Jointly Reinforced User Simulator and Task-oriented Dialog System with Simplified Generative Architecture

Anonymous ACL submission

Abstract

Recently, there has been progress in supervised finetuning pretrained GPT-2 to build end-to-end task-oriented dialog (TOD) systems. However, online reinforcement learning of a GPT-2 based dialog system (DS), together with an end-to-end user simulator (US), has not ever been explored. Moreover, a drawback with existing GPT-2 based TOD systems is that they mostly employ the whole dialog history as input, which brings inefficiencies in memory and compute. In this paper, we first propose Simplified Generative Architectures (SGA) for DS and US respectively, both based on GPT-2 but using shortened history. Then, we successfully develop Jointly Reinforced US and DS, called SGA-JRUD. Our DS with the proposed SGA, when only supervised trained, achieves state-of-the-art performance on MultiWOZ2.1 and is more compute-efficient in both training and generation. Extensive experiments on MultiWOZ2.1 further show the superiority of SGA-JRUD in both offline and online evaluations.

1 Introduction

Task-oriented dialog (TOD) systems, which are mainly designed to assist users to accomplish their goals, often consist of several modules including dialog state tracking (DST), database querying (DB), dialog policy (DP) and natural language generation (NLG). The information flow in a task-oriented dialog is illustrated in Figure 1. Recent studies recast these modules all as conditional generation of tokens and integrate them into a single language model (LM), which usually uses some pretrained language model (LM) such as GPT-2 (Radford et al., 2019) as the backbone. Fine-tuning GPT-2 over annotated dialog datasets such as MultiWOZ (Budzianowski et al., 2018) via supervised learning (SL) has shown state-of-the-art results (Hosseini-Asl et al., 2020; Li et al., 2020; Kuhánek et al., 2021; Yang et al., 2021), thanks to the powerful generation ability of GPT-2.

However, it has long been recognized that supervised learning over annotated dialog datasets alone may not be sufficient to learn a task-oriented dialog agent (Young et al., 2013). Conversations often do not have only a single correct response, multiple responses can be appropriate for the same dialog context (Zhang et al., 2020). Supervised trained agents can become biased by the annotations. Reinforcement learning (RL) for an agent aims to goal-directed learning from interaction between the decision-making agent and its environment (Sutton and Barto, 2018) and is a natural choice for learning task-oriented dialog policies, where the user is modeled as the interactive environment. Offline RL optimizes the policy from the fixed annotated dataset without online environment interaction (Zhou et al., 2017; Jeon and Lee, 2022) but only partially exploits the power of RL. Online RL requires interaction with real humans or user simulators during training. However, building a good user simulator is as challenging as designing a dialog agent, either rule based (Schatzmann et al., 2007) or data driven (Gür et al., 2018; Kreyssig...
et al., 2018; Tseng et al., 2021). There also have been some efforts to jointly optimize end-to-end dialog system (DS) and user simulator (US), but most are based on traditional architectures of using LSTM seq2seq networks (Liu and Lane, 2017b; Papangelis et al., 2019; Tseng et al., 2021).

Inspired by the recent progress of finetuning pretrained LMs such as GPT-2 to develop the end-to-end trainable DS, in this paper we are firstly interested in building a GPT-2 based end-to-end trainable US for online RL of DS, which has not ever been explored. Further, note that how to develop jointly optimized GPT-2 based DS and US in the RL framework is unclear, which requires new design of model architectures. Regarding this, we aim to develop Jointly Reinforced User simulator and task-oriented Dialog system (JRUD), leveraging the recent progress of using pretrained LMs such as GPT-2 as the backbone.

To be clear, GPT-2 (Radford et al., 2019) in this paper refers to the particular class of causal LM, which computes conditional probabilities for next-token generation via self-attention based Transformer neural network (Vaswani et al., 2017). The basic idea in finetuning pretrained GPT-2 to build the dialog agent is to utilize the generation ability empowered by the finetuned causal LM. Given a particular form of conditional model, \( p(\text{output}|\text{input}) \), where \( \text{input} \) and \( \text{output} \) are token sequences, the GPT-2 LM can be finetuned over training samples \( (\text{input}, \text{output}) \) (often referred to as training sequences (Hosseini-Asl et al., 2020)), and after finetuning, the model can be used for generation, i.e., generating \( \text{output} \) after receiving \( \text{input} \).

A limitation of previous methods in GPT-2 based DS, e.g., SimpleTOD (Hosseini-Asl et al., 2020), SOLOIST (Li et al., 2020), AuGPT (Kulhánek et al., 2021) and UBAR (Yang et al., 2021), is that the whole history is used as the input at each turn. This significantly increases the memory and computation cost in both training and generation. Moreover, using the whole history may burden the model with redundant information and hurts the training efficiency. To address the aforementioned limitation and to facilitate the development of JRUD, we propose Simplified Generative Architectures (SGA) for DS and US respectively, both based on GPT-2 but using shortened history.

The main contributions of this work can be summarized as follows:

- Our DS with the proposed SGA, called SGA-DS, when only supervised trained, achieves state-of-the-art performance on MultiWOZ2.1 (Eric et al., 2020) and is more compute-efficient in both training and generation.

- To the best of our knowledge, our US with the proposed SGA, called SGA-US, represents the first GPT-2 based end-to-end trainable US, which could be trained via SL or RL.

- Based on the proposed DS and US, we successfully develop a RL framework, called SGA-JRUD, for building jointly reinforced user simulator and dialog systems, which can be interacted and trained via online RL to significantly improve the performance of the TOD system, as shown in extensive experiments on MultiWOZ2.1.

2 Related Work

End-to-end TOD systems The methodology for building TOD systems is gradually advancing from separate training of individual modules (Mrkšič et al., 2017; Wen et al., 2017a) to the end-to-end (E2E) trainable approach (Wen et al., 2017b; Liu and Lane, 2017a; Lei et al., 2018). Recent studies have exploited the large-scale pre-trained language model such as GPT-2 for building end-to-end TOD systems, e.g., SimpleTOD (Hosseini-Asl et al., 2020), SOLOIST (Li et al., 2020), AuGPT (Kulhánek et al., 2021) and UBAR (Yang et al., 2021). While existing GPT-2 based TOD systems achieve improved performance, these models mostly employ the whole dialog history as input during training and generation, which brings inefficiencies in computation, memory and learning. It is shown in Sec. 4.7 that earlier history beyond the previous turn are in fact weakly attended to in next-token generation. In contrast, the simplified architecture proposed in our SGA-DS only uses the belief state and system response of the previous turn for generating the response in current turn.

RL in TOD systems and user simulators Reinforcement learning, which aims to train an agent towards maximizing long-term cumulative rewards from interactions between the agent and its environment, could be divided in two classes, offline and online (Sutton and Barto, 2018). Both classes have been applied in TOD systems. Offline RL only optimizes the dialog agent over fixed collected data and
thus avoids building user simulators (Zhou et al., 2017; Zhao et al., 2019; Jeon and Lee, 2022). On-line RL, instead, needs to design a user simulator (US) and let the dialog agent interact with the user simulator (acting as the environment) to generate new dialogs, over which the dialog agent can be further optimized. A variety of user simulators have been studied, either rule based or data driven. A typical example of rule based US is the agenda-based user simulator (ABUS) (Schatzmann et al., 2007). In the data driven US approach, different models are proposed to train USs from data using different architectures, e.g. GRU seq2seq (Gür et al., 2018) LSTM seq2seq (Kreyssig et al., 2018).

In this paper, motivated by the recent success of GPT-2 based DS, we propose a new GPT-2 based US and further design its simplified generative architecture.

**Joint training of DS and US** There have been some studies to jointly optimize end-to-end DS and US, but most are based on traditional architectures of using LSTM seq2seq networks (Liu and Lane, 2017b; Papangelis et al., 2019; Tseng et al., 2021). Earlier studies use template-based NLG module for both DS and US (Liu and Lane, 2017b) and work in single domain such DSTC2 (Liu and Lane, 2017b; Papangelis et al., 2019). Progress has been made to use neural network based generation and work in multi-domain (Tseng et al., 2021). Different RL algorithms have been attempted such as policy gradient and actor-critic in (Liu and Lane, 2017b), Q-learning in (Papangelis et al., 2019). This paper represents a further advance in designing GPT-2 based DS and US with new simplified architectures.

### 3 Method

In the following, we first introduce the background, then the simplified generative architectures (SGA) proposed for dialog system (DS) and user simulator (US), finally we describe the jointly reinforced method.

#### 3.1 Background

**Notations** According to the information flow in a task-oriented dialog as illustrated in Figure 1, we let $g_t$ denote the user goal state, $u_t$, $b_t$ the user utterance, $b_t$ the belief state, $d_b$ the database result, $a_t$ the system act and $r_t$ be the delexicalized response, respectively, at turn $t = 1, \cdots , T$, for a dialog of $T$ turns. In this work, all these variables are tokenized into token sequences, following recent studies in (Zhang et al., 2020; Yang et al., 2021). $\oplus$ denotes the concatenation of sequences such as in $u_t \oplus r_t$, $|u_t|$ denotes the length of $u_t$ in tokens. $\{u, r\}_t$ is a shorthand for $u_t, r_t$, and $\{u, r\}_{1:t}$ represents $\{u, r\}_1, \cdots , \{u, r\}_t$.

**Dialog system (DS)** The main task for DS is, for each dialog turn $t$, to generate (or say, predict)$^1$ $b_t$, $a_t$ 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 (belief state tracking, action and response generation) can be unified into a single sequence generation problem, which can be accomplished by a causal language model (Hosseini-Asl et al., 2020; Yang et al., 2021). Particularly, pretrained LM such as GPT-2 is finetuned to yield the conditional generation model $p(output|input)$, where *input* and *output* are token sequences with appro-

---

$^1$Note that database result $d_b$ is deterministically obtained by querying database using the predicted $b_t$. We omit $d_b$ in the discussion for simplicity.
p(\text{output}|\text{input})$, as long as it can perform the required prediction. For example, SimpleTOD (Hosseini-Asl et al., 2020) generates according to a turn-level model \( p(b_t, a_t, r_t|\{u, r\}_{1:t-1}, u_t) \), i.e., \( \{u, r\}_{1:t-1}, u_t \) are concatenated as the input and recursively generate the output, i.e., \( b_t, a_t, r_t \) as shown in Fig. 2(c). UBAR (Yang et al., 2021) generates according to a session-level model, as illustrated in Fig. 2(d).

### 3.2 Simplified Generative Architecture (SGA)

A drawback with existing GPT-2 based TOD systems is that they mostly employ the whole dialog history as input during training and generation, as shown in Fig. 2 for the examples of SimpleTOD and UBAR. This brings inefficiencies in memory, computation and learning. It is shown later in Table 2 and Figure 3 that the memory cost in training and the time for running the DS to complete a dialog can be significantly reduced, if we use a shortened history as input in the DS model, while achieving state-of-the-art result on MultiWOZ2.1. It is also shown in (Jeon and Lee, 2022) that the use of the whole dialogue history increases the training cost. To address the aforementioned drawback and to facilitate the development of JRUD, we propose Simplified Generative Architectures (SGA) for both DS and US, as shown in Fig. 2(a) and (b) respectively, both based on GPT-2 but using shortened history.

**SGA-DS** For DS to predict \( b_t, a_t \) and \( r_t \) at each turn \( t \), we propose to use only the belief state \( b_{t-1} \) and response \( r_{t-1} \) from previous turn along with current user utterance \( u_t \), instead of using the whole dialog history, as the conditioning input. Presumably, this is reasonable since, by definition, belief state \( b_{t-1} \) is generally a summary of dialog history up to turn \( t-1 \), which, together with \( r_{t-1} \) and \( u_t \), should carry enough context information for the DS model to make prediction for \( b_t, a_t \) and \( r_t \). Thus, we obtain our conditional model for DS, referred to as SGA-DS, which can be expressed as \( p_\theta(b_t, a_t, r_t|b_{t-1}, r_{t-1}, u_t) \) and parameterized by \( \theta \).

In supervised learning, SGA-DS can be finetuned from pretrained GPT-2 by maximizing the following conditional likelihood:

\[
J_{\text{DS-SL}} = \log p_\theta(b_t, a_t, r_t|b_{t-1}, r_{t-1}, u_t) = \sum_{i=1}^{310} \log p_\theta(c_i|b_{t-1}, r_{t-1}, u_t, c_{<i})
\]

where \( c_i \) denotes the \( i \)-th token in \( b_t \oplus a_t \oplus r_t \).

**SGA-US** In the information flow shown in Fig. 1, user goal state \( g_t \) and user act \( u_{a_{t-1}} \) are introduced for building user simulator. User goal refers to the predefined user task such as booking a cheap hotel, which is directly obtained from the annotation of the dataset; and the goal state represents the uncompleted part of the user goal. Both are represented by token sequences in this work. The main task of US is to mimic an user, i.e., given the dialog history, to decide user act, generate user utterance, and update internal goal state to track progress towards satisfying the user goal.

In this work, we find that the approach of fine-tuning pretrained GPT-2 for conditional generation can be similarly applied to build US. Particularly, for US to predict \( u_{a_{t-1}} \) and \( u_{t} \) at each turn \( t \), we propose to use the previous response \( r_{t-1} \) and current goal state \( g_t \), as the conditioning input. The goal state \( g_t \) is obtained by removing the slot values of previous user act \( u_{a_{t-1}} \) from previous goal state \( g_{t-1} \). Thus, we obtain the conditional model for US, referred to as SGA-US, which can be expressed as \( p_\phi(u_{a_t}, u_t|r_{t-1}, g_{t}) \) and parameterized by \( \phi \). In supervised learning, SGA-US can be finetuned from pretrained GPT-2 by maximizing the following conditional likelihood:

\[
J_{\text{US-SL}} = \log p_\phi(u_{a_t}, u_t|r_{t-1}, g_{t}) = \sum_{i=1}^{307} \log p_\phi(c_i'|r_{t-1}, g_{t}, c_{<i}')
\]

where \( c_i' \) denotes the \( i \)-th token in \( u_{a_t} \oplus u_t \).

### 3.3 Jointly reinforced US and DS (JRUD)

After we design the architectures for DS and US, we can consider joint optimization of the two agents in the online RL framework. The DS agent view the US as the environment and use its conditional model \( p_\theta(b_t, a_t, r_t|b_{t-1}, r_{t-1}, u_t) \) as its policy. Conversely, the US agent view the DS as...
the environment and use its conditional model $p_\theta(u_{at}, u_t|r_{t-1}, g_t)$ as its policy. Here the policy of SGA-DS involves generating not only system act $a_t$, but also belief state $b_t$ and system response $r_t$. This is different from some previous studies of learning reinforced DS, e.g., (Liu and Lane, 2017b; Papangelis et al., 2019; Tseng et al., 2021), which only use RL to optimize the selection of system acts (but all use traditional architectures). The action space of SGA-DS becomes larger, but thanks to the representation power of GPT-2, recursively predict (or say, decide about) $b_t, a_t$ and $r_t$ in one policy yields the best performance in our experiment. In Sec 4.6.2, we compare different schemes for policy definition for the DS agent with more discussions.

After supervised finetuning of DS and US separately, we apply the policy gradient method (Sutton et al., 2000) to jointly optimize the two agents. We first let the two agents interact with each other based on the user goals sampled from training set and generate mini-batches of dialogs. Then we calculate the reward $R_t$ for each turn, which is described in detail in Sec 4.6.1. The return $U_{i,t}$ for the action of turn $t$ at the $i$-th step is $\gamma^{k-t}R_t$, where $\gamma$ is the discounting factor and $|A_t|$ is the policy sequence length of turn $t$. We update the two agents with the following policy gradients:

$$\nabla_\theta J_{DS-RL} = \sum_{i=1}^{|u_{at} \oplus u_t|} U_{i,t} \nabla \log p_\theta(c_i)$$

$$\nabla_\phi J_{US-RL} = \sum_{i=1}^{|u_{at} \oplus u_t|} U_{i,t} \nabla \log p_\phi(c_i')$$

where $p_\theta(c_i)$ and $p_\phi(c_i')$ are shorthands for $p_\theta(c_i|b_{t-1}, r_{t-1}, u_t, c_{<i})$ and $p_\phi(c_i'|r_{t-1}, g_t, c_{<i})$, respectively.

4 Experiments

4.1 Dataset

We use MultiWOZ2.1 (Eric et al., 2020) for experiments. MultiWOZ2.1 is a large-scale English multi-domain task-oriented dialog datasets of human-human conversations. It contains 10.4k multi-turn dialogs, spanning over seven domains. In our experiments, we removed some inappropriate state values and corrected some spelling errors in the dataset, which is detailed in Appendix A.

4.2 Evaluation

Plenty of methods have been tested on MultiWOZ2.0 or MultiWOZ2.1, but may suffer from the inconsistencies in evaluation, which is analyzed in Nekvinda and Dušek (2021). To rigorously compare our model with others, we use their standardized evaluation scripts, which are now also the scripts adopted in the MultiWOZ website. There are mainly four metrics for offline evaluation (corpus-based evaluation). 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 responses. And the Combined Score is computed as (BLEU + 0.5 * (Inform + Success)). We also use the joint goal accuracy to evaluate DST performance, which is the proportion of dialog turns where all slot values are correctly predicted. Noting that when performing online evaluation, i.e., evaluating the interaction quality of two agents, only Inform and Success can be calculated and the above scripts are no longer applicable, so we calculate Inform and Success rate using the scripts of Tseng et al. (2021). To compare results, we conduct significance test for Success Rate, Inform Rate, and BLEU using matched pairs test (Gillick and Cox, 1989) and report the p-value.

4.3 Training Procedure

We first train the DS and US separately on training set based on the SL objective described in Eq. 1 and Eq. 2. The resulting models are referred to as SGA-DS-SL and SGA-US-SL. Then we conduct RL experiments through the interaction between the two agents. During interaction, we end the dialog according to the following conditions: 1) the number of dialog turns exceeds the threshold (20 in our experiment); 2) the goal state of US is empty; 3) the two agents generate ending intentions such as bye and thank concurrently.

To ameliorate the non-stationarity problem when jointly training the two agents (Liu and Lane, 2017b), we first fix the DS and optimize the US for 100 training cycles (each cycle contains 128 episodes) and obtain the model SGA-US-RL. Then we fix SGA-US-RL and optimize the DS for another 100 training cycles and obtain the model SGA-JRUD. In order to show the effect of joint optimization, we fix US and only optimize DS
for 100 cycles and obtain SGA-DS-RL for comparison. Remarkably, to avoid our dialog system deviating from natural language, we alternate RL updates with supervised learning at a certain ratio (Lewis et al., 2017), which is set to be 1:1 in our experiments. More implementation details in our experiments are available in Appendix B.

### 4.4 Supervised Benchmark Results

We first show the offline evaluation results of different supervised trained DSs, which can be seen in Table 1. We evaluate different dialog systems in an end-to-end setting, which means that the generated belief states and system acts are used in response generation, i.e., the variables from the previous turn ($b_{t-1}$ and $r_{t-1}$) when used as the conditioning input are also the generated ones from the model itself. We can see that when only supervised trained, SGA-DS-SL achieves the highest combined score among all the GPT-2 based supervised DS models, which indicates the superiority of our proposed SGA in building DS.

Moreover, as shown in Figure 3 and Table 7 in Appendix C.1, SGA-DS-SL is more compute-efficient than SimpleTOD and UBAR. Since SGA-DS uses shortened history in training sequences, the training sequences for SGA-DS-RL are generally much shorter than for SimpleTOD and UBAR. Consequently, SGA-DS-SL consumes less training time and achieves faster generation speed. These experiments are all conducted on a single 16GB Tesla-P100 GPU.

### 4.5 RL Results

To evaluate the RL experiments, we perform online evaluation where we let the two agents interact with each other for 1k times using the user goals of the test set and calculate the inform and success rate of generated dialogs. Following Shi et al. (2019), we show the interaction results between different agents, which are shown in Table 2. In Table 2, the inform and success rate increase significantly after RL. The jointly reinforced DS SGA-JRUD and SGA-US-RL achieve the highest success rate, which is higher than the supervised models (SGA-DS-SL and SGA-US-SL) by almost ten points. The substantial improvement implies that the new data generated during the interaction can really enhance our models. Moreover, we can see that SGA-JRUD obtains a higher success rate than SGA-DS-RL. This result indicates that the benefit of joint learning is that the US can improve its policy through the interaction between two agents, so that the DS learns better through the interaction with a better US. The improvement of the US is also reflected in the evaluation result between SGA-DS-RL and SGA-US-RL. The two

---

**Table 1:** Offline evaluation results on MultiWOZ2.1. Above the dashed line are supervised learning (SL) models and below are RL models. The unbolded results are cited from the official website of MultiWOZ, which uses the same evaluation scripts of Nekvinda and Dušek (2021) as in our experiments.

<table>
<thead>
<tr>
<th>Model</th>
<th>Inform</th>
<th>Success</th>
<th>BLEU</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuGPT (Kulhánek et al., 2021)</td>
<td>76.6</td>
<td>60.5</td>
<td>16.8</td>
<td>85.4</td>
</tr>
<tr>
<td>SOLOIST (Li et al., 2020)</td>
<td>82.3</td>
<td>72.4</td>
<td>13.6</td>
<td>90.9</td>
</tr>
<tr>
<td>UBAR (Yang et al., 2021)</td>
<td>83.4</td>
<td>70.3</td>
<td>17.6</td>
<td>94.4</td>
</tr>
<tr>
<td>SGA-DS-SL</td>
<td>84.90</td>
<td>71.50</td>
<td>18.14</td>
<td>96.34</td>
</tr>
<tr>
<td>SGA-DS-RL</td>
<td>82.30</td>
<td>70.70</td>
<td>19.89</td>
<td>96.39</td>
</tr>
<tr>
<td>SGA-JRUD</td>
<td>85.00</td>
<td>74.00</td>
<td>19.11</td>
<td>98.61</td>
</tr>
</tbody>
</table>

**Table 2:** Online evaluation results between DSs and USs. Inform and Success rate are obtained by having the user simulator interacting with the dialog system on 1k user goals from the test corpus.

<table>
<thead>
<tr>
<th>DS</th>
<th>US</th>
<th>Inform</th>
<th>Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGA-DS-SL</td>
<td>SGA-US-SL</td>
<td>87.0</td>
<td>83.0</td>
</tr>
<tr>
<td>SGA-DS-RL</td>
<td>SGA-US-RL</td>
<td>90.0</td>
<td>86.5</td>
</tr>
<tr>
<td>SGA-JRUD</td>
<td>SGA-US-RL</td>
<td>93.0</td>
<td>90.1</td>
</tr>
<tr>
<td>SGA-US-RL</td>
<td>SGA-US-SL</td>
<td>87.0</td>
<td>84.1</td>
</tr>
<tr>
<td>SGA-US-SL</td>
<td>SGA-US-RL</td>
<td>95.0</td>
<td>92.9</td>
</tr>
</tbody>
</table>
models are never been trained together during RL, but they achieve pretty high success rate in the evaluation.

In Figure 4, we also plot the learning curves of SGA-JRUD. In the first 100 training cycles, we fix DS and only optimize US. In the last 100 cycles, we fix the trained US and optimize DS only. We evaluate the two agents after every training cycle on the user goals of dev set. To reduce time cost, the dev set contains only 200 user goals randomly drawn from the original training and validation set before the running of RL. From Figure 4, we can see that inform and success rate are both consistently improved during RL and finally converge to an upper bound in the second stage.

Notably, it is observed in Kottur et al. (2017) that the improvement of success rate does not necessarily mean that the two agents can understand the semantic interaction, but may just invent an uninterpretable language. To address this concern, we conduct offline evaluation to see if our dialog system deviates from natural language after RL. The results are already included in Table 1. We can see that the BLEU scores do not decrease for reinforced DS models (SGA-DS-RL, SGA-JRUD). This indicates that our reinforced DS models can generate interpretable natural language after RL.

It can be also seen from Table 1, RL further improves the offline evaluation performance of SGA-DS in task completion. As shown in Table 3, the jointly reinforced model (SGA-JRUD) significantly improves over UBAR (p-value<0.02) and SGA-DS-SL (p-value<0.04) in Success Rate.

4.6 Analysis and Ablation Study

4.6.1 Different reward settings

A number of different settings for reward have been studied, as described in the following.

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 global reward (proportions of tasks completed) of each agent;
3) A Sigmoid synthetic reward obtained by mapping the synthetic reward to [0,1] interval using Sigmoid function.

The above third setting is designed to exclude the influence of the value range of reward because the value range is different between the Success and the synthetic reward. It is found that using the first setting for award (i.e., 0 or 1 for each dialog according to Success) produces the best results in our experiments. All RL results in this paper are based on using this setting of reward, unless here for ablation study. The results of using different reward settings are reported in Table 4. We can see that all reward settings achieve better results than supervised baseline (Reward=None) and setting Success as reward achieves the best result.

4.6.2 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, 2017b; Papangelis et al., 2019; Tseng et al., 2021), mainly employ RL to optimize the DP module, i.e., use system acts for actions. In contrast, the policy of SGA-DS involves generating not only system act \( a_t \), but also belief state \( b_t \) and system response \( r_t \). To compare policy schemes for reinforced DS, we
conduct two other RL experiments, where the policy includes $a_t$ and $a_t \oplus r_t$ respectively. We show the online and offline evaluation results in Table 5.

It can be seen from Table 5 that using $b_t \oplus a_t \oplus r_t$ for policy achieves the highest online evaluation results with large margins. In offline evaluation, using $b_t \oplus a_t \oplus r_t$ is also among the best. Using $a_t$ achieves higher combined score, but the difference is not significant (p-value=0.355). We provide two points, which may explain the advantage of our model in using $b_t \oplus a_t \oplus r_t$ 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 with RL without considering other modules may mislead other modules. Using $b_t \oplus a_t \oplus r_t$ leads to better overall optimization and decision-making. Second, the conflict 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.

### 4.7 Attention Weight Statistics

In SGA, we propose that $b_{t-1}$, $r_{t-1}$ and $u_t$ could be sufficient for the DS to generate $b_t$, $a_t$ and $r_t$, and the whole dialog history contains redundancy. To support this idea, we calculate the average attention weights for prediction at a certain turn $t$ which point to the variables in previous all turns, using the session-level model UBAR (Yang et al., 2021). Let $t = 4$ and we show the means of the attention weights in generating $b_4$ in Figure 5(a).

We can see that in UBAR, belief state $b_4$ mainly attends to current user utterance ($u_4$) and belief states of all previous turns ($b_1$, $b_2$ and $b_3$). Note that belief state is defined as the accumulation of history information, which means that $b_4$ contains almost all the slots and values of $b_1$ and $b_2$. Thus, the attentions to $b_1$ and $b_2$ are redundant, they appear mainly because there are no mechanism to reduce such redundancy in previous models. In Appendix C.3, we provide an example to show how UBAR attends to previous belief states and we can see that the tokens in $b_1$ and $b_2$ with large attention weights are almost all appeared in $b_3$. If we do not let the model attend to $b_1$ and $b_2$, the model will naturally attend more to $b_3$ and still not miss the information in $b_1$ and $b_2$. This can be indeed seen from the attention weights of our proposed SGA-DS, as shown in Figure 5(b). Different from UBAR, SGA-DS-SL attends more to $b_3$ than to $u_4$ when generating $b_4$. In UBAR, the attentions are scattered across $b_1$, $b_2$, $b_3$. We also show the means of attention weights in predicting system act and response in Appendix C.2. As expected, the attentions mainly point to the variables of current turn, especially the history variables closely nearby.

### 5 Discussion and Conclusion

In this paper, we first propose Simplified Generative Architectures (SGA) for DS and US respectively, both based on GPT-2 but using shortened history. Then, we successfully develop Jointly Reinforced US and DS, called SGA-JRUD. The supervised trained DS with the proposed SGA achieves state-of-the-art performance on MultiWOZ2.1 and is more compute-efficient in both training and generation. To develop and demonstrate JRUD, extensive experiments on MultiWOZ2.1 are conducted with both offline and online evaluations, we study different reward settings, different policy schemes. More discussions are provided in Appendix C.4, C.5, C.6 on exploration of actions, example of improvement, examples of generated dialogs about our models, respectively.

This work represents a new step towards jointly reinforced end-to-end US and DS, but its performance may be limited by the pretrained GPT-2 backbone and the policy gradient algorithm used. Attempting larger backbone and new RL algorithms are interesting further directions.
References


Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequnicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In 56th Annual Meeting of the Association for Computational Linguistics (ACL).


A Dataset Cleaning

The details of how we clean up the dataset can be seen in Table 6. Specifically, when training with delexicalized responses, the belief states of some turns become incorrect because they contain a redundant slot `name` and its corresponding value. These slots and values are originally from some lexicalized responses in the belief state. Another change is to correct some spelling errors of the word `portuguese` in belief states and user utterances.

<table>
<thead>
<tr>
<th>Correction</th>
<th>Example</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belief state</td>
<td>[hotel] stars 4 internet yes name cambridge belfry</td>
<td>24091</td>
</tr>
<tr>
<td>Spelling</td>
<td>portuguese → portuguese</td>
<td>415</td>
</tr>
</tbody>
</table>

Table 6: Correction of the MultiWOZ2.1.

B Implementation Details

We implement the models with Huggingface Transformers repository of version 3.5.1. We initialize SGA-DS and SGA-US with DistilGPT-2 (Sanh et al., 2019), a distilled version of GPT-2. During supervised pre-training, we use AdamW optimizer and a linear scheduler with 20% warm up steps and maximum learning rate $1e^{-4}$. The minibatch base size is set to be 8 with gradient accumulation steps of 4. The total epochs are 50 and we monitor the performance on validation set and apply early stopping (stop when the current best model keeps the best for subsequent 4 epochs). We select the best model on the validation set then evaluate it on test set. During RL, we no longer use scheduler and fix the learning rate to $2e^{-5}$. The batch size is set to be 16 with gradient accumulation steps of 12. As described in Sec 4.3, we first optimize US for 100 training cycles and select the US with best online evaluation result. Then we optimize DS for another 100 training cycles with the selected US and select the best DS. Thanks to our proposed simplified structure, all the experiments above can be performed on a single 16GB Tesla-P100 GPU.

C Analysis and Case Study

C.1 Time Costs

We use re-implementation of SimpleTOD and UBAR under the same optimizer and scheduler as SGA-DS-SL; The minibatch base sizes and gradient accumulation steps are (8,4) for SGA-DS-SL, and (2, 16) for SimpleTOD and UBAR, respectively. We monitor the performance on validation set and apply early stopping (stop when the current best model keeps the best for subsequent 4 epochs).

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time</th>
<th>Generation Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGA-DS-SL</td>
<td>204min 229s</td>
<td></td>
</tr>
<tr>
<td>SimpleTOD</td>
<td>396min 276s</td>
<td></td>
</tr>
<tr>
<td>UBAR</td>
<td>369min 352s</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Time costs of different models. The right two columns report the training time and the time of generation on test set respectively.

C.2 Attention Weights

We show the statistical results for system act and response in UBAR, which are shown in Figure 6. It can be observed that the two variables almost only attend to the variables of current turn, especially the history variables closely nearby. For instance, system act $a_t$ most attends to the database result $db_t$ and response $r_t$ most attends to the system act $a_t$. This is reasonable because TOD systems always make decisions based on what the system finds in the database and generate a response that is highly consistent with the selected action.

C.3 Example of UBAR’s Attention

We provide an example of a training sequence in Figure 7 to show how the belief state of 4-th turn in UBAR (Yang et al., 2021) attends to belief states and user utterances of the previous three turns. As we can see, each token of belief state in the fourth turn basically gives the same token the maximum attention weight in previous belief states. But the tokens of the first two belief states are included in the third belief state, which indicates that if we constrain the model to only attend to the third belief state, it can also absorb enough information.
C.4 Exploration of Actions

In order to investigate whether our model has the ability to generate richer dialog actions after RL, we count the number of actions that have not appeared in training set during evaluation (unseen acts). The statistics can be seen in Table 8. It can be seen that the number of both unseen system acts and unseen user acts has increased after RL. RL does improve the richness of dialog actions to a certain extent. However, the total unseen system acts in the original test set is 556, much larger than the number of generated unseen system acts. This means that although the diversity of dialog actions has improved, it is not enough and needs further study.

C.5 Example of improvement from RL

To help understanding the improvement from RL, we show some examples in Table 9. We selected two dialogs in test set and compare the system act and delexicalized response generated by the supervised model SGA-DS-SL and the RL model SGA-JRUD.

From the first example, we can see that SGA-DS-SL forgets to inform some important attributes such as address, while SGA-JRUD tends to generate more attributes to accomplish tasks. In the second example, SGA-DS-SL finishes booking when the user has not provided enough information, which leads to a wrong response in the next turn. However, SGA-JRUD learns to ask the user more information when the booking cannot be completed, and to complete the booking once the information is enough.

We also provide some dialogs generated during the interaction between the US and DS agents, which can be seen in Appendix C.6.

C.6 Example of Generated Dialogs

We provide a dialog sample generated during the interaction between SGA-JRUD and SGA-US-RL in Table 10. We can see that there’s a big difference between the generated dialog and the original dialog, very conducive to model learning.
User: I am looking for some info on an attraction called Kettle's Yard.

Resp: Absolutely! The entrance is [price]! It is located in the [area], the address is [address]!

User: Sup [attraction] [inform] price area

Resp: Sup [attraction] [inform] price area.

User: Sup [attraction] [inform] price area name type address

Resp: Sup [attraction] [inform] price area name type address.

User: I'll be staying for 3 nights starting on Tuesday.

Resp: For how many guests please, sir?

User: I'll be staying for 3 nights.

Resp: Booking was successful. Reference number is: [reference].

User: How many people will be staying?

Resp: How many people will be staying?

User: 1 guest for 3 nights.

Resp: Booking was successful. Reference number is: [reference].

User: What about a museum?

Resp: Yes, there are [choice] [type]. I would recommend [name]. It is at [address] and has [price] entrance. Would that work for you?

User: Yes, that will work great. Can I get their phone number please?

Resp: Certainly. [name]'s phone number is [phone]. Is there anything else I can help you with today?

User: Thank you so much for your help.

Resp: Thank you for using our system today!

User: No, that is all I need. Thanks!

Resp: You are welcome. Have a great day. Bye.

Table 9: Two dialog examples (only 1 or 2 turns are shown) selected from test set. Act and Resp denote the system act and response from dataset. Act or Resp with the subscript sup and RL denote they are generated by SGA-DS-SL and SGA-JRUD.

Table 10: Interaction case between SGA-JRUD and SGA-US-RL. The generated dialog ends at the third turn.