N	Y	N	Y
Lin et al. (2021)	N	Y	Y	N	N	Y
Tseng et al. (2021)	N	Y	N	Y	Y	Y
GUS	Y	Y	Y	Y	Y	Y

Table 1: Comparison of prior different user simulators from a number of important features in building USs. DS-SL denotes dialog system (DS) trained by supervised learning (SL). See Section 2 for detailed meaning of each feature by column.

2 Related Work

Novelty In Table 1, we review prior work on USs from a number of important features in building USs, including whether or not the US is based on any PLMs, the US conducts goal state tracking, the cross-model evaluation (Schatztnann et al., 2005) is conducted to assess the performance of the US, the DS trained via RL with the US is compared to the DS trained via supervised learning, the US and the DS interact in natural languages [2], the US is designed to work for multi-domain dialogs. It is clear from Table 1 that our proposed GUS is distinctive, which represents the first US that possesses all these desirable features, to the best of our knowledge. More discussions are provided in the

[1] It can be seen that the discussion and the proposed method in the remainder of this paper can also be applied to other PLMs such as T5 (Raffel et al., 2020), not limited to GPT-2.

[2] This means that during reinforcement training of the DS with the US, the US accepts the system response in natural language. In contrast, for those USs with 'N' marked in the 'Natural Lang. Interaction' column, the system acts are directly fed to the US so that the US does not need a natural language understanding module. For such as case, the US is also said to work at the semantic level.

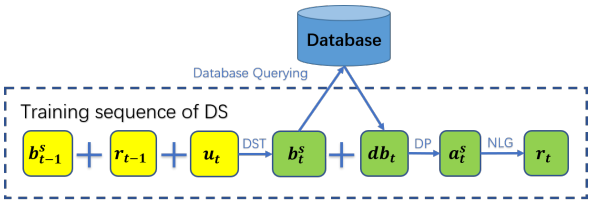
following.

US Architecture A variety of user simulators have been studied, either rule-based or data-driven. A classic rule-based US is the agenda-based user simulator (ABUS) (Schatzmann et al., 2007). Different data-driven US models are proposed with different architectures and characteristics. Asri et al. (2016) develops a LSTM-based seq2seq US on the single-domain DSTC2 dataset and generates semantic-level user acts. Gür et al. (2018) proposes a GRU-based hierarchical seq2seq framework for US (HUS) and further introduces a latent variable to control the diversity of dialogue (VHUS). NUS (Kreyssig et al., 2018) extracts feature vectors related to current goal states and feeds to a LSTM seq2seq model to output natural languages. Shi et al. (2019) make extensive comparisons for six user simulators, based on two user policy modules (seq2seq or agenda based) and three NLG modules (template, retrieval or seq2seq). TUS in (Lin et al., 2021) designs domain-independent features and implements the user policy as multi-class classification so that TUS could be easily adapted to new domains. Some studies aim to jointly optimize DS and US. The USs used in these studies are mostly based on LSTM seq2seq architectures (Liu and Lane, 2017; Papangelis et al., 2019; Tseng et al., 2021), or simply as multi-class classification for action selection with feed-forward networks (Takanobu et al., 2020).

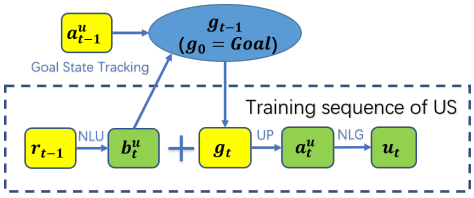
Goal State Tracking in US ABUS is classic in goal state tracking, where the pending user acts are tracked in a stack-like structure, called agenda. ABUS is rule-based, generating user acts by pushing and popping hand-crafted rules from agenda. The goal state tracking process is overlooked in some later studies of data-driven USs (Asri et al., 2016; Gür et al., 2018; Papangelis et al., 2019), where the US is always conditioned on the whole initial user goal at each turn. Some data-driven USs explicitly track goal states but employ binary vectors (Kreyssig et al., 2018; Lin et al., 2021; Tseng et al., 2021). The US in (Takanobu et al., 2020) represents goal states by tokens, which is flexible, but the US only interacts with the DS at the semantic level (not end-to-end trainable).

3 Preliminaries

Notations According to the information flow in a task-oriented dialog between a US and a DS as illustrated in Figure 1, we let g_t denote the user goal state, a_t^u the user act, u_t the user utterance, b_t^s the system belief state, db_t the database result, a_t^s the system act, and r_t the system response, respectively, at turn $t = 1, \cdots, T$, for a dialog of T turns. Moreover, in this paper we are interested in building end-to-end trainable US that can interact with the DS in natural languages. Thus, we introduce a NLU module in the US, which takes the system response r_t as input and infer system intent. The NLU result is denoted by b_t^u, or loosely speaking, referred to as the user belief state. Notably, the US belief state b_t^u denotes the US's understanding only about the previous system response, and accordingly is labeled as a_{t-1}^s in training. b_t^u is not of accumulated nature, since the US uses the goal state g_t to summarize the dialog history encountered by the US[3].



(a) Architecture of Dialog System (DS)



(b) Architecture of User Simulator (US)

Figure 2: The generative architecture of dialog system and user simulator in our experiments.Yellow boxes represent the conditioning input of the model during generation, and green boxes the targeting output.

GPT-2-based Generative Architecture In this work, all variables defined in the last paragraph for the US and DS are converted to token sequences, like in DAMD (Zhang et al., 2020). So pretrained language models (LMs) such as GPT-2 can be fine-tuned to build end-to-end trainable DS and US, as will be introduced later. To be clear, GPT-2 (Radford et al., 2019) in this paper refers to the particular class of causal LMs, which computes conditional probabilities for next-token generation

[3]In contrast, the system belief state b_t^s summarizes the dialog history encountered by the DS. This subtle difference makes sense, since the roles of the DS and US are different.

via self-attention based Transformer neural network (Vaswani et al., 2017). Given a particular form of conditional model, $p(output|input)$, where $input$ and $output$ are token sequences, the GPT-2 model can be finetuned over training samples $(input, output)$ (often referred to as training sequences (Hosseini-Asl et al., 2020)), and after finetuning, the model can be used for generation, i.e., generating $output$ after receiving $input$.

Generative Dialog System The main task for a dialog system (DS) is, for each dialog turn t, to generate (or say, predict)[4] b_t^s, a_t^s and r_t, given u_t and dialog history $u_1, r_1, \cdots, u_{t-1}, r_{t-1}$. A recent progress in building DS is that all variables are represented by token sequences, and the workflow of a dialog system (DST, DP and NLG) is unified into a single sequence generation problem, which can be accomplished by a causal LM such as GPT-2 (Hosseini-Asl et al., 2020; Liu et al., 2022). In this paper, we employ the Markov generative architecture (MGA) for DS, which is introduced in Liu et al. (2022) and shows efficiency advantages in memory, computation and learning over non-Markov DS models like SimpleTOD (Hosseini-Asl et al., 2020). Specifically, for DS to predict b_t^s, a_t^s and r_t at each turn t, we use only the belief state b_{t-1} and response r_{t-1} from previous turn along with current user utterance u_t, as shown in Figure 2(a). The DS can thus be trained via finetuning GPT-2 by maximizing the following conditional likelihood over labeled training sequences for supervised learning (SL):

$$\begin{aligned}\mathcal{J}_{\text{DS-SL}} &= \log p_\theta(b_t^s, a_t^s, r_t | b_{t-1}^s, r_{t-1}, u_t) \\ &= \sum_{i=1}^{|b_t^s \oplus a_t^s \oplus r_t|} \log p_\theta(c_i | b_{t-1}^s, r_{t-1}, u_t, c_{<i})\end{aligned} \tag{1}$$

where $\oplus$ denotes the concatenation of sequences, $|b_t^s \oplus a_t^s \oplus r_t|$ denotes the length in tokens, and c_i denotes the i-th token. The DS parameters are actually a set of GPT-2 parameters, collectively denoted by θ.

4 Method: Generative User Simulator

An end-to-end trainable US needs several modules - natural language understanding (NLU) of system responses, goal state tracking (GST), user policy (UP), and natural language generation (NLG). Inspired by the recent approach of finetuning PLMs such as GPT-2 to build end-to-end trainable generative DSs, we propose an end-to-end trainable generative user simulator (GUS), which generally refer to the approach of recasting all the modules in the US (NLU, UP, and NLG) as conditional generation of tokens based on finetuning PLMs such as GPT-2. In the following, we first introduce the GUS model including goal state tracking and GPT-2 based architecture. Then, we describe how GUS is trained and used for reinforcement training of the DS.

4.1 GUS Model

Goal State Definition Crucially, the interaction between the user and the system is motivated by the user goal, which is composed of user constraints and requests such as booking a cheap hotel. The goal state, in this paper, is defined as the uncompleted part of the user goal at each turn. Similar to Kreyssig et al. (2018), we accumulate the annotated user acts backwards turn by turn to obtain the goal state annotation at each turn. The accumulation process is illustrated in Appendix A.1.The goal state obtained at the first turn corresponds to the initial user goal for the whole dialog session.

Goal State Tracking Given the goal state annotations at each turn, the US can be trained via teacher-forcing to mimic the user behaviors. When the US is applied to interact with the DS for evaluation or for reinforcement training of the DS, the US needs to track the completion process of the goal to update the goal state turn by turn, which we call goal state tracking. There are three types of user intents in the goal state g_t - *inform, book and request*. The slots and values for the first two types of intents in g_t are denoted by $g_t^{constraint}$ and those of the *request* intent as $g_t^{request}$. The update rule of g_t at turn t is designed to be as follows:

$$\begin{aligned} g_t^{constraint} &= g_{t-1}^{constraint} \ominus a_{t-1}^{u,inform} \\ g_t^{request} &= g_{t-1}^{request} \ominus b_t^{u,inform} \end{aligned} \tag{2}$$

where $a_{t-1}^{u,inform}$, $b_t^{u,inform}$ are the informable slots and values in user act a_{t-1}^u and user belief state b_t^u respectively and $\ominus$ denotes removing the corresponding slots and values. Moreover, the slot values in the initial user goal may be changed during the interaction (i.e., goal change). When the DS expresses *no-offer* intent, which means no entities in the database satisfy the constraints of the

[4]Note that database result db_t is deterministically obtained by querying database using the predicted b_t^s. We omit db_t in the discussion for simplicity.

goal, we randomly select one slot in the *no-offer* intent and replace its value with another value in the ontology.

GPT-2-based Architecture The main task for a US is, conditional on the user goal, to iteratively understand the system response, track goal state, decide user act, and generate user utterance. In this work, we find that the recent approach of finetuning GPT-2 for conditional generation can be similarly applied to build US. Specifically, we employ Markov generative architecture (Liu et al., 2022). The US is designed to firstly infer the system intent, i.e., user belief state b_t^u of turn t from the previous system response r_{t-1}, which could be modeled as $p_\phi(b_t^u|r_{t-1})$. After obtaining b_t^u, the goal state will be updated according to the rule in Eq. (2). Then, the US will generate user act and user utterance sequentially conditioned on the previous system response, user belief state, and the updated goal state. The resulting US is called GUS and could be modeled as $p_\phi(a_t^u, u_t|r_{t-1}, b_t^u, g_t)$. The GUS parameters are actually another set of GPT-2 parameters, collectively denoted by ϕ.

4.2 GUS Training

The GUS model can thus be trained via finetuning GPT-2 by maximizing the following conditional likelihood over labeled training sequences for supervised learning (SL):

$$\begin{aligned}\mathcal{J}_{\text{US-SL}} &= \log p_\phi(b_t^u|r_{t-1}) \\ &+ \log p_\phi(a_t^u, u_t|r_{t-1}, b_t^u, g_t)\end{aligned} \tag{3}$$

Note that during supervised learning, the user belief state b_t^u is labeled by directly copying the system act a_{t-1}^s from the previous turn.

4.3 Reinforcement Optimization of DS through Interaction with US

RL Setup The DS and US described above will first be trained using supervised learning with the objectives in Eq. (1) and Eq. (3) respectively. After supervised learning, we can perform RL optimization on the DS through interactions with the US. The DS agent view the US as the environment and use its conditional model $p_\theta(b_t^s, a_t^s, r_t|b_{t-1}^s, r_{t-1}, u_t)$ as its policy. Here the policy of the DS involves generating not only system act a_t^s, but also belief state b_t^s and system response r_t. This is different from some previous studies of learning reinforced DS, e.g., (Liu and Lane, 2017; Papangelis et al., 2019; Tseng et al., 2021), which only use RL to optimize the selection of system acts (but all use traditional LSTM seq2seq architectures). However, thanks to the representation power of GPT-2, recursively predict (or say, decide about) b_t^s, a_t^s and r_t in one policy yields the best performance in our experiment. In Section 7.3, we compare different schemes for policy definition for the DS agent with more discussions.

RL Optimization We apply the policy gradient method (Sutton et al., 2000) to optimize the DS for RL. We first let the two agents interact with each other based on the user goals from the goal generator provided by ConvLab-2 (Zhu et al., 2020). Then we calculate the reward R_t for each turn, as detailed below. The return $U_{i,t}$ for the action of turn t at the i-th step is $\gamma^{|A_t|-i}R_t$, where γ is the discounting factor and $|A_t|$ is the policy sequence length of turn t. We update the DS with the following policy gradients:

$$\nabla_\theta \mathcal{J}_{\text{DS-RL}} = \sum_{i=1}^{|b_t^s \oplus a_t^s \oplus r_t|} U_{i,t} \nabla_\theta \log p_\theta(c_i) \tag{4}$$

where $p_\theta(c_i)$ denotes $p_\theta(c_i|b_{t-1}^s, r_{t-1}, u_t, c_{<i})$.

Reward Settings A number of different settings for reward have been studied, as described in the following. The three settings are separately tested, and the experimental results are given in Section 7.2.

1) Success. If a dialog is successful, we set the reward of each turn to 1, otherwise it is set to be 0;

2) A turn-level synthetic reward similar to Tseng et al. (2021); Takanobu et al. (2020), which consists of requesting reward (+0.1 for each), repeating punishment (-0.5 for each) and task completion reward (the proportion of tasks completed) of the DS;

3) A Sigmoid synthetic reward obtained by mapping the synthetic reward to [0,1] interval using the Sigmoid function. This setting is designed to exclude the influence of the value range of reward because the value range is different between the Success reward and the synthetic reward.

5 Experiments

5.1 Dataset

Experiments are conducted on MultiWOZ2.1 (Eric et al., 2020), which is an English multi-domain task-oriented dialog dataset of human-human conversations. It contains 10.4k dialogs, collected in a Wizard-of-Oz setup over seven domains. The

dataset contains the annotations of system belief state, system act, and user act for every turn.

5.2 Evaluation Metrics

Evaluating the quality of a US is not trivial. The performance of the reinforced DS trained with a specific US gives an *indirect* assessment of the quality of the US. Considering that a main purpose of developing USs is to help train RL based DSs, this indirect assessment makes sense and is widely employed (Kreyssig et al., 2018; Shi et al., 2019; Lin et al., 2021). We conduct both automatic evaluation and human evaluation of the DSs trained with different USs. Additionally, we also ask human graders to *directly* assess the performance of different USs, by reading and scoring the generated utterances from the USs.

Automatic Evaluation It could be interaction-based or corpus-based. For both manners, we can calculate *Inform* and *Success* for measuring the performance of the DSs. *Inform Rate* measures how often the entities provided by the system are correct. *Success Rate* refers to how often the system is able to answer all the requested attributes by user. *BLEU Score* is used to measure the fluency of the generated system responses when conducting corpus-based evaluation.

Human Evaluation We conduct human evaluation, where human graders are recruited to assess the quality of dialogs generated by the US and the DS trained with it. Similar to Su et al. (2021), for each dialog, the grader will score the conversation on a 3-point scale (0, 1, or 2)[5] by the following 3 metrics for the DS and 2 metrics for the US:

- Success. This metric measures if the DS successfully completes the user goal by interacting with the US;
- DS Coherency (DS-coh). This metric measures whether the system's response is logically coherent with the dialogue context;
- DS Fluency (DS-Flu). This metric measures the fluency of the system's response.
- US Coherency (US-Coh). This metric measures whether the simulator's utterance is logically coherent with the dialogue context;
- US Fluency (US-Flu). This metric measures the fluency of the simulator's utterance.

[5]Three scales (0, 1 and 2) denote three degrees - not at all, partially and completely, respectively.

5.3 Baseline

The DS model is as described in Section 3. We compare GUS with the classic rule-based simulator ABUS (Schatzmann et al., 2007). We use the simulator in the ConvLab-2 (Zhu et al., 2020) toolkit, which provides an instantiation of ABUS on MultiWOZ (Budzianowski et al., 2018), including BERT-based NLU and template-based NLG. The ABUS in ConvLab-2 has a goal generator module, which we use for driving the interaction between the DSs and the proposed GUS. Remarkably, the TUS paper (Lin et al., 2021) has revealed the shortcoming of VHUS (Gür et al., 2018), which performs much worse than ABUS. Also, it is concluded that TUS has a comparable performance to the rule-based ABUS in cross-model evaluation. Thus, in this paper, we mainly compare GUS with ABUS, which is a very strong baseline.

6 Main Results

6.1 Cross-Model Evaluation

Cross-model evaluation is a type of automatic evaluation (Schatztnann et al., 2005) to compare different USs. The main idea is that if the DS trained with a specific US performs well on all USs (not just on the one that the DS was trained with), it indicates the specific US with which the DS was trained is of good quality (realistic), and thus the DS is likely to perform better on real users.

Specifically, we first train a DS and a US separately on training data based on the supervised learning objectives described in Eq. (1) and Eq. (3). The resulting models are referred to as DS-SL and GUS respectively. Then we further optimize DS-SL by policy gradient in Eq. (4) on interaction with either ABUS or GUS, and obtain DS-ABUS and DS-GUS respectively. For either of ABUS and GUS, RL trainings (all starting from DS-SL) are independently taken for three times with different random seeds. Each specific DS model is then tested on both ABUS and GUS. We use the same 1000 randomly generated goals for each test. Further implementation details can be found in Appendix A.2. Table 2 shows the cross-model evaluation results[6].

It can be seen from Table 2 that the DS trained with GUS (DS-GUS) performs well on both ABUS

[6]Similar tables to Table 2 have been used in previous work such as NUS (Kreyssig et al., 2018) and TUS (Lin et al., 2021). A common practice of reading such tables is row-by-row comparison. This is exactly what the cross-model evaluation means.

DS \ US	ABUS		GUS	
	Inform	Success	Inform	Success
DS-SL	0.864	0.791	0.781	0.736
DS-ABUS$_{best}$	0.885	0.816	0.783	0.741
DS-GUS$_{best}$	0.881	0.810	0.864	0.808
DS-ABUS$_{avg}$	0.889	0.793	0.793	0.735
DS-GUS$_{avg}$	0.872	0.801	0.859	0.802

Table 2: Cross-model evaluation results. The subscripts $best$ and avg denote the best and the average from 3 independent RL experiments with different random seeds.

DS	Inform	Success	BLEU	Combined
AuGPT (Kulhánek et al., 2021)	76.6	60.5	16.8	85.4
SOLOIST (Li et al., 2020)	82.3	72.4	13.6	90.9
UBAR (Yang et al., 2021)	83.4	70.3	17.6	94.4
DS-SL	84.10	72.10	19.24	97.34
DS-ABUS$_{best}$	84.20	71.00	19.44	97.04
DS-ABUS$_{avg}$	85.37	69.70	19.10	96.64
DS-GUS$_{best}$	85.70	74.60	19.80	99.95
DS-GUS$_{avg}$	85.17	73.33	19.83	99.01

Table 3: Corpus-based evaluation. Above the dashed line are GPT-2-based results from the official website of MultiWOZ. Below are the results from DS-SL and the DSs trained with ABUS and GUS respectively.

and GUS, while the DS trained with ABUS (DS-ABUS) only performs well on ABUS and achieves much lower Inform and Success when tested with GUS. This indicates the superiority of GUS over ABUS, being more helpful in training reinforced DSs that perform well on both USs. Moreover, DS-GUS also outperforms the supervised baseline (DS-SL) on both USs. This shows the practical benefit brought by training DSs via RL on interaction with the proposed GUS. Such comparison of RL and SL is overlooked in some prior work, as reviewed in Table 1.

6.2 Corpus-based Evaluation

Corpus-based evaluation has become a widely-used type of automatic evaluation to compare different end-to-end DSs. In the context of studying USs, it is relevant to conduct corpus-based evaluation for the following two aspects. First, running testing of the DS trained with a specific US over a fixed testing set of dialogs could be an indirect assessment of the quality of the US. Second, it is possible for the trained DS via RL to achieve high task success and yet not generate human language (Zhao et al., 2019), particularly when the reward is mainly defined to encourage task success. With the fixed testing set, we could calculate BLEU which measures the NLG performance of the trained DS.

We use the standard evaluation scripts from Nekvinda and Dušek (2021) for corpus-based evaluation. The results are shown in Table 3 with some interesting findings. First, the DS trained with GUS (DS-GUS) achieves higher combined score than the DS trained with ABUS (DS-ABUS). This is consistent with the results in Table 2 and again demonstrate the advantage of GUS over ABUS. Second, note that DS-GUS is initialized from DS-SL and further trained via RL on interaction with GUS, and Table 2 shows that DS-GUS improves over DS-SL not only in Inform and Success but also in BLEU. This result indicates that RL training of the DS with GUS does not suffer from the tradeoff problem between policy learning and NLG in offline RL (Zhao et al., 2019)[7], achieving higher success and being faithful to human language. See more discussions in Section 7.3.

6.3 Human Evaluation

We further perform human evaluation of the performances of USs and DSs. For each pair of US and DS, 100 dialogs were gathered, which were scored by 5 human graders. The details of evaluation metrics have been described in Sec. 5.2 and the results are shown in Table 4. For convenience, we refer to the results of each row by the name of the DS in the table. It can seen that the overall performance of DS-GUS is superior over both DS-ABUS and DS-SL. Further, we conduct significance tests by comparing either DS-ABUS or DS-SL with DS-GUS respectively, using the matched-pairs method (Gillick and Cox, 1989) and add a superscript * to the score in the first two rows in Table 4 if the p-value is less than 0.05. All the specific p-values can be seen in Appendix A.4. The results show that DS-GUS significantly improves over DS-SL for Success and US-Coh, while the differences in terms of DS-Coh, DS-Flu and US-Flu are not significant. Moreover, all the human evaluation metrics by DS-GUS are stronger than or equal to those by DS-ABUS. Particularly, DS-GUS significantly outperforms DS-ABUS for DS-Flu, US-Coh and US-Flu. This indicates that GUS is able to generate more coherent and fluent utterances than ABUS. To illustrate this point, we provide some generated dialogues in Appendix A.3.

[7]This problem for offline RL is further studied and alleviated in Lubis et al. (2020).

DS	US	Success	DS-Coh	DS-Flu	US-Coh	US-Flu
DS-ABUS	ABUS	1.71	1.51	1.65*	1.27*	1.30*
DS-SL	GUS	1.73*	1.60	1.85	1.61*	1.88
DS-GUS	GUS	1.84	1.52	1.79	1.75	1.90

Table 4: Human evaluation of the dialogs generated by different DSs and USs. The score with * in the first two rows denotes the difference between this score and the score in the last row (DS-GUS with GUS) is significant (p-value<0.05); otherwise, the difference is not significant (p-value>=0.05).

US	Inform	Success
ABUS	0.863	0.790
GUS	0.825	0.777
GUS w/o GST	0.743	0.502

Table 5: The ablation results about goal state tracking (GST). The DS trained with GUS w/o GST is tested on ABUS, GUS and GUS w/o GST respectively.

Reward	Inform	Success
None	0.781	0.736
Success	0.842	0.787
Synthetic	0.864	0.808
Sigmoid synthetic	0.850	0.780

Table 6: Interaction-based results of testing DS-GUS on GUS under different reward settings, as introduced in Section 7.2. "None" denotes the testing results of DS-SL with GUS, as also reported in the first row in Table 2.

Policy	Inform	Success
$b_t^s \oplus a_t^s \oplus r_t$	0.864	0.808
$a_t^s \oplus r_t$	0.845	0.770
a_t^s	0.848	0.796

Table 7: The ablation experiments of using different policy schemes.

7 Ablation Study

7.1 The Importance of Goal State Tracking

In our GUS model, we use Eq. (2) to update the goal state at every turn. In the section, we consider a variant of GUS, which sets the goal state at all turns to be the initial goal, that is, $g_t = g_0, t = 1, ..., T$, like in Asri et al. (2016); Gür et al. (2018); Papangelis et al. (2019). Such model is referred to as GUS w/o GST, and could be similarly trained according to Eq. (3). Then we train a DS with this US (called "DS-GUS w/o GST") and test it with ABUS, GUS and GUS w/o GST respectively. The results are shown in Table 5. We can see that the Inform and Success rates obtained by "DS-GUS w/o GST" are lower than those by DS-GUS as shown in Table 2, when testing on ABUS and GUS. This indicates the importance of using GST in GUS. Besides, we can see that the results are pretty low when testing on GUS w/o GST. Presumably, this is because GUS w/o GST cannot accurately distinguish the uncompleted part in the complex goal, which will easily cause omission and repetition when generating user acts.

7.2 Different Reward Settings

The results of optimizing DS on GUS using different reward settings are reported in Table 6. It is found that all reward settings achieve better results than supervised baseline (Reward=None) and the synthetic reward setting achieves the best result, which is reasonable since the fine-grained rewards reflect more than simple success rate in terms of the nature of the tasks (Tseng et al., 2021). All RL results in this paper are based on this setting of reward, unless here for ablation study.

7.3 Different Policy Schemes for DS

The policy in RL refers to the probabilistic mapping from states to actions. Previous studies of learning reinforced DS, e.g., (Liu and Lane, 2017; Papangelis et al., 2019; Tseng et al., 2021), mainly employ RL to optimize the policy module, i.e., use system acts for actions. In contrast, the policy of DS-GUS and DS-ABUS involves generating not only system act a_t^s, but also belief state b_t^s and system response r_t, which can be represented as $b_t^s \oplus a_t^s \oplus r_t$, as illustrated in Eq. (4). To compare policy schemes for reinforced DS, we try two other policy schemes when optimizing DS-GUS. The first policy scheme only involves the generation of system act a_t^s and the second one involves the generation of both system act a_t^s and system response r_t. We denote the two policy schemes as a_t^s and $a_t^s \oplus r_t$ respectively. Table 7 shows the interaction results when the DS-GUS trained under different policy schemes is tested with GUS.

It can be seen from Table 7 that using $b_t^s \oplus a_t^s \oplus r_t$ for policy achieves the highest Inform and Success rate. We provide two points, which may explain the advantage of our model in using $b_t^s \oplus a_t^s \oplus r_t$ for RL. First, since the DST, DP and NLG modules in GPT-2 based DS share the model parameters, parameter adjust in one module will affect other modules. Only optimizing DP during RL without considering other modules may mislead other modules. Using

$b_t^s \oplus a_t^s \oplus r_t$ leads to better overall optimization and decision-making. Second, the balance between policy learning and NLG, which was a concern in previous studies when using modular or small-capacity architectures (Zhao et al., 2019), could be relieved, thanks to the high-capacity of GPT-2.

8 Conclusion

In this paper, towards developing an end-to-end trainable US for multi-domains, a generative user simulator (GUS) with GPT-2 based architecture and goal state tracking is proposed and systematically evaluated. We train GPT-2 based DSs and USs and conduct cross-model evaluation, corpus-based evaluation and human evaluation. The results show that the DS trained with GUS outperforms both the supervised trained DS and the DS trained with ABUS. The human evaluation further confirms the superiority of GUS and shows that GUS can generate much more coherent and fluent utterances than ABUS. Moreover, GUS is simple and easy to use, in addition to its strong performance. Hope this work will stimulate further work on developing and using user simulators in the study of building dialog systems.

9 Limitations

There are some limitations of this work. First, due to computational constraints, both the DSs and the USs are experimented based on a distilled version of GPT-2. Studies using larger GPT-2 and other classes of larger PLMs such as T5 (Raffel et al., 2020) would enhance our results in this paper. Second, we only utilize the policy gradient method for RL in this paper. Other advanced RL methods such as proximal policy optimization (PPO) and actor-critic are also worth trying in future work. Those being said, while we agree that experimenting with larger PLMs and more complex RL methods are meaningful, we believe the extensive experiments presented in this paper (cross-model evaluation, corpus-based evaluation, human evaluation, and ablation studies) can well support the evaluations of GUS and should not affect the main finding and contribution of this paper.

References

Layla El Asri, Jing He, and Kaheer Suleman. 2016. A sequence-to-sequence model for user simulation in spoken dialogue systems. In *INTERSPEECH*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 *LREC*.

Laurence Gillick and Stephen J Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 532–535. IEEE.

Izzeddin Gür, Dilek Hakkani-Tür, Gokhan Tür, and Pararth Shah. 2018. User modeling for task oriented dialogues. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 900–906.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Florian Kreyssig, Iñigo Casanueva, Paweł Budzianowski, and Milica Gašić. 2018. Neural user simulation for corpus-based policy optimisation of spoken dialogue systems. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 60–69, Melbourne, Australia. Association for Computational Linguistics.

Jonáš Kulhánek, Vojtěch Hudeček, Tomáš Nekvinda, and Ondřej Dušek. 2021. Augpt: Dialogue with pre-trained language models and data augmentation. *arXiv preprint arXiv:2102.05126*.

Yohan Lee. 2021. Improving end-to-end task-oriented dialog system with a simple auxiliary task. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1296–1303, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Baolin Peng Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics (TACL), 2021*.

Hsien-chin Lin, Nurul Lubis, Songbo Hu, Carel van Niekerk, Christian Geishauser, Michael Heck, Shutong Feng, and Milica Gasic. 2021. Domain-independent user simulation with transformers for task-oriented dialogue systems. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 445–456, Singapore and Online. Association for Computational Linguistics.

Bing Liu and Ian R. Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*, pages 482–489. IEEE.

Hong Liu, Yucheng Cai, Zhijian Ou, Yi Huang, and Junlan Feng. 2022. Building Markovian generative architectures over pretrained LM backbones for efficient task-oriented dialog systems. *ArXiv preprint arXiv:2204.06452*.

Nurul Lubis, Christian Geishauser, Michael Heck, Hsien-Chin Lin, Marco Moresi, Carel van Niekerk, and Milica Gasic. 2020. LAVA: Latent action spaces via variational auto-encoding for dialogue policy optimization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 465–479.

Tomáš Nekvinda and Ondřej Dušek. 2021. Shades of BLEU, flavours of success: The case of MultiWOZ. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 34–46, Online. Association for Computational Linguistics.

Alexandros Papangelis, Yi-Chia Wang, Piero Molino, and Gokhan Tur. 2019. Collaborative multi-agent dialogue model training via reinforcement learning. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 92–102, Stockholm, Sweden. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152, Rochester, New York. Association for Computational Linguistics.

Jost Schatztnann, Matthew N Stuttle, Karl Weilhammer, and Steve Young. 2005. Effects of the user model on simulation-based learning of dialogue strategies. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2005.*, pages 220–225. IEEE.

Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. 2019. How to build user simulators to train RL-based dialog systems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1990–2000, Hong Kong, China. Association for Computational Linguistics.

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. *CoRR*, abs/2109.14739.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Ryuichi Takanobu, Runze Liang, and Minlie Huang. 2020. Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 625–638, Online. Association for Computational Linguistics.

Bo-Hsiang Tseng, Yinpei Dai, Florian Kreyssig, and Bill Byrne. 2021. Transferable dialogue systems and user simulators. 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 152–166, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Yichi Zhang, Zhijian Ou, Min Hu, and Junlan Feng. 2020. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (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. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1208–1218, Minneapolis, Minnesota. Association for Computational Linguistics.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

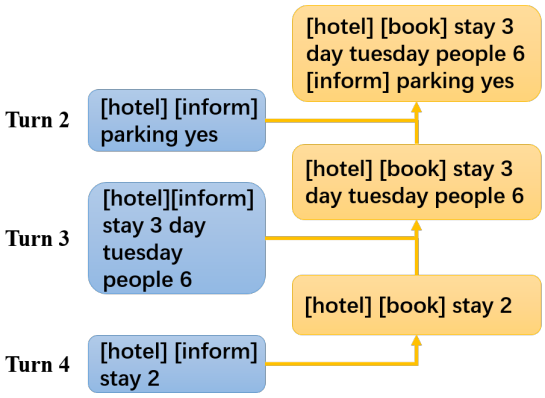


Figure 3: An example of how turn-level goal state annotations are obtained. The blue boxes are user acts and the yellow ones are goal states.

SNG0616	
User	Sorry, actually I need an expensive restaurant in the north. The first on your list would be great.
Bspan	[restaurant] pricerange expensive area north
Act	[restaurant] [inform] name
Resp	Sure how about [value_name]?
Bspan_{SL}	[restaurant] pricerange expensive area north food north
Act_{SL}	[restaurant] [nooffer] food area [request] food
Resp_{SL}	I am sorry, there are no [value_food] restaurants in the [value_area] . Would you like to try a different type of cuisine?
Bspan_{RL}	[restaurant] pricerange expensive area north
Act_{RL}	[restaurant] [inform] choice price area [request] food
Resp_{RL}	There are [value_choice] [value_price] restaurants in the [value_area] . What type of food would you like?

Table 8: One dialog turn in the test set. Bspan and Act denote the sequence forms of belief state and system act. The subscript SL and RL represent the supervised trained model DS-SL and the RL model DS-GUS respectively.

A Appendices

A.1 Data Processing

We delexicalize system responses following Zhang et al. (2020) to reduce surface language variability. Specifically, we replace values in the ontology with specific placeholders such as $[value\_name]$ and $[value\_price]$. The proposed DS and US are both trained on the delexicalized dataset. During human evaluation or interaction with ABUS, the system responses need to be lexicalized. We then replace those placeholders with corresponding values in the predicted entities by querying the given database with the predicted belief states.

For building US, we need to accumulate the annotated user acts backwards turn by turn to obtain the goal state annotation at each turn as we described in Sec 4. The accumulation process is depicted in Figure 3.

A.2 Implementation Details

We use Huggingface Transformers repository. GPT-2 based DSs and USs are initialized with DistilGPT-2 (Sanh et al., 2019), a distilled version of GPT-2, with 6 transformer decoder layer. During supervised learning, we use the AdamW optimizer and a linear scheduler with 20% warm up steps and maximum learning rate 10^{-4}. The minibatch base size is set to be 8 with gradient accumulation steps of 4. During RL, we no longer use scheduler and fix the learning rate to 2×10^{-5}. The minibatch base size is set to be 16 with gradient accumulation steps of 12. For each interaction, the dialog will end in the following three cases: 1) both the DS and US generate `bye` intent; 2) the goal state of the US is empty; 3) the content of the current turn is exactly the same as that of the previous turn. Besides, to increase the diversity of dialogues, beam search decoding is applied during generating user acts and system acts. The beam size is set to be 10 and the final act will be sampled by probability from the 10 candidates. All the SL and RL experiments are conducted on a single 16GB Tesla-P100 GPU.

A.3 Case Study

Interaction Case To illustrate the advantage of GUS over ABUS, we let DS-ABUS and DS-GUS interact with their corresponding USs under the same user goal. The generated dialogs are shown in Table 9. It can be seen that the user utterances generated by ABUS are unlike natural languages and sometimes logically coherent. For instance, ABUS says "I want a train" when it has said "please book the train for me" before in the first turn. On the contrary, GUS can generates very fluent and logically coherent utterances, and is able to understand the system response and provide necessary information to complete the dialogue.

Corpus-based Evaluation Case We can see from Table 3 that DS-GUS improves the Success rate over DS-SL on test set. We show an dialog example in Table 8 to explain how the RL improves the DS performance. In this turn, DS-SL predicts a wrong Bspan with a redundant slot "food" and incorrect value "north". As a result, no entity can be found when querying database and DS-SL generates "nooffer" intent, which finally leads to an unsuccess dialog. However, DS-GUS can predict a correct Bspan and generate an appropriate system act in this turn. This indicates that RL can improve the ability of dialog state tracking of DS, thus improving the success rate. In fact, this advantage

User Goal
Train: Inform (arrive=12:45; day=Thursday; departure=Birmingham new street; destination: Cambridge), Book (people=8)
Attraction: Inform (area=centre), Request (postcode, phone)
Dialogue generated through the interaction between DS-ABUS and ABUS
$user_1$: Please book the train for me. I am departing from Birmingham new street. I want a train leaving on Thursday. I would like to go to Cambridge please.
$resp_1$: There are 19 trains that meet your criteria. What time would you like to leave?
$user_2$: Please pick one for me. I don't care about the leave of the train.
$resp_2$: Tr4235 leaves at 05:40. Would you like me to book it for you?
$user_3$: I want to get there by 12:45.
$resp_3$: Tr4235 arrives at 08:23. Would you like me to book it for you?
$user_4$: I would like 8 tickets please.
$resp_4$: Booking was successful, the total fee is 600.8 pounds payable at the station. Reference number is: 0000002206. Is there anything else I can help you with?
$user_5$: Can you help me plan a trip to see a particular attraction? I'd like something in the centre.
$resp_5$: There are 44 theatre and architecture in the centre of town. What type of attraction are you interested in?
$user_6$: Just select one for me. I don't care about the type of the attraction.
$resp_6$: There are 44 museums in the centre of town. May I suggest Adc theatre ?
$user_7$: Sounds good. Could I get the phone number of the attraction? Can you give me the postcode of the attraction?
$resp_7$: The phone number is 01223300085 and the postcode is CB58AS. Is there anything else I can help you with?
$user_7$: That's all the help I need. Take care. Bye.
$resp_7$: Thank you for contacting us and have a nice day.
Dialogue generated through the interaction between DS-GUS and GUS
$user_1$: I am looking for places to go in the centre of town.
$resp_1$: There are 44 attractions in the centre. Do you have a specific type in mind?
$user_2$: No, I don't have a preference.
$resp_2$: I would recommend Adc Theatre on Park Street. It is free to get in.
$user_3$: That sounds great. Can I get the phone number and postcode?
$resp_3$: The phone number is 01223300085 and the postcode is CB58AS. Is there anything else i can help you with?
$user_4$: I am also looking for a train going to Cambridge.
$resp_4$: Where will you be departing from and what day would you like to travel?
$user_5$: I will be departing from Birmingham new street on Thursday.
$resp_5$: There are 19 trains that meet your criteria. Do you have a preferred departure or arrival time?
$user_6$: I would like to arrive by 12:45. I would like to book for 8 people.
$resp_6$: Booking was successful, the total fee is 600.8 pounds payable at the station. Reference number is: 0000002206.
$user_7$: Thank you so much! that is all I need.
$resp_7$: Thank you for contacting us and have a nice day.

Table 9: Dialogs generated through the interaction between DSs and USs.

comes from the special policy scheme employed in this paper, as discussed in Sec 7.3.

A.4 Significance Test

In Sec. 6.3, we conduct significance tests to show whether the differences between the first two groups and the last group in Table 4 are significant. The p-values are listed in Table 10.

DS	Success	DS-Coh	DS-Flu	US-Coh	US-Flu
DS-ABUS vs DS-GUS	0.065	0.535	0.036	0.000	0.000
DS-SL vs DS-GUS	0.045	0.220	0.273	0.020	0.639

Table 10: Significance tests for human evaluation. We refer to the results of each row in Table 4 by the name of the DS.