DORA: Towards Policy Optimization for Task-oriented Dialogue System with Efficient Context

Hyunmin Jeon[a], Gary Geunbae Lee[a,b,]

[a] *Computer Science and Engineering, Pohang University of Science and Technology, South Korea*
[b] *Graduate School of Artificial Intelligence, Pohang University of Science and Technology, South Korea*

Abstract

Recently, reinforcement learning (RL) has been applied to task-oriented dialogue systems by using latent actions to solve shortcomings of supervised learning (SL). In this paper, we propose a multi-domain task-oriented dialogue system, called **D**ialogue System with **O**ptimizing a **R**ecurrent **A**ction Policy using Efficient Context (DORA), that uses SL, with subsequently applied RL to optimize dialogue systems using a recurrent dialogue policy. This dialogue policy recurrently generates explicit system actions as a both word-level and high-level policy. As a result, DORA is clearly optimized during both SL and RL steps by using an explicit system action policy that considers an efficient context instead of the entire dialogue history. The system actions are both interpretable and controllable, whereas the latent actions are not. DORA improved the success rate by 6.6 points on MultiWOZ 2.0 and by 10.9 points on MultiWOZ 2.1.

Keywords: Task-oriented dialogue system, Multi-domain dialogue, Policy optimization, Recurrent Action Policy, Efficient context

1. Introduction

Task-oriented dialogue systems are designed to use conversations to help users achieve goals in specific domains. In multi-domain dialogues, users have goals across multiple domains, so to respond adequately, the systems should track the flow of conversation among domains. This necessity complicates development of task-oriented dialogue systems that can process multi-domain dialogues.

Typical dialogue systems have a pipeline architecture that consists of a natural language understanding (NLU) module, belief tracker, dialogue policy, and natural language generation (NLG) module. Recent work has focused on neural dialogue systems that are end-to-end trainable by using supervised learning (SL) (Zhang et al., 2020b; Hosseini-Asl et al., 2020; Peng et al., 2020; Yang et al., 2020). SL is an efficient method to train neural networks, but use of SL to train task-oriented dialogue systems has some limitations. First, training using SL requires annotations, which can be erroneous. Furthermore, task-oriented dialogues do not have a definite optimal answer; many options must be considered when formulating an answer to a user utterance. Thus, the systems can become biased by the annotations. To address these problems, several approaches have attempted to train dialogue systems by using reinforcement learning (RL) to optimize dialogue policy that samples latent variables as actions (Zhao et al., 2019; Wang et al., 2020; Lubis et al., 2020; Lee et al., 2020).

Even though use of latent actions has improved the success rate of dialogue systems, the strategy has several limitations in task-oriented dialogue systems. Training the systems from scratch using RL is almost impossible, so they are generally pre-trained using SL. However, latent actions have no gold-standard outputs; therefore, the NLG module, which follows the dialogue policy, cannot be given clear prior information for response generation during the SL step. Furthermore, humans cannot interpret latent actions; this weakness complicates the task of determining the appropriate action space of latent variables, and of judging whether the latent variables represent this space well.

Another drawback of previous methods is that they use the entire dialogue history as the system input. Use of large-scale pre-trained language models, such as BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019), has

E-mail: jhm9507@postech.ac.kr (Jeon), gblee@postech.ac.kr (Lee).

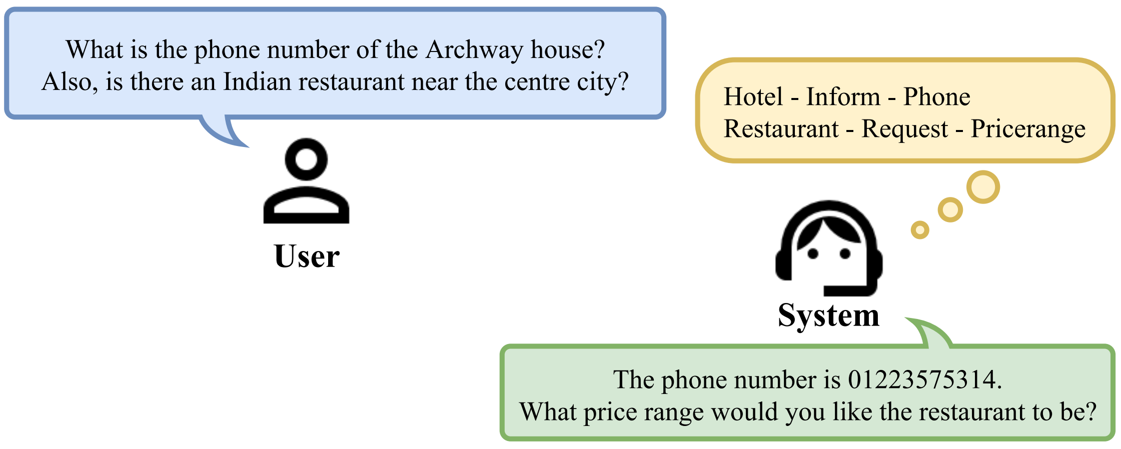

What is the phone number of the Archway house?
Also, is there an Indian restaurant near the centre city?
User
Hotel - Inform - Phone
Restaurant - Request - Pricerange
System
The phone number is 01223575314.
What price range would you like the restaurant to be?

Figure 1: Two system actions across two domains and a corresponding system response to respond to a complex utterance in a multi-domain dialogue.

been a trend in NLP field and has improved the success rate of dialogue systems, but it significantly increases the model size. Even with the same increase in input length, the memory usage increases faster on larger model, and the dialogue history lengthens as the conversation progresses, so use of the dialogue history increases training cost.

In this study, we propose **D**ialogue System with **O**ptimizing a **R**ecurrent **A**ction Policy using Efficient Context (DORA), a task-oriented dialogue system that uses explicit system actions instead of latent actions, and summarized input instead of dialogue history, to address the above limitations. In multi-domain dialogues, systems should generate multiple system actions at once to respond to complex user utterances. DORA processes this by recurrently generating system actions, not just choosing one action. Figure 1 shows an example of this process.

Use of explicit system actions instead of latent actions enables clear optimization of an NLG module given gold-standard system actions even when the dialogue policy is not optimized, during the SL step. Furthermore, the system actions are obviously interpretable. Thus, the generated system actions can be used for reward shaping during the RL step, and the optimized dialogue policy can be controlled by post-processing for various purposes.

DORA only uses a current user utterance instead of the entire dialogue history. However, abandoning dialogue history causes losses of contextual information from previous turns. To prevent the losses, DORA also uses the domain state and belief state as input. The domain state and belief state are updated as the conversation progresses, so they can efficiently represent contextual information that is accumulated during multiple turns. Removing dialogue history makes the memory usage almost constant regardless of the length of conversations; therefore, DORA can be stably and efficiently trained even with long conversations.

We evaluated DORA on MultiWOZ 2.0 (Budzianowski et al., 2018) and MultiWOZ 2.1 (Eric et al., 2020), which are standard benchmark datasets for multi-domain task-oriented dialogue systems. DORA achieved higher success rate than previous methods. The results demonstrate that use of explicit system actions enables clear optimization of the system during both the SL and RL steps, and that the input context using the domain state and belief state can efficiently perform the role of dialogue history.

2. Related Work

Recent work on task-oriented dialogue systems has focused on building end-to-end trainable dialogue systems in multi-domain dialogues (Zhang et al., 2020b; Hosseini-Asl et al., 2020; Peng et al., 2020; Lin et al., 2020; Yang et al., 2020; Zhang et al., 2020a). Some researchers have attempted to use large-scale pre-trained language models, such as BERT and GPT-2, to transfer abundant contextual representation trained on vast corpus. BERT performs well as an NLU module by encoding the dialogue history (Lee et al., 2020). Some methods consider task-oriented dialogues as language modeling tasks using GPT-2 (Hosseini-Asl et al., 2020; Peng et al., 2020; Yang et al., 2020). Pre-trained sequence-to-sequence models, such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2019), also can

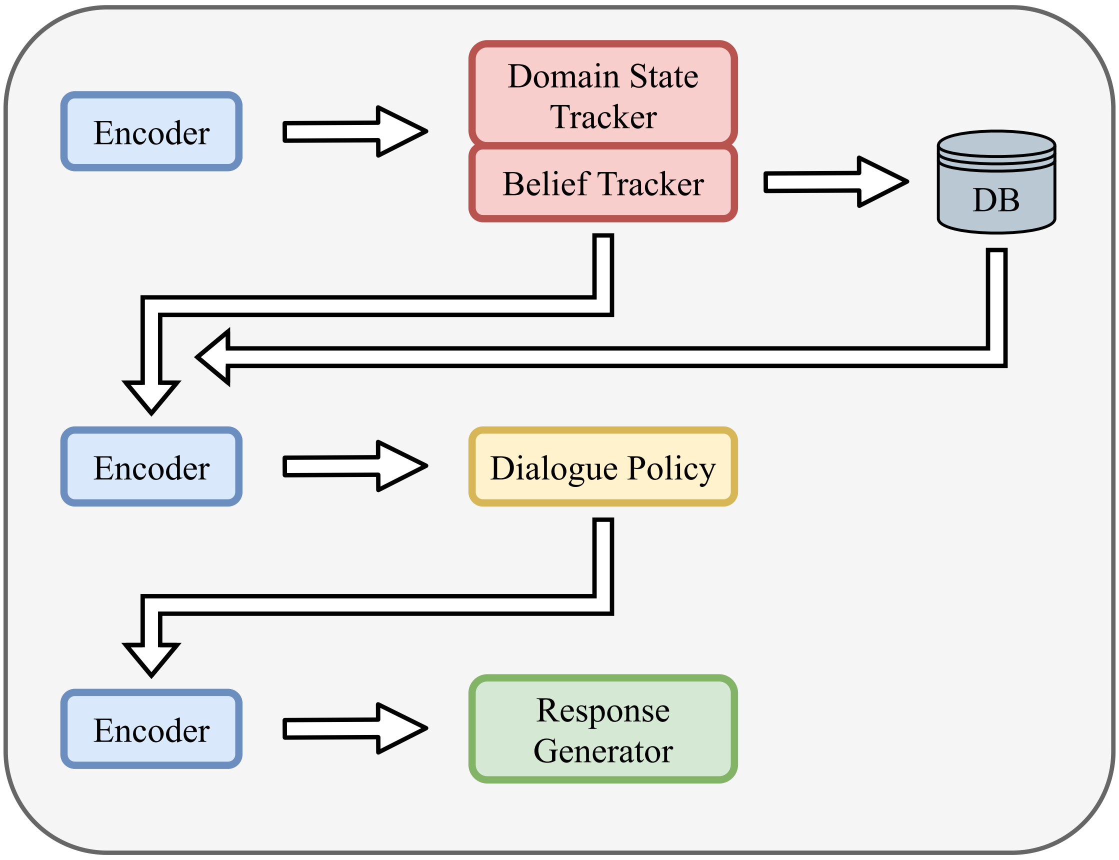


Figure 2: Architecture overview of DORA: a neural network consisting of three sequence-to-sequence sub networks with a shared encoder and a task-specific DB.

be used for task-oriented dialogue systems. BART and T5 have been evaluated as backbone models for multi-domain task-oriented dialogues (Lin et al., 2020).

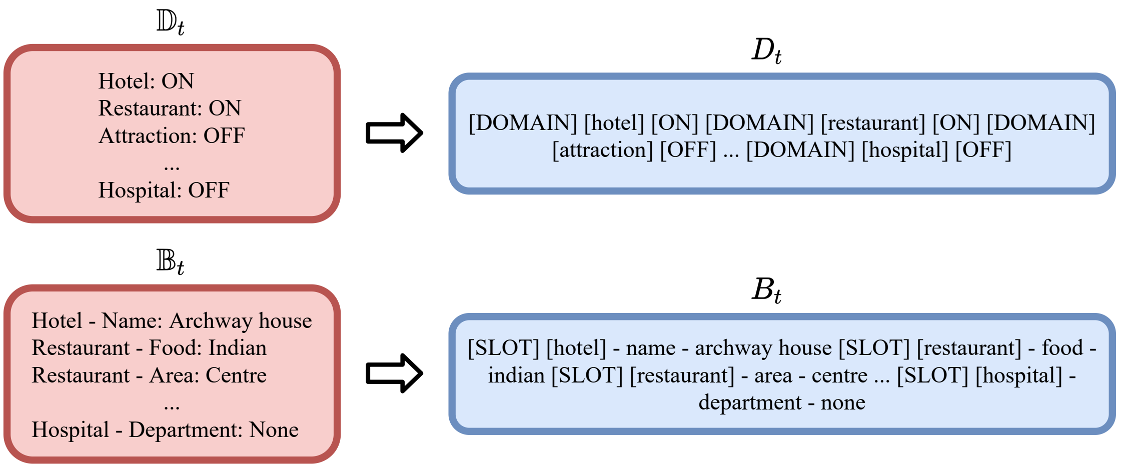
To address the limitations of SL for task-oriented dialogue systems, various approaches have used RL. To apply RL in dialogue systems, a policy sampling actions given a state should be included in the systems. The naïve approach is to treat an NLG module as a low-level policy and to use words as actions (Lewis et al., 2017; Das et al., 2017; Kottur et al., 2017). However, the low-level and word-level policies have large action space, so optimization during the RL step is difficult and inefficient. Therefore, recent studies have considered latent-level policies that sample latent variables, instead of words, as actions (Zhao et al., 2019; Wang et al., 2020; Lubis et al., 2020; Lee et al., 2020). Optimization of latent action policies as high-level policies have improved the success rate of task-oriented dialogue systems.

To reduce the burdens of using the entire dialogue history, several methods to efficiently represent the history have been proposed. One encodes utterances and efficiently accumulates latent representations of the history as the conversation progresses (Gupta et al., 2018). Another presents an efficient belief tracking system by using the previous belief state as an additional input instead of the entire history (Kim et al., 2020).

In our work, we demonstrate a method that solves both the shortcomings of latent action policies and the inefficiency of dialogue history.

3. Dialogue System with Optimizing a Recurrent Action Policy using Efficient Context

In this section, we describe the architecture (Figure 2) of DORA, and then we explain how to optimize the system and to construct the efficient context for system input. In every turn, the neural network sequentially predicts the



$\mathbb{D}_t$
Hotel: ON
Restaurant: ON
Attraction: OFF
...
Hospital: OFF
D_t
[DOMAIN] [hotel] [ON] [DOMAIN] [restaurant] [ON] [DOMAIN] [attraction] [OFF] ... [DOMAIN] [hospital] [OFF]
$\mathbb{B}_t$
Hotel - Name: Archway house
Restaurant - Food: Indian
Restaurant - Area: Centre
...
Hospital - Department: None
B_t
[SLOT] [hotel] - name - archway house [SLOT] [restaurant] - food - indian [SLOT] [restaurant] - area - centre ... [SLOT] [hospital] - department - none

Figure 3: Conversion processes of the domain state and belief state from sets to sentences. Words enclosed in brackets are special words for distinguishing from general words.

domain state and belief state, makes system actions, and generates a system response.

3.1. Context Construction

The belief state indicates the constraints of user goals; it consists of various predefined slots and their corresponding values. Tracking the belief state is an essential task for task-oriented dialogue systems that should catch user goals during long conversations. In addition to the belief state, DORA tracks the domain state, which indicates whether each domain is activated in the current conversation. By tracking the domain state, the system can trace the flow of conversation from a domain perspective, thereby simplifying the task of understanding the purpose of a user in multi-domain dialogues.

Instead of using the entire dialogue history, we efficiently construct an input context using the domain state and belief state in addition to user utterance. The context changes over three steps of domain state and belief tracking, system action generation, and system response generation. The initial context C_t^{Init} for domain state and belief tracking on turn t consists of the current user utterance U_t, the previous domain state $\mathbb{D}_{t-1}$, and the previous belief state $\mathbb{B}_{t-1}$. On current turn t, $\mathbb{D}_{t-1}$ and $\mathbb{B}_{t-1}$ have contextual information that had accumulated until the previous turn, and they can perform as system inputs instead of the dialogue history. $\mathbb{D}_{t-1}$ is a set of binary values that indicates whether each domain is activated. $\mathbb{B}_{t-1}$ is a set of values for each slot. To feed the domain state and belief state into the encoder, we convert $\mathbb{D}_{t-1}$ to D_{t-1} and $\mathbb{B}_{t-1}$ to B_{t-1}, in sentence forms by using some special words (Figure 3). C_t^{Init} is represented as

$$C_t^{Init} = \texttt{[CLS]} \oplus U_t \oplus \texttt{[SEP]} \oplus D_{t-1} \oplus B_{t-1} \oplus \texttt{[SEP]} = \left\{c_{t,1}^{Init}, \cdots, c_{t,|C_t^{Init}|}^{Init}\right\}, \tag{1}$$

where special word `[CLS]` represents the entire sentence and `[SEP]` distinguishes between the utterance and states; $\oplus$ indicates concatenation. By considering the current domain state $\mathbb{D}_t$ and belief state $\mathbb{B}_t$, the DB operator queries a task-specific DB to obtain a list of matched entries, and converts the list to sentence form DB_t (Figure 4). The belief context C_t^{Belief} for system action generation includes D_t and B_t instead of D_{t-1} and B_{t-1}, respectively, and additionally includes DB_t. C_t^{Belief} is represented as follows:

$$C_t^{Belief} = \texttt{[CLS]} \oplus U_t \oplus \texttt{[SEP]} \oplus D_t \oplus B_t \oplus DB_t \oplus \texttt{[SEP]} = \left\{c_{t,1}^{Belief}, \cdots, c_{t,|C_t^{Belief}|}^{Belief}\right\}. \tag{2}$$

The action context C_t^{Act} for system response generation additionally contains the generated system actions A_t. C_t^{Act} is represented as follows:

$$C_t^{Act} = \texttt{[CLS]} \oplus U_t \oplus \texttt{[SEP]} \oplus D_t \oplus B_t \oplus DB_t \oplus A_t \oplus \texttt{[SEP]} = \left\{c_{t,1}^{Act}, \cdots, c_{t,|C_t^{Act}|}^{Act}\right\}. \tag{3}$$

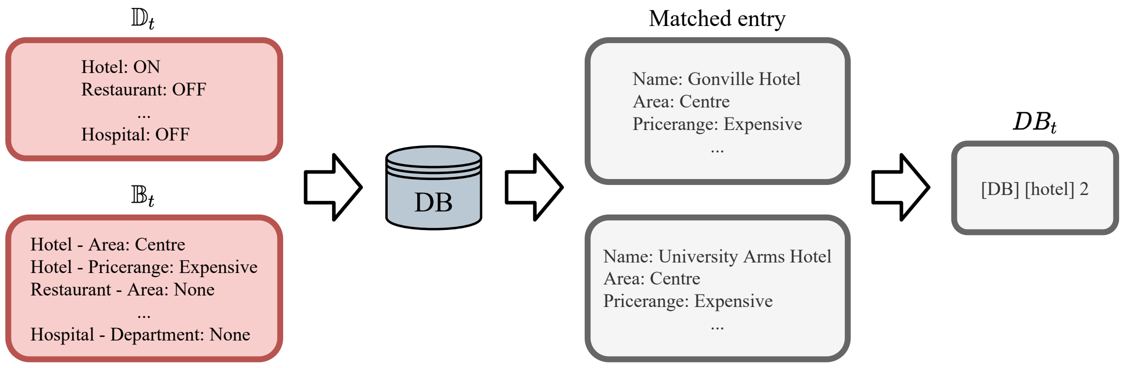


Figure 4: Process of DB search to obtain DB_t.

3.2. *Model Formulation*

We use pre-trained BERT for the shared encoder. C_t^{Init}, C_t^{Belief}, and C_t^{Act} are fed into the shared encoder to perform sequential tasks of domain state and belief tracking, system action generation, and system response generation, respectively.

3.2.1. *Domain State Tracker*

BERT encodes C_t^{Init} to vector representations H_t^{Init}, and then $h_{t,1}^{Init}$ that represents the `[CLS]` word is fed into a fully connected (FC) layer $W_{pool} \in \mathbb{R}^{dim_H \times dim_H}$. Then, a `tanh` activation function is applied to obtain a pooled output o_t^{Init} as

$$H_t^{Init} = \mathtt{BERT}\left(C_t^{Init}\right) = \left\{h_{t,1}^{Init}, \cdots, h_{t,|C_t^{Init}|}^{Init}\right\} \in \mathbb{R}^{|C_t^{Init}| \times dim_H}, \tag{4}$$

$$o_t^{Init} = \mathtt{tanh}\left(W_{pool} h_{t,1}^{Init}\right) \in \mathbb{R}^{dim_H}, \tag{5}$$

where dim_H is the hidden size. We use an FC layer $W_{domain} \in \mathbb{R}^{1 \times dim_H}$ for the domain state tracker, followed by a sigmoid activation function σ. D_{t-1} contains `[DOMAIN]`, a special word for representing each domain, and the vector representations of `[DOMAIN]` are used to predict the current domain state D_t as

$$H_t^{Domain} = \left\{\sigma\left(W_{domain} h_{t,i}^{Init}\right) \middle| c_{t,i}^{Init} = \mathtt{[DOMAIN]}\right\} \in \mathbb{R}^{N_D}, \tag{6}$$

$$\mathbb{D}_t = \left\{\mathbb{D}_t^d \middle| h_{t,d}^{Domain} \in H_t^{Domain}, \quad \mathbb{D}_t^d = \begin{cases} ON & \text{if } h_{t,d}^{Domain} \geq 0.5 \\ OFF & \text{else} \end{cases}\right\}, \tag{7}$$

where $d \in [1, N_D]$ is domain index, and N_D is the number of domains; i.e., a constant that indicates the number of `[DOMAIN]` in D_{t-1}.

3.2.2. *Belief Tracker*

Belief tracking is divided into two steps: slot-gate prediction and slot-value generation. The number of slots varies according to the slot design. However, predicting all slot-values every turn is inefficient because only some slots are activated at a time, generally. Therefore, we predict slot-gates that indicates whether each slot is activated in current conversation, and then we generate slot-values for activated slots. We use an FC layer $W_{gate} \in \mathbb{R}^{4 \times dim_H}$ for slot-gate prediction and a GRU (Cho et al., 2014) for slot-value generation. B_{t-1} contains `[SLOT]`, a special word to represent each slot, and the vector representations of `[SLOT]` are used for slot-gate prediction and slot-value generation. The current slot-gate G_t is classified as

$$H_t^{Slot} = \left\{h_{t,i}^{Init} \middle| c_{t,i}^{Init} = \mathtt{[SLOT]}\right\} \in \mathbb{R}^{N_S \times dim_H}, \tag{8}$$

$$G_t = \left\{g_t^s \middle| h_{t,s}^{Slot} \in H_t^{Slot}, \quad g_t^s = \mathtt{argmax}\left(\mathtt{softmax}\left(W_{gate} h_{t,s}^{Slot}\right)\right)\right\}, \tag{9}$$

where $s \in [1, N_S]$ is slot index, and N_S is the number of slots; i.e., a constant that indicates the number of `[SLOT]` in B_{t-1}. The slot-gates have four classes: $g_t^s \in \{\texttt{Update}, \texttt{Copy}, \texttt{Dontcare}, \texttt{Delete}\}$. `Update` indicates that the slot is activated, and the slot-value is generated. `Copy` indicates that the slot is not activated, and the slot-value is copied from the previous turn including `None`. `Dontcare` means that the user do not care about the slot to achieve the goals. `Delete` means that the constraint about the slot to achieve user goals is deleted, and the corresponding value reverts to `None`. If $g_t^s = \texttt{Update}$, a GRU decoder with attention recurrently generates slot-value $V_t^s = \left\{v_{t,1}^s, \cdots, v_{t,|V_t^s|}^s\right\}$ for the s-th slot. First, the GRU decoder generates $z_{t,j}^s$, hidden state of the j-th step on turn t as

$$z_{t,j}^s = \texttt{GRU}\left(\tilde{v}_{t,j-1}^s, z_{t,j-1}^s\right) \in \mathbb{R}^{dim_H}, \quad \left\{ \begin{array}{l} \tilde{v}_{t,j-1}^s = \left\{ \begin{array}{ll} h_{t,s}^{Slot} \in H_t^{Slot} & \text{if } j = 1 \\ E\left(v_{t,j-1}^s\right) & \text{else} \end{array} \right. \\ z_{t,0}^s = o_t^{Init} \end{array} \right. , \tag{10}$$

where E is an embedding layer. Then, attention score $\mathbb{A}_{t,j}^s$ is calculated, and context vector $\mathbb{C}_{t,j}^s$ is obtained depending on $\mathbb{A}_{t,j}^s$ as

$$\mathbb{A}_{t,j}^s = \texttt{softmax}\left(H_t^{Init} z_{t,j}^s\right) \in \mathbb{R}^{\left|C_t^{Init}\right|}, \quad \mathbb{C}_{t,j}^s = \left(H_t^{Init}\right)^T \mathbb{A}_{t,j}^s \in \mathbb{R}^{dim_H}. \tag{11}$$

Finally, $v_{t,j}^s$, the j-th word to compose value of the s-th slot, is generated depending on $z_{t,j}^s$ and $\mathbb{C}_{t,j}^s$ as

$$\tilde{z}_{t,j}^s = W_{vocab}^{belief} \begin{bmatrix} z_{t,j}^s \\ \mathbb{C}_{t,j}^s \end{bmatrix} \in \mathbb{R}^{dim_V}, \quad v_{t,j}^s = \texttt{argmax}\left(\texttt{softmax}\left(\tilde{z}_{t,j}^s\right)\right), \tag{12}$$

where $W_{vocab}^{belief} \in \mathbb{R}^{dim_V \times 2dim_H}$ is an FC layer for projection to vocabulary space, and dim_V is the vocabulary size. Otherwise, V_t^s is determined as follows:

$$V_t^s = \left\{ \begin{array}{ll} V_{t-1}^s & \text{if } g_t^s = \texttt{Copy} \\ \texttt{don't care} & \text{if } g_t^s = \texttt{Dontcare} \\ \texttt{none} & \text{if } g_t^s = \texttt{Delete} \end{array} \right. . \tag{13}$$

The current belief state $\mathbb{B}_t$ consists of the slot-values: $\mathbb{B}_t = \left\{V_t^1, \cdots, V_t^{N_S}\right\}$.

3.2.3. Dialogue Policy

BERT encodes C_t^{Belief} to vector representations H_t^{Belief}, and then $h_{t,1}^{Belief}$ that represents the `[CLS]` word is used to obtain a pooled output o_t^{Belief}, like Equation 5, as follows:

$$H_t^{Belief} = \texttt{BERT}\left(C_t^{Belief}\right) = \left\{h_{t,1}^{Belief}, \cdots, h_{t,\left|C_t^{Belief}\right|}^{Belief}\right\} \in \mathbb{R}^{\left|C_t^{Belief}\right| \times dim_H}, \tag{14}$$

$$o_t^{Belief} = \texttt{tanh}\left(W_{pool} h_{t,1}^{Belief}\right) \in \mathbb{R}^{dim_H}. \tag{15}$$

System actions $A_t = \{a_{t,1}, \cdots, a_{t,|A_t|}\}$ are recurrently generated by the GRU based dialogue policy with attention, like the process of belief tracker (Subsection 3.2.2), as

$$z_{t,k} = \texttt{GRU}\left(E\left(a_{t,k-1}\right), z_{t,k-1}\right) \in \mathbb{R}^{dim_H}, \quad \left\{ \begin{array}{l} a_{t,0} = \texttt{[CLS]} \\ z_{t,0} = o_t^{Belief} \end{array} \right. , \tag{16}$$

$$\mathbb{A}_{t,k} = \texttt{softmax}\left(H_t^{Belief} z_{t,k}\right) \in \mathbb{R}^{\left|C_t^{Belief}\right|}, \quad \mathbb{C}_{t,k} = \left(H_t^{Belief}\right)^T \mathbb{A}_{t,k} \in \mathbb{R}^{dim_H}, \tag{17}$$

$$\tilde{z}_{t,k} = W_{vocab}^{act} \begin{bmatrix} z_{t,k} \\ \mathbb{C}_{t,k} \end{bmatrix} \in \mathbb{R}^{dim_V}, \quad a_{t,k} = \texttt{argmax}\left(\texttt{softmax}\left(\tilde{z}_{t,k}\right)\right), \tag{18}$$

where $W_{vocab}^{act} \in \mathbb{R}^{dim_V \times 2dim_H}$ is an FC layer for projection to vocabulary space.

3.2.4. Response Generator

BERT encodes C_t^{Act} to vector representations H_t^{Act}, and then $h_{t,1}^{Act}$ that represents the `[CLS]` word is used to obtain a pooled output o_t^{Act}, like Equation 5, as follows:

$$H_t^{Act} = \mathtt{BERT}\left(C_t^{Act}\right) = \left\{h_{t,1}^{Act}, \cdots, h_{t,\left|C_t^{Act}\right|}^{Act}\right\} \in \mathbb{R}^{\left|C_t^{Act}\right| \times dim_H}, \tag{19}$$

$$o_t^{Act} = \mathtt{tanh}\left(W_{pool} h_{t,1}^{Act}\right) \in \mathbb{R}^{dim_H}. \tag{20}$$

The response generator that uses GRU with attention recurrently generates system response $X_t = \{x_{t,1}, \cdots, x_{t,|X_t|}\}$ by greedy decoding, like the process of belief tracker (Subsection 3.2.2), as

$$z_{t,l} = \mathtt{GRU}\left(E\left(x_{t,l-1}\right), z_{t,l-1}\right) \in \mathbb{R}^{dim_H}, \quad \left\{ \begin{array}{l} x_{t,0} = \mathtt{[CLS]} \\ z_{t,0} = o_t^{Act} \end{array} \right., \tag{21}$$

$$\mathbb{A}_{t,l} = \mathtt{softmax}\left(H_t^{Act} z_{t,l}\right) \in \mathbb{R}^{\left|C_t^{Act}\right|}, \quad \mathbb{C}_{t,l} = \left(H_t^{Act}\right)^T \mathbb{A}_{t,l} \in \mathbb{R}^{dim_H}, \tag{22}$$

$$\tilde{z}_{t,l} = W_{vocab}^{resp} \begin{bmatrix} z_{t,l} \\ \mathbb{C}_{t,l} \end{bmatrix} \in \mathbb{R}^{dim_V}, \quad x_{t,l} = \mathtt{argmax}\left(\mathtt{softmax}\left(\tilde{z}_{t,l}\right)\right), \tag{23}$$

where $W_{vocab}^{resp} \in \mathbb{R}^{dim_V \times 2dim_H}$ is an FC layer for projection to vocabulary space.

3.3. Pre-training with SL

DORA is optimized in two steps: pre-training with SL and policy optimization with RL. During the SL step, the entire system is trained using cross-entropy loss with gold-standard annotations. The parameters are updated for each turn by backpropagation. Equation 7 is used to calculate a domain state loss $\mathcal{L}_t^{Domain}$ by using binary cross-entropy with domain state labels $Y_t^{Domain} = \left\{y_{t,1}^{Domain}, \cdots, y_{t,N_D}^{Domain}\right\}$ as

$$\mathcal{L}_t^{Domain} = -\frac{1}{N_D} \sum_{d=1}^{N_D} y_{t,d}^{Domain} log P\left(\mathbb{D}_t^d\right) + \left(1 - y_{t,d}^{Domain}\right) log\left(1 - P\left(\mathbb{D}_t^d\right)\right), \tag{24}$$

where $y_{t,d}^{Domain}$ is a binary value indicating whether the d-th domain is activated on turn t.

Equation 9 is used to calculate a slot-gate loss $\mathcal{L}_t^{Gate}$ by using cross-entropy with slot-gate labels $Y_t^{Gate} = \left\{y_{t,1}^{Gate}, \cdots, y_{t,N_S}^{Gate}\right\}$ as

$$\mathcal{L}_t^{Gate} = -\frac{1}{N_S} \sum_{s=1}^{N_S} \left(y_{t,s}^{Gate}\right)^T log P\left(g_t^s\right), \tag{25}$$

where $y_{t,s}^{Gate} \in \mathbb{R}^4$ is an one-hot vector that indicates the s-th slot-gate on turn t.

Equation 12 is used to calculate a slot-value loss $\mathcal{L}_t^{Value}$ by using cross-entropy with slot-value labels Y_t^{Value}, but $\mathcal{L}_t^{Value}$ is calculated only for `Update` slots $\mathbb{U}_t$, unlike the slot-gate loss, as

$$\mathbb{U}_t = \left\{s \middle| s \in [1, N_S], \quad g_t^s = \mathtt{Update}\right\}, \tag{26}$$

$$Y_t^{Value} = \left\{y_{t,u}^{Value} \middle| u \in \mathbb{U}_t, \quad y_{t,u}^{Value} = \left\{y_{t,u,1}^{Value}, \cdots, y_{t,u,\left|V_t^u\right|}^{Value}\right\}\right\}, \tag{27}$$

$$\mathcal{L}_t^{Value} = -\frac{1}{\left|\mathbb{U}_t\right|} \sum_{u \in \mathbb{U}_t} \frac{1}{\left|V_t^u\right|} \sum_{j=1}^{\left|V_t^u\right|} \left(y_{t,u,j}^{Value}\right)^T log P\left(v_{t,j}^u\right), \tag{28}$$

where $y_{t,u,j}^{Value} \in \mathbb{R}^{dim_V}$ is an one-hot vector that indicates the j-th word of the slot-value for the u-th `Update` slot on turn t.

The dialogue policy is trained as a conditional language model during the SL step. Equation 18 is used to calculate a system action loss $\mathcal{L}_t^{Act}$ by using cross-entropy with system action labels $Y_t^{Act} = \left\{y_{t,1}^{Act}, \cdots, y_{t,|A_t|}^{Act}\right\}$ as

$$\mathcal{L}_t^{Act} = -\frac{1}{|A_t|}\sum_{k=1}^{|A_t|}\left(y_{t,k}^{Act}\right)^T logP\left(a_{t,k}\right), \tag{29}$$

where $y_{t,k}^{Act} \in \mathbb{R}^{dim_V}$ is an one-hot vector that indicates the k-th word of the system action on turn t.

Equation 23 is used to calculate a system response loss $\mathcal{L}_t^{Resp}$ by using cross-entropy with system response labels $Y_t^{Resp} = \left\{y_{t,1}^{Resp}, \cdots, y_{t,|X_t|}^{Resp}\right\}$ as

$$\mathcal{L}_t^{Resp} = -\frac{1}{|X_t|}\sum_{l=1}^{|X_t|}\left(y_{t,l}^{Resp}\right)^T logP\left(x_{t,l}\right), \tag{30}$$

where $y_{t,l}^{Resp} \in \mathbb{R}^{dim_V}$ is an one-hot vector that indicates the l-th word of the system response on turn t.

The final joint loss for pre-training with SL is sum of the above losses:

$$\mathcal{L}_t^{SL} = \mathcal{L}_t^{Domain} + \mathcal{L}_t^{Gate} + \mathcal{L}_t^{Value} + \mathcal{L}_t^{Act} + \mathcal{L}_t^{Resp}. \tag{31}$$

During the SL step, DORA is trained by minimizing $\mathcal{L}_t^{SL}$ using Adam optimizer (Kingma and Ba, 2015) for each turn.

3.4. Policy Optimization with RL

In this subsection, we describe how to optimize the pre-trained dialogue policy and to shape rewards for RL.

3.4.1. Recurrent Action Policy Optimization

During the RL step, only the dialogue policy is optimized, and other modules are fixed. The parameters are updated for each episode during the RL step because the task success is determined at the end of the conversation, whereas the parameters are updated for each turn during the SL step. We apply REINFORCE algorithm (Williams, 1992), a basic policy gradient method, to optimize the policy. We treat the dialogue policy as a word-level policy sampling actions from a probability distribution, rather than a greedy decoder, by rewriting Equation 18 as

$$P_\theta\left(a_{t,k}\middle|C_t^{Belief},\ a_{t,<k}\right) = \texttt{softmax}\left(\tilde{z}_{t,k}\right). \tag{32}$$

The policy samples a word from the probability distribution P_θ, instead of using `argmax` function to select a word, to apply RL. Even though the policy is word-level, the action space has low variance because the system actions consist of a few specific words, including some special words. Depending on C_t^{Belief}, the dialogue policy recurrently samples actions until `[EOS]`, a special word to terminate the generation, is detected. Several words comprise a system action, and several system actions comprise A_t on turn t. We optimize the policy to maximize rewards on collected training datasets, in other words, on the offline environment. The policy gradient is:

$$\nabla_\theta J(\theta) = \mathbb{E}_\theta\left[\sum_{t=1}^{T}\sum_{k=1}^{|A_t|} R_{t,k}\nabla_\theta logP_\theta\left(a_{t,k}\middle|C_t^{Belief},\ a_{t,<k}\right)\right], \tag{33}$$

where T is the number of conversation turns in the dialogue episode, and $R_{t,k}$ is a return of k-th action on turn t. We demonstrate the details of $R_{t,k}$ in Subsection 3.4.2. During the RL step, the policy is optimized by Equation 33 using stochastic gradient descent (SGD) optimization algorithm.

Algorithm 1: Calculation of the system action rate reward

```
Result: System action rate rewards
initialize the set of accumulated system action rates: $\Phi_A = \{\}$ ;
while RL step is not done do
    initialize training dataset ;
    for each episode in training dataset do
        for each turn $t$ do
            calculate system action rate $\mathcal{A}_t \in [0, 1]$ depending on $A_t$ ;
            append $\mathcal{A}_t$ to $\Phi_A$ ;
            standardize $\mathcal{A}_t$: $\mathcal{A}_t \leftarrow \frac{\mathcal{A}_t - \mathtt{mean}(\Phi_A)}{\mathtt{max}(\mathtt{std}(\Phi_A), \epsilon)}$, where $\epsilon$ is a hyperparmeter ;
            assign $\mathcal{A}_t$ to system action rate reward of the last word of $A_t$: $r^{action}_{t,|A_t|} \leftarrow \mathcal{A}_t$ ;
            assign zeros to system action rate reward of the other words of $A_t$: $r^{action}_{t,1}, \cdots, r^{action}_{t,|A_t|-1} \leftarrow 0$ ;
        end
    end
end
```

3.4.2. Reward Shaping

To apply RL, we use the success rate of dialogues as a reward. The success rate is a rate of user goals that were successfully achieved by the dialogue system, and the success rate generally indicates the main evaluation metric of task-oriented dialogue systems. By using RL, we can directly maximize the success rate, which is non-differentiable and therefore cannot be directly set as an objective during the SL step. However, the success rate is significantly sparse because an episode should finish to calculate it, and as a result, the RL step is inefficient and unstable.

In this work, we use the system action rate as an auxiliary reward to exploit use of explicit system actions and simultaneously to solve inefficiency and instability of the RL step. The system action rate means the precision of generated system actions, not word-by-word, but action-by-action, and it is obtained for each turn. Furthermore, some system actions are more important than others for achieving user goals, so for calculation of the system action rate, the system actions can be weighted according to their importance. The total reward of the k-th word on turn t is calculated as the weighted sum of the success rate reward $r^{success}_{t,k}$ and the system action rate reward $r^{action}_{t,k}$ as

$$r^{total}_{t,k} = r^{success}_{t,k} + \beta r^{action}_{t,k}, \tag{34}$$

where $\beta \in [0, 1]$ is a hyperparameter.

Rewards should be discounted over time steps to prevent bias toward long actions. Previous methods that use a latent policy with RL discount the success rate reward over turns; i.e., the episode-level rewards calculated by the success rate assign a low weight to the first turn's actions and a high weight to the the last turn's actions. However, this method is not clear to optimize a task-oriented dialogue system that achieves user goals across multiple turns in multi-domain dialogues. In DORA, generated system actions do not affect the next turn's actions in offline environments because DORA does not use system responses as inputs. Thus, we can convert the success rate reward from episode-level to turn-level by applying the same success rate reward every turn and by discounting the total reward over words for each turn as

$$r^{success}_{1,|A_1|} = \cdots = r^{success}_{T,|A_T|}, \quad R_{t,k} = \sum_{i=0}^{|A_t|-k} \gamma^i r^{total}_{t,k+i}, \tag{35}$$

where $\gamma \in [0, 1]$ is a discount factor. By using turn-level rewards, we can weight the actions for every turn equally. The system action rate is calculated for each turn. Algorithm 1 demonstrates how to calculate $r^{action}_{t,k}$. Unlike the system action rate, the success rate is calculated for each episode. Algorithm 2 demonstrates how to calculate $r^{success}_{t,k}$.

3.5. System Action Control

Explicit system actions are obviously interpretable, so they can be controlled manually. In multi-domain task-oriented dialogues, various system actions can match a given context, and the importance of each system action can

Algorithm 2: Calculation of the success rate reward

```
Result: Success rate rewards
initialize the set of accumulated success action rates: $\Phi_S = \{\}$ ;
while RL step is not done do
    initialize training dataset ;
    for each episode in training dataset do
        calculate success rate $\mathcal{S} \in [0, 1]$ depending on $X_1, \cdots, X_T$ ;
        append $\mathcal{S}$ to $\Phi_S$ ;
        standardize $\mathcal{S}$: $\mathcal{S} \leftarrow \frac{\mathcal{S} - \mathtt{mean}(\Phi_S)}{\mathtt{max}(\mathtt{std}(\Phi_S), \epsilon)}$ ;
        assign $\mathcal{S}$ to success rate reward of the last word of $A_t$ for each turn: $r^{success}_{1,|A_1|}, \cdots, r^{success}_{T,|A_T|} \leftarrow \mathcal{S}$ ;
        assign zeros to success rate reward of the other words of $A_t$ for each turn:
          $r^{success}_{1,1}, \cdots, r^{success}_{1,|A_1|-1}, \cdots, r^{success}_{T,1}, \cdots, r^{success}_{T,|A_T|-1} \leftarrow 0$ ;
    end
end
```

vary depending on tasks. During inference, A_t is generated by the optimized dialogue policy, and then we can modify A_t depending on some rules manually designed for specific purposes. We define simple rules by using heuristics (Subsection 4.2.2) to improve the task success on MultiWOZ datasets.

4. Experiments

In this section, we demonstrate our experiments including datasets, evaluation metrics, detail processes, results, and ablation study.

4.1. Experimental Setups

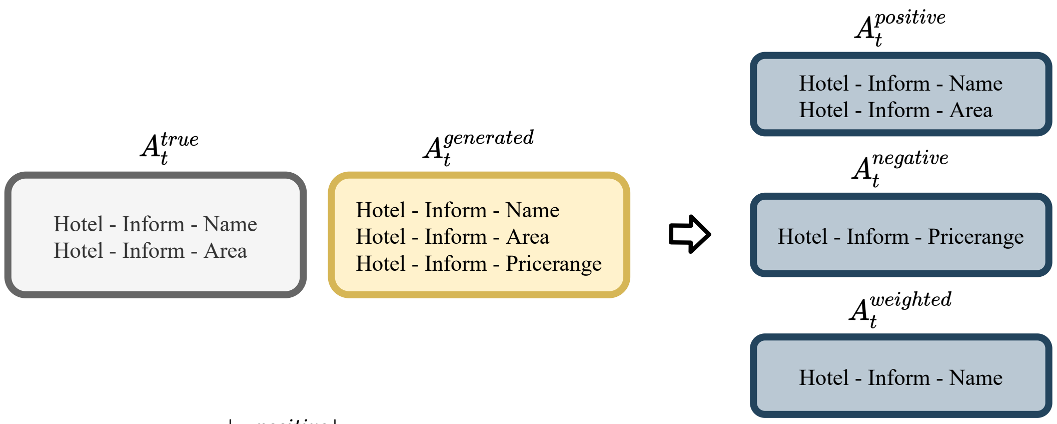
We used MultiWOZ 2.0 and MultiWOZ 2.1 [1] for datasets of experiments. MultiWOZ 2.0 is a large dataset for task-oriented dialogue systems, and MultiWOZ 2.1 is an improved version in which some annotation errors have been corrected. MultiWOZ consists of roughly 10400 conversations that had been collected using Wizard-Of-Oz setup. The conversations consider seven domains about travel in Cambridge, e.g., restaurants and hotels. The dialogues in MultiWOZ are divided into about 8400 for training, 1000 for validation, and 1000 for test.

The evaluation metrics on MultiWOZ are inform rate, success rate, and BLEU score. The inform rate counts the number of times the system provides appropriate entries that satisfy the constraints of user goals. The success rate counts the number of times the system successfully provides information requested by users, in addition to the conditions for the inform rate. BLEU score measures the similarity between generated responses and response labels in the dataset. However, the response labels are not the optimal ones to achieve user goals in task-oriented dialogues. Rather, they were simply written by humans. Thus, BLEU score is not convincing method to evaluate task-oriented dialogue systems, so we aim to increase the inform rate and especially the success rate, rather than BLEU score.

4.2. Experimental Details

In experiments, we use BERT-base-uncased [2] as the shared encoder, and one-layer GRUs for the decoders. MultiWOZ provides three types of setting to evaluate task-oriented dialogue systems: (1) belief tracking evaluation with gold-standard system responses, (2) policy optimization evaluation with gold-standard belief state, and (3) end-to-end evaluation without any gold-standard labels for inference. We only evaluate DORA on the end-to-end setting, which is closest to real world problems. We optimize DORA over two steps: pre-training during the SL step and fine-tuning during the RL step. The SL step optimizes the entire system, whereas the RL step only optimizes the dialogue policy

[1] `https://github.com/budzianowski/multiwoz`.
[2] The pre-trained model is available at `https://github.com/huggingface/transformers`.



$$\mathcal{A}_t^{base} = \frac{\left|A_t^{positive}\right|}{\left|A_t^{generated}\right|} = \frac{2}{3}$$

$$\mathcal{A}_t^{negative} = \frac{\left|A_t^{positive}\right| - \left|A_t^{negative}\right|}{\left|A_t^{generated}\right|} = \frac{2-1}{3} = \frac{1}{3}$$

$$\mathcal{A}_t^{weigthed} = \frac{\left|A_t^{positive}\right| + \left|A_t^{weighted}\right|}{\left|A_t^{generated}\right|} = \frac{2+1}{3} = 1$$

$$\mathcal{A}_t^{combined} = \frac{\left|A_t^{positive}\right| + \left|A_t^{weighted}\right| - \left|A_t^{negative}\right|}{\left|A_t^{generated}\right|} = \frac{2+1-1}{3} = \frac{2}{3}$$

Figure 5: Process of system action rate calculation for each method on MultiWOZ.

with other modules fixed. We apply an early-stopping method that stops the training when validation score does not improve over five epochs. We conducted the experiments on a TitanRTX GPU, and the average training time was about two days for each step.

4.2.1. System Action Rate

We attempted four approaches to calculating system action rate $\mathcal{A}_t$ on turn t. First, the base method is to calculate precision by considering generated system actions $A_t^{generated}$ and system action labels A_t^{true} as follows:

$$A_t^{positive} = \left\{a \middle| a \in A_t^{generated} \cap A_t^{true}\right\}, \quad \mathcal{A}_t^{base} = \frac{\left|A_t^{positive}\right|}{\left|A_t^{generated}\right|}. \tag{36}$$

Second, the negative method gives a penalty for system actions that do not exist in A_t^{true} to prevent the dialogue policy from generating too many system actions as follows:

$$A_t^{negative} = \left\{a \middle| a \in A_t^{generated} - A_t^{true}\right\}, \quad \mathcal{A}_t^{negative} = \frac{\left|A_t^{positive}\right| - \left|A_t^{negative}\right|}{\left|A_t^{generated}\right|}. \tag{37}$$

Third, the weighted method increases the weight given to system actions that are important to achieve task success as:

$$A_t^{weighted} = \left\{a \middle| a \in A_t^{positive} \cap A^{important}\right\}, \quad \mathcal{A}_t^{weighted} = \frac{\left|A_t^{positive}\right| + \left|A_t^{weighted}\right|}{\left|A_t^{generated}\right|}, \tag{38}$$

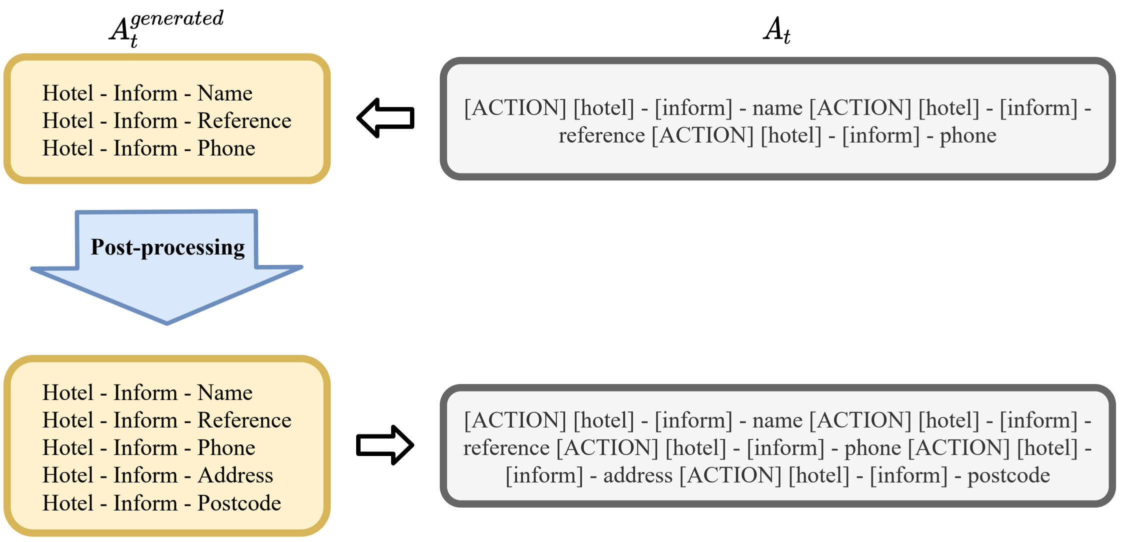

$A_t^{generated}$
A_t
Hotel - Inform - Name
Hotel - Inform - Reference
Hotel - Inform - Phone
[ACTION] [hotel] - [inform] - name [ACTION] [hotel] - [inform] - reference [ACTION] [hotel] - [inform] - phone
Post-processing
Hotel - Inform - Name
Hotel - Inform - Reference
Hotel - Inform - Phone
Hotel - Inform - Address
Hotel - Inform - Postcode
[ACTION] [hotel] - [inform] - name [ACTION] [hotel] - [inform] - reference [ACTION] [hotel] - [inform] - phone [ACTION] [hotel] - [inform] - address [ACTION] [hotel] - [inform] - postcode

Figure 6: Process of system action control by post-processing on MultiWOZ.

where the set of important system actions $A^{important}$ is predefined by heuristic (Tabel C.7). Finally, the combined negative and weighted method combines the two methods as follows:

$$A_t^{combined} = \frac{\left|A_t^{positive}\right| + \left|A_t^{weighted}\right| - \left|A_t^{negative}\right|}{\left|A_t^{generated}\right|}. \tag{39}$$

Figure 5 shows examples of system action rate calculation process. Each system action is a triple of domain, action, and slot. In Figure 5, three system actions are generated by the dialogue policy: `Hotel-Inform-Name`, `Hotel-Inform-Area`, and `Hotel-Inform-Pricerange`.

4.2.2. System Action Control

We control generated system actions by post-processing depending on some rules. By this control of system actions, we aim to improve the success rate, which is the main evaluation metric on MultiWOZ and is matched with the objective of task-oriented dialogue systems in the real world. Several system actions are more important than others for the task success on MultiWOZ; e.g., to provide a phone number requested by the user. In our experiments, we define simple post-processing rules that adjust generated system actions:

1. If the dialogue policy generates system actions to provide some information about a hotel,
2. Adjust the actions to necessarily include phone number, address, and postcode of the hotel.
3. Repeat 1-2 for restaurants and attractions.

Figure 6 shows how A_t is parsed to $A_t^{generated}$ and how the generated system actions are controlled by post-processing. We apply the post-processing only during inference, not training. In this paper, we only attempt to simply control system actions for the task success.

4.3. Experimental Results

Table 1 compares the results of four methods for calculating the system action rate as mentioned in Subsection 4.2.1 and the results without the system action rate as reward for the RL step. The results further contains the effects of system action control by post-processing. In our experiments, use of the system action rate as a reward for RL quite improved the success rate except of the negative method, but it had little effect on the success rate when we applied system action control in addition to the use of the system action rate. These results suggest that use of the system action rate and control of system actions had similar effects in the experiments by increasing the weights of certain

Table 1: Comparison of various configurations of DORA on test set of MultiWOZ 2.0 and MultiWOZ 2.1.

		MultiWOZ 2.0			MultiWOZ 2.1		
		Inform↑	Success↑	BLEU↑	Inform↑	Success↑	BLEU↑
w/o post-processing	w/o action rate	94.9	90.0	12.61	94.8	90.4	13.29
	base	94.6	**92.0**	12.70	94.4	91.1	12.58
	negative	94.1	88.9	**13.94**	94.3	90.4	**13.40**
	weighted	94.9	91.4	12.23	**95.2**	92.3	12.31
	combined	**95.3**	91.3	13.31	95.0	91.4	12.80
w/ post-processing	w/o action rate	94.9	91.9	12.58	94.8	91.6	12.95
	base	94.7	**92.0**	12.72	94.4	91.0	12.46
	negative	94.0	90.6	12.78	94.3	90.8	13.10
	weighted	94.9	**92.0**	12.17	**95.2**	**92.7**	12.21
	combined	95.2	91.6	13.26	95.0	91.9	12.20

Table 2: Results of end-to-end evaluation on test set of MultiWOZ 2.0 and MultiWOZ 2.1. SUMBT+LaRL uses a previous system response instead of the entire history.

Model	History	MultiWOZ 2.0			MultiWOZ 2.1		
		Inform↑	Success↑	BLEU↑	Inform↑	Success↑	BLEU↑
DAMD (Zhang et al., 2020b)	✓	76.30	60.40	16.60	-	-	-
LABES-S2S (Zhang et al., 2020a)	✓	-	-	-	78.07	67.06	**18.13**
SimpleTOD (Hosseini-Asl et al., 2020)	✓	84.40	70.10	15.01	85.00	70.50	15.23
SOLOIST (Peng et al., 2020)	✓	85.50	72.90	16.54	-	-	-
MinTL-BART (Lin et al., 2020)	✓	84.88	74.91	17.89	-	-	-
LAVA (Lubis et al., 2020)	✓	91.80	81.80	12.03	-	-	-
UBAR (Yang et al., 2020)	✓	**95.40**	80.70	17.00	**95.70**	81.80	16.50
SUMBT+LaRL (Lee et al., 2020)	✓	92.20	85.40	**17.90**	-	-	-
DORA		94.60	**92.00**	12.70	94.40	**91.10**	12.58

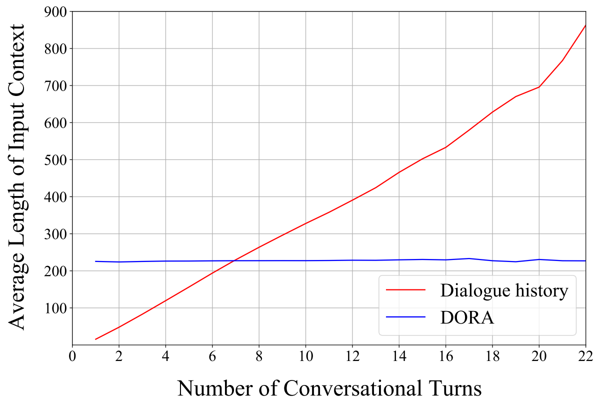
system actions to enable the task success. Also, we believe that the negative method gave wrong penalties for system actions that did not exist in true system actions of datasets even though they were appropriate system actions.

In our experiments, DORA achieved higher task success than previous methods on end-to-end evaluation setting of MultiWOZ 2.0 and MultiWOZ 2.1 without any previous utterances (Table 2). DORA also achieved higher task success during the SL step than previous methods using latent actions for RL even though several of the previous methods used gold-standard belief state for inference (Table 3). These results demonstrate that pre-training with latent policy is unstable, whereas DORA becomes well optimized during the SL step.

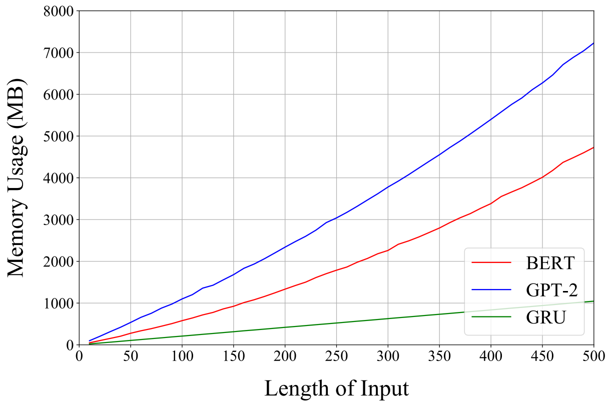
We conducted additional experiments to confirm the efficiency of our input context. As the number of turns increased, the length of input context used in DORA was almost constant, whereas the dialogue history lengthened (Figure 7a). We compared memory usage depending on the input length on three language models: BERT, GPT-2, and GRU (Figure 7b). The memory usage of the three language models increased with different speeds as the input length increased. The language models have 12 layers with hidden size of 768, but GPT-2 uses a larger vocabulary

Table 3: SL step results of previous models with latent actions and DORA on MultiWOZ 2.0.

Model	Belief State	Inform↑	Success↑	BLEU↑
LaRL	oracle	67.98	57.36	19.10
LAVA	oracle	71.97	57.96	18.00
SUMBT+LaRL	generated	72.10	66.20	**19.36**
DORA	generated	**85.60**	**74.60**	15.35



(a) Lengths of two system inputs on MultiWOZ.



(b) Memory usage depending on input length.

Figure 7: Input length comparison and memory usage acceleration depending on input length.

than the other two. The memory usage indicates the allocated amount of memory on GPU, except to retain the model, after an input with batch size of 8 is fed into the model. Even with same increment of input length, the memory usage increased strongly as model was large.

4.3.1. Ablation Study

To verify the effectiveness of our approach, we conducted an ablation study (Table 4). We sequentially removed system action rate, RL step, and the domain state. Furthermore, we attempted using the dialogue history instead of our efficient context, as in previous methods. Removal of the system action rate from rewards for the RL step slightly decreased the success rate. Subsequent removal of the RL step and then optimization using only the SL greatly degraded the success rate. Additionally, removal of the domain state from the input context further decreased the success rate. Finally, when dialogue history was used as input, the success rate was lower than when we used the efficient context. The results show that the components of our approach help to improve the success rate.

Table 4: Ablation study on MultiWOZ 2.0.

	Inform↑	Success↑	BLEU↑
DORA	94.6	92.0	12.70
- system action rate	94.9	90.0	12.61
- RL	85.6	74.6	15.35
- domain state	85.2	68.5	14.14
using history	79.3	63.5	14.48

5. Discussion

The SL step is the cornerstone of the RL step, and explicit system action policy makes the SL step clearer (Table 3). The system action policy further enables use of system actions to shape rewards during the RL step and to control the system actions generated by the optimized policy. Use of the system action rate as reward generally improved success rate except when the negative method was applied (Table 1). Calculation of the system action rate used labels that had been annotated on datasets as A_t^{true}. On turn t, A_t^{true} is a set of appropriate system actions, but it is not the optimal one. This problem seems to be similar to the bias that occurs when SL is used to train task-oriented dialogue systems. The negative method may have given wrong penalties for system actions that were appropriate on turn t but not included in the labels.

In our experiments, system action control improved the success rate more when rewards were calculated without the system action rate, than with it. The purpose of system action control was to improve the task success on MultiWOZ (Subsection 4.2.2). Use of the system action rate as a reward for RL has a semantically similar purpose: to give more rewards when the dialogue policy generates appropriate system actions for the task success. Therefore, system action control improved the success rate more when reward was shaped without the system action rate, and use of the system action rate improved the success rate more when system action was not controlled (Table 1). The results show that use of the system action rate and control of system actions performed semantically similar roles for task success; i.e., the system action control fulfilled our purpose even though it depended on a simple heuristic.

Furthermore, control of the optimized dialogue policy by post-processing enables the system to operate as a hybrid system. Recently, deep learning methods have begun to be more successful than conventional rule-based methods. Models that use deep learning operate well in general and cover wide domains. However, the models are not perfect and have not completely replaced rule-based models in the real world. The conventional models are more accurate than the deep learning models in specific areas. Also, deep learning models require large cost for training on new areas. Therefore, research to develop hybrid approaches that have advantages of both approaches may have significant benefits, and we believe that our approach has value as a way to solve practical problems, in addition to its increase in the success rate.

Use of pre-trained language models has been a trend in NLP field, including task-oriented dialogues. The strategy has improved the success rate, but large models require a large amount of memory (Figure 7b) and thereby increase the training cost. Use of the dialogue history makes memory usage unstable and thereby impedes memory management on GPUs, in addition to increasing the memory usage, especially on large models. Furthermore, using the dialogue history as input induces a dependency on generated system responses during inference, as an NLU module encodes the generated response on next turn. This observation seems counterintuitive because the responses were generated by depending on information already known to the system. Also, the use of dialogue history as input can cause propagation of errors to future turns. For these reasons, approaches to efficient input context should be sought for use in task-oriented dialogue systems.

We construct the input context by combining the previous domain state, belief state, and the current user utterance. The previous states contain abstract information accumulated during previous turns, and the current user utterance indicates new information from the user. By using the context, the system can efficiently track the flow of conversation by comparing them and then updating the states. Furthermore, the context stabilizes the system by removing dependency on generated system responses.

6. Conclusion

In this paper, we have proposed DORA, which is an efficient task-oriented dialogue system that uses effective optimization methods. We have demonstrated that clear pre-training with SL is important for effective fine-tuning with RL, and the explicit system action policy clarifies the SL step. Our experiments show that the task success is increased by using the system action rate to shape rewards during the RL step, and by post-processing to control system actions. We have further presented a fresh perspective of task-oriented dialogue systems as a hybrid approach to address practical problems as well as academic research. Also, we have proposed an efficient method to construct input context that can replace the entire dialogue history.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-01906, Artificial Intelligence Graduate School

Program(POSTECH)); and the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2021-2020-0-01789) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation)

Appendix A. Hyperparameters

We report the hyperparameters used in the experiments for reproducibility. Table A.5 lists the hyperparameters for the SL and RL steps.

Table A.5: Hyperparameters used for the experiments in this paper.

	SL step	
Hidden size	768	The size of hidden layers.
Embedding size	768	The size of embedding layer.
Vocabulary size	30522	The size of vocabulary.
Max context length	512	The maximum length of input context.
Dropout	0.2	The rate of dropout.
Early stopping count	5	The maximum count to early stop the training.
Max epochs	40	The maximum number of epochs.
Min epochs	20	The minimum number of epochs ignoring the early stopping.
Optimizer	Adam	The type of optimizer for the SL step.
Batch size	8	The size of mini batch for the SL step.
Learning rate	3e-5	The learing rate during the SL step.
Gradient clipping	10	The maximum norm of gradient during the SL step.
	RL step	
β (MultiWOZ 2.0)	1e-3	The weight of action rate to calculate the total reward.
β (MultiWOZ 2.1)	1e-2	
γ	0.99	The discount factor during the RL step.
Optimizer	SGD	The type of optimizer for the RL step.
Batch size	8	The size of mini batch for the RL step.
Learning rate	1e-2	The learing rate during the RL step.
Gradient clipping	1	The maximum norm of gradient during the RL step.
ϵ	1e-4	The epsilon for standardization of rewards during the RL step.

Appendix B. Input Context Flow

Figure B.8 shows an example of flow of the input context: initial context C_t^{Init}, belief context C_t^{Belief}, and action context C_t^{Act}. On the initial turn, each domain in the previous domain state $\mathbb{D}_0$ is OFF, and each slot-value in the previous belief state $\mathbb{B}_0$ is `None`.

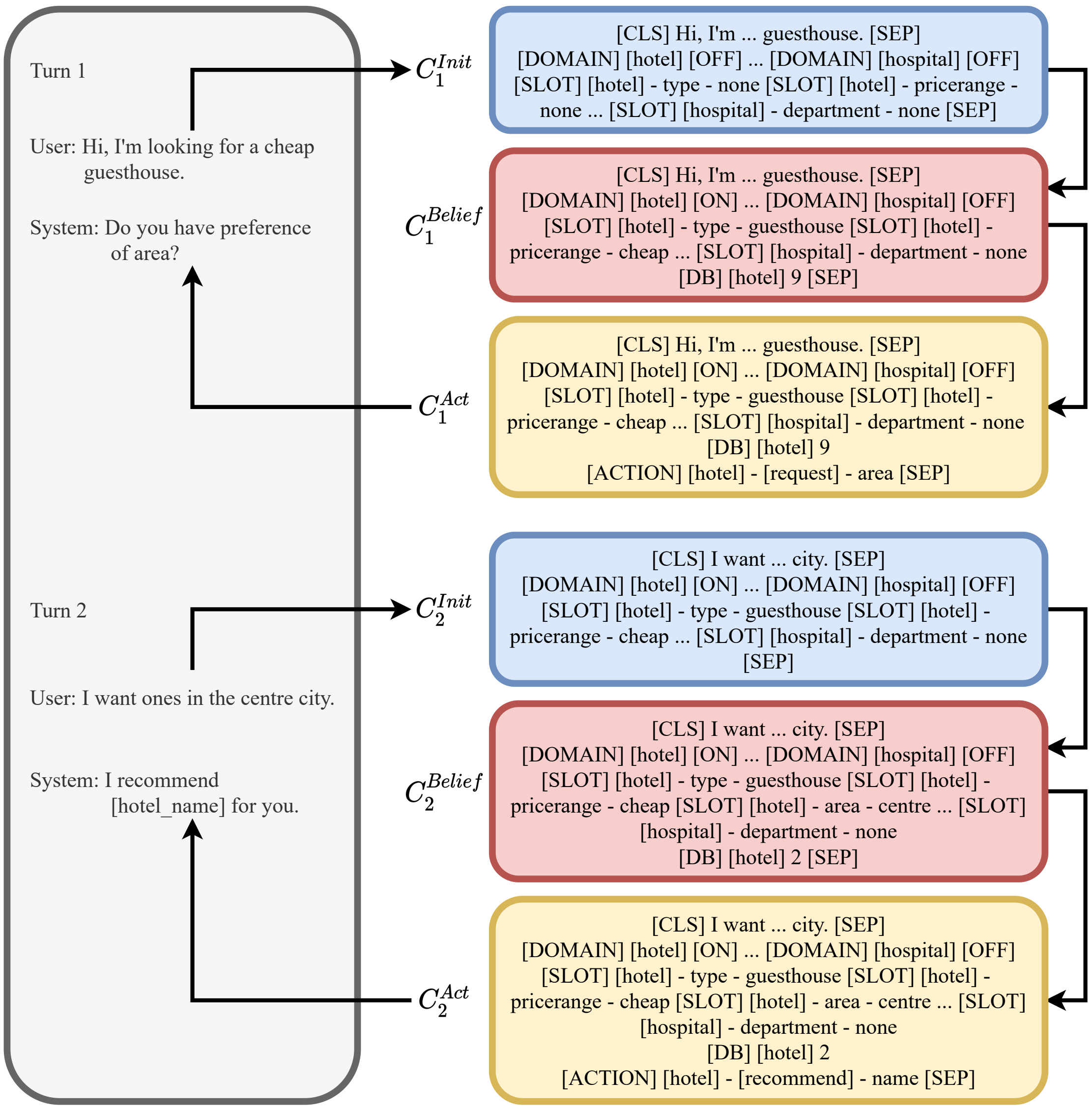

Turn 1
User: Hi, I'm looking for a cheap guesthouse.
System: Do you have preference of area?
C_1^{Init}
[CLS] Hi, I'm ... guesthouse. [SEP] [DOMAIN] [hotel] [OFF] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - none [SLOT] [hotel] - pricerange - none ... [SLOT] [hospital] - department - none [SEP]
C_1^{Belief}
[CLS] Hi, I'm ... guesthouse. [SEP] [DOMAIN] [hotel] [ON] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - guesthouse [SLOT] [hotel] - pricerange - cheap ... [SLOT] [hospital] - department - none [DB] [hotel] 9 [SEP]
C_1^{Act}
[CLS] Hi, I'm ... guesthouse. [SEP] [DOMAIN] [hotel] [ON] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - guesthouse [SLOT] [hotel] - pricerange - cheap ... [SLOT] [hospital] - department - none [DB] [hotel] 9 [ACTION] [hotel] - [request] - area [SEP]
Turn 2
User: I want ones in the centre city.
System: I recommend [hotel_name] for you.
C_2^{Init}
[CLS] I want ... city. [SEP] [DOMAIN] [hotel] [ON] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - guesthouse [SLOT] [hotel] - pricerange - cheap ... [SLOT] [hospital] - department - none [SEP]
C_2^{Belief}
[CLS] I want ... city. [SEP] [DOMAIN] [hotel] [ON] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - guesthouse [SLOT] [hotel] - pricerange - cheap [SLOT] [hotel] - area - centre ... [SLOT] [hospital] - department - none [DB] [hotel] 2 [SEP]
C_2^{Act}
[CLS] I want ... city. [SEP] [DOMAIN] [hotel] [ON] ... [DOMAIN] [hospital] [OFF] [SLOT] [hotel] - type - guesthouse [SLOT] [hotel] - pricerange - cheap [SLOT] [hotel] - area - centre ... [SLOT] [hospital] - department - none [DB] [hotel] 2 [ACTION] [hotel] - [recommend] - name [SEP]

Figure B.8: Flow of the context indicating the input sequence of shared BERT encoder on MultiWOZ. The words enclosed in brackets are special words.

Appendix C. Definition of System Action Set

We report the set of system actions defined on MultiWOZ. Tabel C.6 lists the system actions annotated on MultiWOZ for each domain. We define the set of important system actions $A^{important}$ to calculate the weighted system action rate. Table C.7 lists the system actions in $A^{important}$ regarded as important actions in terms of the task success on MultiWOZ.

Table C.6: Set of system actions defined on MultiWOZ.

Domain	Action	Slot					
Hotel	Inform	Name Address	Type Postcode	Area Phone	Pricerange Stars	Internet Choice	Parking Reference

Hotel	Request	Name Start	Type	Area	Pricerange	Internet	Parking
	Recommend	Name Address	Type Postcode	Area Phone	Pricerange Stars	Internet Choice	Parking
	Select	Name Address	Type Phone	Area Stars	Pricerange Choice	Internet	Parking
	NoOffer	Name Stars	Type Choice	Area	Pricerange	Internet	Parking
Restaurant	Inform	Name Phone	Food Choice	Area Reference	Pricerange	Address	Postcode
	Request	Name	Food	Area	Pricerange		
	Recommend	Name Phone	Food Choice	Area	Pricerange	Address	Postcode
	Select	Name	Food	Area	Pricerange	Address	Choice
	NoOffer	Name	Food	Area	Pricerange	Choice	
Attraction	Inform	Name Postcode	Type Phone	Area Choice	Price	Open	Address
	Request	Name	Type	Area	Price		
	Recommend	Name Postcode	Type Phone	Area Choice	Price	Open	Address
	Select	Name Choice	Type	Area	Price	Address	Phone
	NoOffer	Name	Type	Area	Price	Address	Choice
Train	Inform	Id Day	ArriveBy People	LeaveAt Price	Departure Choice	Destination Reference	Duration
	Request	ArriveBy	LeaveAt	Departure	Destination	Day	People
	OfferBooked	Id Day	ArriveBy People	LeaveAt Price	Departure Choice	Destination Reference	Duration
	OfferBook	Id Day	ArriveBy People	LeaveAt Price	Departure Choice	Destination Reference	Duration
	Select	Id People	ArriveBy Price	LeaveAt Choice	Departure	Destination	Day
	NoOffer	Id Choice	ArriveBy	LeaveAt	Departure	Destination	Day
Taxi	Inform	ArriveBy	LeaveAt	Departure	Destination	Car	Phone
	Request	ArriveBy	LeaveAt	Departure	Destination		
Hospital	Inform	Department	Address	Postcode	Phone		
	Request	Department					
Police	Inform	Name	Address	Postcode	Phone		
Booking	Inform	Name	Time	Stay	Day	People	Reference
	Request	Time	Stay	Day	People		
	Book	Name	Time	Stay	Day	People	Reference
	NoBook	Name	Time	Stay	Day	People	Reference
General	Bye	-					
	Greet	-					
	Reqmore	-					
	Welcome	-					

Table C.7: Predefined important system actions on MultiWOZ.

Domain	Action	Slot				
Hotel	Inform	Name	Address	Postcode	Phone	Reference
Restaurant	Inform	Name	Address	Postcode	Phone	Reference
Attraction	Inform	Name	Address	Postcode	Phone	
Train	Inform	Id	Reference			
	OfferBooked	Id	Reference			
	OfferBook	Id				
Taxi	Inform	Phone				
Booking	Inform	Name				
	Book	Name	Reference			

Appendix D. Sample Dialogues

We report sample dialogues generated by DORA. Table D.8 shows the results on PMUL2275 dialogue of Multi-WOZ. Table D.9, D.10 show the results on interactive dialogues with a user.

Table D.8: Sample generation results of DORA on the PMUL2275 dialogue of test set.

User:	what is the address for holy trinity church?
Belief:	{attraction-name: holy trinity church}
Action:	[attraction]-[inform]-postcode, [attraction]-[inform]-address, [attraction]-[inform]-name, [general]-[reqmore]
Response:	[attraction_name] is located on [attraction_address], postcode [attraction_postcode]. do you need any other information?
User:	thank you. i wish to get entrance fee
Belief:	{attraction-name: holy trinity church}
Action:	[attraction]-[inform]-price, [attraction]-[inform]-phone, [attraction]-[inform]-name, [attraction]-[inform]-postcode
Response:	[attraction_name] has free admission. their phone number is [attraction_phone]. the postcode is [attraction_postcode].
User:	yes, i am looking for an indian restaurant in the east.
Belief:	{restaurant-food: indian, restaurant-area: east, attraction-name: holy trinity church}
Action:	[booking]-[inform], [restaurant]-[inform]-name, [restaurant]-[inform]-address, [restaurant]-[inform]-phone
Response:	i have the [restaurant_name] located at [restaurant_address]. the phone number is [restaurant_phone]. would you like to make reservations?
User:	how about an expensive one. what are my choices in that range?
Belief:	{restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Action:	[booking]-[inform], [restaurant]-[inform]-name, [restaurant]-[inform]-address, [restaurant]-[inform]-phone, [restaurant]-[inform]-phone
Response:	there is one. [restaurant_name] is located at [restaurant_address]. their phone number is [restaurant_phone]. would you like to book a table?
User:	i want to book for 7 people at 18:00 on monday
Belief:	{restaurant-time: 18:00, restaurant-day: monday, restaurant-people: 7, restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Action:	[booking]-[book]-reference, [booking]-[book]-name
Response:	i have booked you at [restaurant_name]. your reference number is [restaurant_reference].

User:	yes, one more thing. i will need a taxi from holy trinity church to pipasha restaurant.
Belief:	{taxi-destination: pipasha restaurant, taxi-departure: holy trinity church, restaurant-time: 18:00, restaurant-day: monday, restaurant-people: 7, restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Action:	[taxi]-[request]-arrive, [taxi]-[request]-leave
Response:	what time would you like to leave or arrive by?
User:	i need to arrive by the booked time at the restaurant.
Belief:	{taxi-destination: pipasha restaurant, taxi-departure: holy trinity church, taxi-arrive: 18:00, restaurant-time: 18:00, restaurant-day: monday, restaurant-people: 7, restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Action:	[general]-[reqmore], [taxi]-[inform]-phone, [taxi]-[inform]-car
Response:	i have booked a [taxi_car] for you. the contact number is [taxi_phone]. is there anything else i can help you with?
User:	there will be 7 people travelling.
Belief:	{taxi-destination: pipasha restaurant, taxi-departure: holy trinity church, taxi-arrive: 18:00, restaurant-time: 18:00, restaurant-day: monday, restaurant-people: 7, restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Action:	[taxi]-[inform]-phone, [taxi]-[inform]-car
Response:	i have booked a [taxi_car] for you. the contact number is [taxi_phone].
User:	thank you. that is all i need.
Belief:	{taxi-destination: pipasha restaurant, taxi-departure: holy trinity church, taxi-arrive: 18:00, restaurant-time: 18:00, restaurant-day: monday, restaurant-people: 7, restaurant-food: indian, restaurant-pricerange: expensive, restaurant-area: east, attraction-name: holy trinity church}
Response:	you're welcome, have a great day!

Table D.9: Sample generation results of DORA on the interactive setting.

User:	i need information about a guesthouse called warkworth house.
Belief:	{hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[booking]-[inform], [hotel]-[inform]-area, [hotel]-[inform]-name, [hotel]-[inform]-address, [hotel]-[inform]-phone, [hotel]-[inform]-postcode
Response:	[hotel_name] is located in the [hotel_area] at [hotel_address] [hotel_postcode]. their phone number is [hotel_phone]. would you like to book a room?
User:	yes, book a room for two people.
Belief:	{hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[booking]-[request]-day, [booking]-[request]-stay
Response:	what day will you be arriving and how long will you be staying?
User:	for two days from monday.
Belief:	{hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[booking]-[book]-reference, [general]-[reqmore]
Response:	booking was successful. reference number is : [hotel_reference]. can i help with anything else?
User:	i want to book a indian restaurant called curry prince.
Belief:	{restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[booking]-[request]-people, [restaurant]-[inform]-name, [restaurant]-[inform]-address
Response:	[restaurant_name] is located at [restaurant_address]. how many people will be dining?
User:	book the restaurant for two people at 19:300 on monday.

Belief:	{restaurant-time: 19:30, restaurant-day: monday, restaurant-people: 2, restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[booking]-[book]-reference, [general]-[reqmore]
Response:	booking was successful. the table will be reserved for 15 minutes. reference number is : [restaurant_reference]. is there anything else i can help you with?
User:	can you give me the phone number?
Belief:	{restaurant-time: 19:30, restaurant-day: monday, restaurant-people: 2, restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[restaurant]-[inform]-phone, [restaurant]-[inform]-name
Response:	the phone number for [restaurant_name] is [restaurant_phone]
User:	i also need a taxi from the guesthouse to the restaurant.
Belief:	{taxi-destination: curry prince, taxi-departure: warkworth house, restaurant-time: 19:30, restaurant-day: monday, restaurant-people: 2, restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[taxi]-[request]-leave, [taxi]-[request]-arrive
Response:	what time would you like to leave or arrive by?
User:	i will leave at 19:00.
Belief:	{taxi-leave: 19:00, taxi-destination: curry prince, taxi-departure: warkworth house, restaurant-time: 19:30, restaurant-day: monday, restaurant-people: 2, restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[general]-[reqmore], [taxi]-[inform]-phone, [taxi]-[inform]-car
Response:	i have booked a [taxi_car] for you. the contact number is [taxi_phone]. is there anything else i can help you with?
User:	thank you for your help. have a nice day!
Belief:	{taxi-leave: 19:00, taxi-destination: curry prince, taxi-departure: warkworth house, restaurant-time: 19:30, restaurant-day: monday, restaurant-people: 2, restaurant-food: indian, restaurant-name: curry prince, hotel-stay: 2, hotel-day: monday, hotel-people: 2, hotel-name: warkworth house, hotel-type: guesthouse}
Action:	[general]-[bye], [general]-[welcome]
Response:	you're welcome! have a great day!

Table D.10: Another sample generation results of DORA on the interactive setting.

User:	i am looking for a cheap hotel in centre city. can you recommend one for me?
Belief:	{hotel-area: centre, hotel-pricerange: cheap}
Action:	[hotel]-[inform]-name, [hotel]-[inform]-address, [hotel]-[inform]-phone, [hotel]-[inform]-postcode
Response:	i have [hotel_name] located at [hotel_address] [hotel_postcode]. their phone number is [hotel_phone].
User:	i will stay alone. can you book the hotel from friday to sunday?
Belief:	{hotel-stay: 3, hotel-day: friday, hotel-people: 1, hotel-area: centre, hotel-pricerange: cheap}
Action:	[booking]-[book]-reference, [booking]-[book]-name, [general]-[reqmore]
Response:	i have you booked at the [hotel_name]. your reference number is [hotel_reference]. can i help with anything else?

User:	i also need a italian restaurant in the same area.
Belief:	{restaurant-food: italian, restaurant-area: centre, hotel-stay: 3, hotel-day: friday, hotel-people: 1, hotel-area: centre, hotel-pricerange: cheap}
Action:	[restaurant]-[inform]-name, [restaurant]-[inform]-address, [restaurant]-[inform]-phone
Response:	[restaurant_name] is located at [restaurant_address] and their phone number is [restaurant_phone].
User:	book the restaurant on 18:00 saturday for one
Belief:	{restaurant-time: 18:00, restaurant-day: saturday, restaurant-people: 1, restaurant-food: italian, restaurant-area: centre, hotel-stay: 3, hotel-day: friday, hotel-people: 1, hotel-area: centre, hotel-pricerange: cheap}
Action:	[booking]-[book]-reference, [general]-[reqmore]
Response:	booking was successful. the table will be reserved for 15 minutes. reference number is : [restaurant_reference]. anything else i can help you with?
User:	can you recommend a architecture for me in the same area?
Belief:	{restaurant-time: 18:00, restaurant-day: saturday, restaurant-people: 1, restaurant-food: italian, restaurant-area: centre, hotel-stay: 3, hotel-day: friday, hotel-people: 1, hotel-area: centre, hotel-pricerange: cheap, attraction-type: architecture, attraction-area: centre}
Action:	[attraction]-[recommend]-name, [attraction]-[recommend]-address, [attraction]-[recommend]-postcode
Response:	i recommend [attraction_name], it is located at [attraction_address], postcode [attraction_postcode].
User:	that's all i need. thank you
Belief:	{restaurant-time: 18:00, restaurant-day: saturday, restaurant-people: 1, restaurant-food: italian, restaurant-area: centre, hotel-stay: 3, hotel-day: friday, hotel-people: 1, hotel-area: centre, hotel-pricerange: cheap, attraction-type: architecture, attraction-area: centre}
Action:	[general]-[bye], [general]-[welcome]
Response:	you're welcome. have a great day!

References

Budzianowski, P., Wen, T.H., Tseng, B.H., Casanueva, I., Ultes, S., Ramadan, O., Gasic, M., 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, pp. 5016–5026.

Cho, K., van Merriënboer, B., Bahdanau, D., Bengio, Y., 2014. On the properties of neural machine translation: Encoder–decoder approaches, in: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, pp. 103–111.

Das, A., Kottur, S., Moura, J.M., Lee, S., Batra, D., 2017. Learning cooperative visual dialog agents with deep reinforcement learning, in: Proceedings of the IEEE international conference on computer vision, pp. 2951–2960.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding, 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), pp. 4171–4186.

Eric, M., Goel, R., Paul, S., Sethi, A., Agarwal, S., Gao, S., Kumar, A., Goyal, A., Ku, P., Hakkani-Tur, D., 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines, in: Proceedings of The 12th Language Resources and Evaluation Conference, pp. 422–428.

Gupta, R., Rastogi, A., Hakkani-Tür, D., 2018. An efficient approach to encoding context for spoken language understanding. Proc. Interspeech 2018 , 3469–3473.

Hosseini-Asl, E., McCann, B., Wu, C.S., Yavuz, S., Socher, R., 2020. A simple language model for task-oriented dialogue. arXiv preprint arXiv:2005.00796 .

Kim, S., Yang, S., Kim, G., Lee, S.W., 2020. Efficient dialogue state tracking by selectively overwriting memory, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 567–582.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization, in: Bengio, Y., LeCun, Y. (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. URL: `http://arxiv.org/abs/1412.6980`.

Kottur, S., Moura, J., Lee, S., Batra, D., 2017. Natural language does not emerge 'naturally'in multi-agent dialog, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2962–2967.

Lee, H., Jo, S., Kim, H., Jung, S., Kim, T.Y., 2020. Sumbt+ larl: End-to-end neural task-oriented dialog system with reinforcement learning. arXiv preprint arXiv:2009.10447 .

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L., 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880.

Lewis, M., Yarats, D., Dauphin, Y., Parikh, D., Batra, D., 2017. Deal or no deal? end-to-end learning of negotiation dialogues, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2443–2453.

Lin, Z., Madotto, A., Winata, G.I., Fung, P., 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3391–3405.

Lubis, N., Geishauser, C., Heck, M., Lin, H.c., Moresi, M., van Niekerk, C., Gasic, M., 2020. Lava: Latent action spaces via variational auto-encoding for dialogue policy optimization, in: Proceedings of the 28th International Conference on Computational Linguistics, pp. 465–479.

Peng, B., Li, C., Li, J., Shayandeh, S., Liden, L., Gao, J., 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. arXiv preprint arXiv:2005.05298 .

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., 2019. Language models are unsupervised multitask learners .

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683 .

Wang, J., Zhang, Y., Kim, T.K., Gu, Y., 2020. Modelling hierarchical structure between dialogue policy and natural language generator with option framework for task-oriented dialogue system. arXiv preprint arXiv:2006.06814 .

Williams, R.J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning 8, 229–256.

Yang, Y., Li, Y., Quan, X., 2020. Ubar: Towards fully end-to-end task-oriented dialog systems with gpt-2. arXiv preprint arXiv:2012.03539 .

Zhang, Y., Ou, Z., Hu, M., Feng, J., 2020a. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 9207–9219.

Zhang, Y., Ou, Z., Yu, Z., 2020b. Task-oriented dialog systems that consider multiple appropriate responses under the same context, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 9604–9611.

Zhao, T., Xie, K., Eskenazi, M., 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), pp. 1208–1218.