

# RLPA-ODDS: An Open-domain Dialogue System Framework based on Reinforcement Learning with Prompts as Actions

Tao Xiang<sup>✉</sup>, Yangzhe Li<sup>✉</sup>, Ana Pecini<sup>✉</sup>, and Dominika Młynarczyk<sup>✉</sup>

Department of Informatics, Technical University of Munich

✉ tao.xiang@tum.de

✉ yangzhe.li@tum.de

✉ ana.pecini@tum.de

✉ dominika.mlynarczyk@tum.de

August 8, 2022

**Abstract** — The performance of most current open-domain dialogue systems is limited to the (training) dialogue corpora due to either generation-based or retrieval-based learning patterns. To break the limitation by (training) dialogue corpora, we propose RLPA-ODDS, an **Open-domain Dialogue System** framework based on **Reinforcement Learning with Prompts as Actions**. This framework requires a (fixed) open-domain dialogue system as the backbone and trains a behavior policy using reinforcement learning to guide the backbone system to respond appropriately with respect to the conversation situations. Here the action space is defined as a finite set of behaviors in form of natural language prompts. We show that with the guidance of the behavior policy, the backbone system could generate more empathetic and diverse responses.

## 1 Introduction

Open-domain dialogue systems is a popular natural language processing (NLP) task because of its promise in real-life applications, such as Microsoft XiaoIce<sup>1</sup>, Google Meena [1], Facebook Blenderbot[2], etc. Current methods for open-domain dialogue systems can be generally categorized into retrieval-based methods and generation-based methods, where both kinds of methods require high-quality dialogue corpora: Retrieval-based systems need a pre-collected paired conversation dataset for retrieving responses, while generation-based systems need a large amount of annotated training data for supervised learning [3]. Therefore, the performance of dialogue systems depends heavily on the quality of the dialogue corpus and in theory, it is difficult for dialogue systems trained with these methods to exceed the level of the training set.

Inspired by recent reinforcement learning applications that beat human performance such as AlphaGo<sup>2</sup>, in this work we explore whether reinforcement learning can further improve the performance of dialogue systems, in order to outperform training set level or even reach a human-like level. Furthermore, similar to the idea from Yann LeCun [4], we try to simulate the real human brain mechanism: Human brain has a *perception* module and an *actor* module. During the conversation, the perception module first analyzes the current conversation situation and chooses the next action accordingly, then passes this information to the actor module that actually generates the natural language. More specifically, for the perception module, we train a behavior policy that decides which system action to perform according to the current conversation situation with reinforcement learning. The system actions are defined as general human behaviors during one-on-one conversations in form of natural language prompts, such as "greeting the other", "comforting the other", "congratulating the other", "giving advice to the other", etc. For the actor module, we use a fixed dialogue system that has been pre-trained on dialogue corpora in a supervised fashion. It takes the dialogue history together with the system action output by the policy as input and generates a response.

We further formulate this approach as RLPA-ODDS, an **Open-domain Dialogue System** framework based on **Reinforcement Learning with Prompts as Actions**. It's considered as a framework because the definition of the action space, the backbone dialogue system, and the training of the policy network could all be modified in future work. The general pipeline for this framework is given as follows:

---

<sup>1</sup><https://www.xiaoice.com/>

<sup>2</sup><https://www.deeppmind.com/research/highlighted-research/alphago>

1. define the action space. Actions are any acceptable behaviors during conversation in form of natural language prompts such as "saying goodbye to the other".
2. train or use a pre-trained open-domain dialogue system as the fixed backbone.
3. train a policy network that maps dialogue history to actions with reinforcement learning.
4. concatenate dialogue history and action outputted from policy network as input and feed to the backbone to generate responses.

The sections in this paper are organized as follows: In section 2 we review related work in recent years, in section 3 we dive into details of our methods, in section 4 we introduce the evaluation strategies used in this work, in section 5 we demonstrate and analyze the evaluation results and in section 6 we conclude our work and discuss possible future work.

## 2 Related Work

### 2.1 Open-domain Dialogue Systems

In the past few years, the area of open-domain dialogue systems has achieved significant progress with the development of deep learning. Typically, deep learning methods for open-domain dialogue systems can be categorized into retrieval-based and generation-based.

A retrieval-based open-domain dialogue system matches user utterances with present queries in a pre-collected human conversation dataset and retrieves responses of similar queries as candidate responses. Then a scoring algorithm is applied to score these responses and the response with the highest score is selected as the final response. Recent work on retrieval-based systems includes [5], [6], [7], etc. One drawback of retrieval-based systems is that they depend heavily on the pre-collected dataset, which is difficult to collect and construct. What's more, the pre-existed responses can only cover a limited scope of conversations. So they will never be enough for real-world open domain conversations, which include an arbitrarily wide range of topics.

Generation-based dialogue systems, on the other hand, are fully data-driven and possess the potential of generating unseen responses. Recent work on generation-based systems focuses on fine-tuning pre-trained language models on dialogue datasets, such as [8], [9], [1], etc. Although generation-based dialogue systems alleviate the limited scope problem in retrieval-based systems, their performance still heavily depends on the quality of training dialogues.

### 2.2 Reinforcement Learning for Dialogue

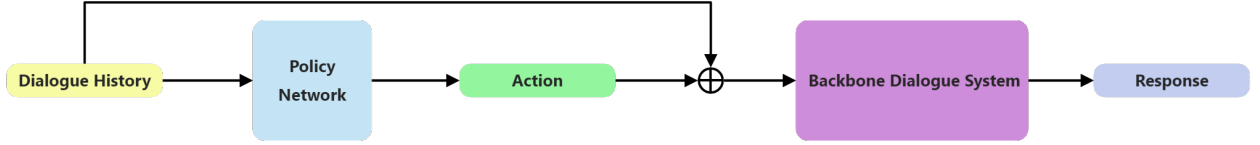
Compared with supervised learning, reinforcement learning (RL) in the area of open-domain dialogue systems is still in the exploratory stage. In recent years, one popular RL research direction is to optimize the dialogue system trained with supervised learning before. For example, Jaques et al. (2019) optimize for sentiment and several other conversation metrics by learning from a static batch of human-bot conversations using Batch RL [10]. Saleh et al. (2019) propose using RL to reduce toxicity in an open-domain dialog setting, in order to ensure the model produces more appropriate and safe conversations [11]. In these settings, the action space is usually infinite with actions being system responses of various lengths. In contrast, Xu et al. (2018) define explicitly an action space consisting of dialogue acts that represent human behaviors during conversation and train a policy model that decides appropriate dialogue acts w.r.t dialogue history [12]. The definition of dialogue acts is appended in Appendix A.

To simulate real human brain mechanism and inspired by the recent popular *prompting* technique (more details in subsection 2.3), in this work we define explicitly an action space similarly to Xu et al. (2018) [12], but with actions defined as natural language prompts. To our best knowledge, we believe we are the first to propose using natural language prompts as actions in RL for optimizing dialogue systems.

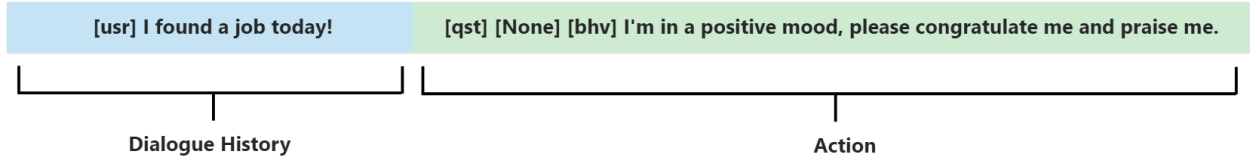
## 2.3 Prompting Language Models

Extending the knowledge of a language model via prompts is an active research direction. Radford et al. (2019) use prompts to guide zero-shot generation without fine-tuning on any prompts [13]. Raffel et al. (2019) use task-specific prompts in both finetuning and testing phase[14]s. More recently, Lee et al. (2021) use natural language descriptions for requested domains and slots as prompts to guide the language model to generate slot value for the requested domain and slot in the dialogue state tracking task. Inspired by this body of work, we propose to use natural language prompts as actions in reinforcement learning to guide the backbone dialogue system to behave accordingly.

## 3 Methods



**Figure 1** Framework structure of RLPA-ODDS.



**Figure 2** Input representation for the backbone dialogue system. We use special tokens [usr], [sys], [qst], and [bhv] to represent a following user utterance, system utterance, question-or-not action, and behavior action respectively.

Type of Actions	Value 1	Value 2	Value 3	Value 4
Question-or-not Actions (QST)	"ask me for further details."	"[None]"		
Behavior Actions (BHV)	"I'm in a positive mood, please congratulate me and praise me."	"I'm in a negative mood, please comfort me."	"give me some advice"	"[None]"

**Table 1** Definition of the actions. We define two values for QST actions and four values for BHV actions. Here "[None]" value represents no actions. The action space  $\mathcal{A}$  is then a two-dimensional space combining QST actions and BHV actions.

### 3.1 Problem Statement & Notation

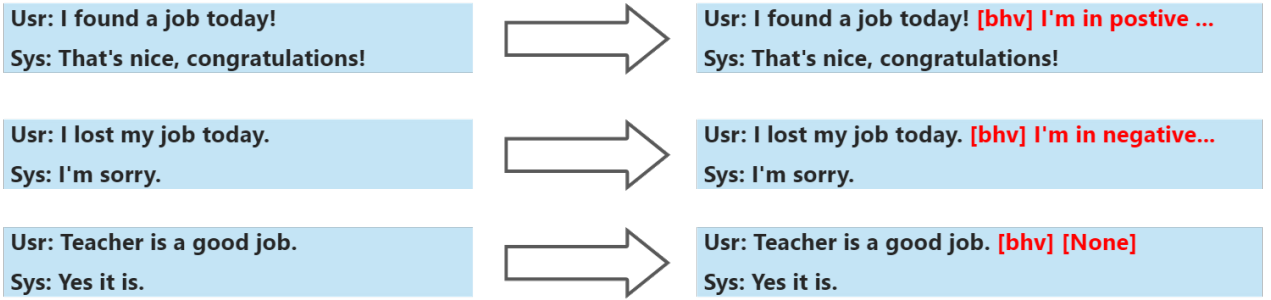
In this section, we introduce the primary notations utilized in this paper and formulate the task briefly. The main task of RLPA-ODDS is to train a behavior policy that takes the dialogue history as input and outputs a behavior action with reinforcement learning. Then we combine the dialogue history and behavior action as input to our fixed pre-trained dialogue system, which we call *backbone*. The backbone then generates a system response. The framework structure of RLPA-ODDS can be found in Figure 1.

We consider a dialogue as a sequence of utterances alternating between two parties,  $U_1, S_1, \dots, U_T, S_T$ , where  $U$  and  $S$  represent the user utterance and the system response, respectively. In a turn  $t$ , the user provides a new utterance  $U_t$  and the system responds with utterance  $S_t$ . Then we denote the dialogue history at turn  $t$  as  $C_t = \{U_1, S_1, \dots, S_{t-1}, U_t\}$ , which excludes the latest system response  $S_t$ . Furthermore, we denote the action space in RL as  $\mathcal{A}$ , and the policy network is defined as  $\pi = P(A_t | C_t)$  with  $A_t \in \mathcal{A}$ .

### 3.2 Action Space Definition

As discussed above, in this work we define the action space  $\mathcal{A}$  as a finite set of general human behaviors during one-on-one conversation. More specifically, we define two types of actions: *question-or-not* actions and *behavior* actions. Question-or-not actions indicate whether the next system response should be a question, whereas behavior actions represent behaviors the backbone system should perform next. In this implementation we define four human behaviors for the backbone system: 1) to congratulate and praise the user 2) to comfort the user 3) to give the user some advice 4) no behaviors. The concrete values for these two types of actions are listed in Table 1. Note that the values are in form of natural language prompts, and in particular, they are written in form of requesting the system (from the user’s perspective). We define so because later we will concatenate these action values with the last user utterance  $U_t$  from the dialogue history  $C_t$ , and feed it as input to the backbone system. An example input representation can be found in Figure 2. Such processing has the same effect as the user says additionally these prompts and we believe it will easily guide the backbone system to better understand the user sentiment and thus generate more appropriate and emotional responses.

### 3.3 Backbone Dialogue System



**Figure 3** Illustration of augmenting the EmpatheticDialogues dataset.

For the backbone dialogue system, we use Blenderbot-400M-distill from Facebook [2], which is a generation-based seq2seq transformer model and trained on Blended Skill Talk dataset [15]. To let the backbone system better understand the prompts, we further fine-tuned the Blenderbot on the augmented EmpatheticDialogues dataset [16]. More specifically, we first augment EmpatheticDialogues dataset by appending suitable prompts to each last user utterance  $U_t$  in dialogue history  $C_t$ , then we fine-tuned the Blenderbot on this dataset for only 10 epochs to avoid overfitting. An example of augmenting the dataset can be found in Figure 3. In implementation, to assign a proper question-or-not prompt to each dialogue, we simply check whether the current system response has a question mark. For behavior prompts, we train a sentiment classifier that can tell whether the user is in a positive/negative/neutral emotion and then add prompts according to the classification. To train this classifier, we first manually label 500 dialogue samples from EmpatheticDialogues dataset with the label set defined as {positive, negative, neutral}. Then we use this classifier to tag the whole EmpatheticDialogues dataset and finally, we manually review this augmented dataset and revise obvious classification mistakes. We have uploaded the fine-tuned Blenderbot<sup>3</sup>, the sentiment classifier<sup>4</sup> and the augmented EmpatheticDialogues dataset<sup>5</sup> onto Huggingface.

The purpose of the fine-tuning is to make sure the backbone system can understand action prompts and respond accordingly. This consistency between action prompts and system responses is necessary for later reinforcement learning.

<sup>3</sup>[https://huggingface.co/Adapting/dialogue\\_agent\\_nlp1ab2022](https://huggingface.co/Adapting/dialogue_agent_nlp1ab2022)

<sup>4</sup>[https://huggingface.co/Adapting/comfort\\_congratulations\\_neutral-classifier](https://huggingface.co/Adapting/comfort_congratulations_neutral-classifier)

<sup>5</sup>[https://huggingface.co/datasets/Adapting/empathetic\\_dialogues\\_with\\_special\\_tokens](https://huggingface.co/datasets/Adapting/empathetic_dialogues_with_special_tokens)

### 3.4 Policy Network

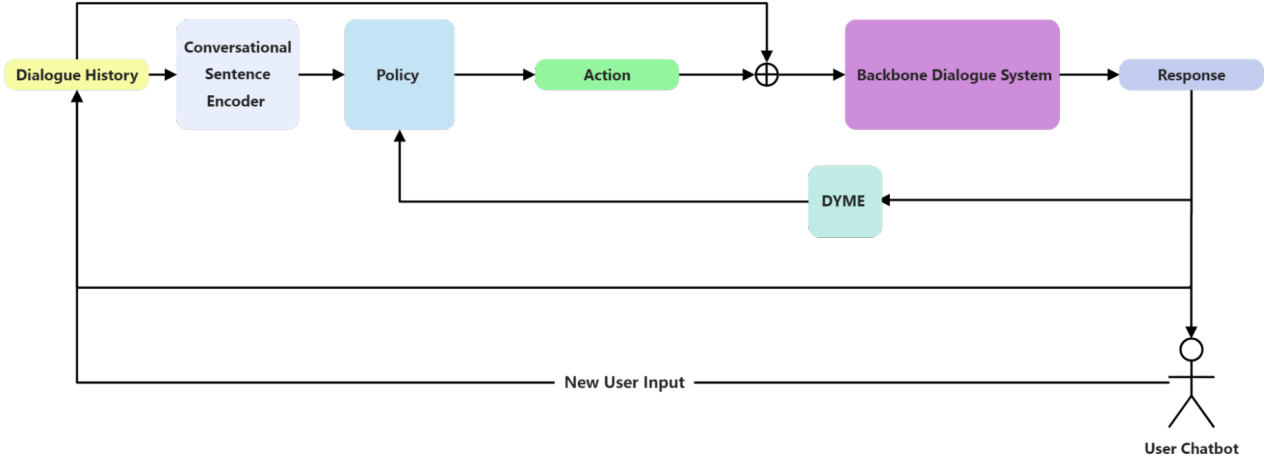
#### 3.4.1 Model Architecture

To select appropriate actions according to different conversation situations, we design a policy network that takes dialogue history as input and outputs the action prompts defined in subsection 3.2. In implementation, the policy network consists of only fully-connected layers. The input is the embedding of dialogue history and the output is two-dimensional logits, where the first dimension represents behavior actions and the second represents question-or-not actions.

To map the logits to corresponding action values, we further employ activation functions to restrict the logits into a fixed interval, then we slice this interval into subintervals and each subinterval corresponds to a certain action value. For example, for the second dimension logits we apply Tanh function so that the interval is  $(-1, 1)$ . Then we divide the interval into two subintervals  $(-1, 0]$  and  $(0, 1)$ , where  $(-1, 0]$  corresponds to the QST action value "Ask me for further details" and  $(0, 1)$  corresponds to the QST action value "[None]".

To get the embedding of dialogue history, we use a pre-trained conversational representation model Conversational Sentence Encode (ConveRT) and keep it fixed. ConveRT is a specialized encoder that can compress the dialogue history into a 512-dimensional embedding. We further apply the arctan function to this embedding element-wisely so that the domain of each dimension is restricted to  $(-1, 1)$ . We believe such processing can improve exploration efficiency for later training while ensuring distinguishability of the embeddings around the origin.

#### 3.4.2 Training



**Figure 4** Flow for training the policy network.

In order to make the policy network close to or even exceed the level of human decision-making, we train it with reinforcement learning. More specifically, we choose Soft Actor-Critic (SAC) algorithm, which is an off-policy actor-critic deep RL algorithm based on the maximum entropy reinforcement learning [17]. The reason we use SAC is that it can explore very diverse policies and preserve near-optimal policies while pursuing convergence as much as possible. This fits our behavior policy quite a lot since human behaviors could be very complicated, and different people often react differently to the same conversation situation.

The process of training the policy network can be divided into the following parts: 1) action decision & system response generation 2) reward calculation 3) new user input generation 4) dialogue history update & repeat. An example training flow can be found in Figure 4.

**1) action decision & system response generation.** Firstly, the embedding of dialogue history  $C_t = \{U_1, S_1, \dots, S_{t-1}, U_t\}$  is fed to the policy network, which then selects an appropriate action. The action (prompts) is then concatenated with the original dialogue history and fed as input to the backbone dialogue system, which then generates a new system response  $S_t$ . An input example for the backbone dialogue system can be found in Figure 2.

**2) reward calculation.** Once we have the generated system response  $S_t$ , we compute a reward for it. We train a metric model, namely DYnamic MEtric for dialog modeling (DYME) [18], on EmphateticDialogues [16] and DailyDialog [19] datasets. DYME is a model that can predict utterance metrics of the next sentence for a given dialogue. DYME in total considers 14 metrics such as emotional reaction level, conversation repetition, etc.

More specifically, we first use DYME to compute a ground truth utterance metrics of the next system response given the current dialogue history  $C_t$ , denoted as  $m_t \in \mathbb{R}^{14}$ . Then we use the same metric algorithms as in DYME to compute utterance metrics of the generated response  $S_t$ , denoted as  $\hat{m}_t \in \mathbb{R}^{14}$ . The reward function is defined as a distance function that measures the similarity between  $m_t$  and  $\hat{m}_t$ , denoted as  $l : \mathbb{R}^{14} \times \mathbb{R}^{14} \rightarrow \mathbb{R}$ . In this work, we use the negative mean square error function as the reward function:

$$l(m_t, \hat{m}_t) = -\frac{1}{n} \sum_{i=1}^n (m_{t_i} - \hat{m}_{t_i})^2 \quad (1)$$

**3) new user input generation.** To continue the conversation with the backbone dialogue system, a new user input  $U_{t+1}$  is required to reply the system response  $S_t$ . In our setup, real human interaction or another user chatbot are both allowed to produce user inputs. For the training, we mostly employ a user chatbot, namely Blenderbot-1B-distill from Facebook [2], which is a variant of the backbone system but with more parameters.

**4) dialogue history update & repeat.** Once we have the new system response  $S_t$  and user input  $U_{t+1}$ , we update the dialogue history as:  $C_{t+1} = C_t \oplus \{S_t, U_{t+1}\}$ , where  $\oplus$  stands for concatenation operation. Then we repeat the whole training process.

## 4 Evaluation

We compare the performance of the baseline and RLPA-ODDS in two folds: automatic and human evaluation. Here the baseline is the fine-tuned Blenderbot introduced in subsection 3.3. The dataset we use for the baseline and RLPA-ODDS to generate responses is the test set of the augmented EmphateticDialogues dataset. Notice that we have removed the action prompts in each dialogue history.

### 4.1 Automatic Evaluation

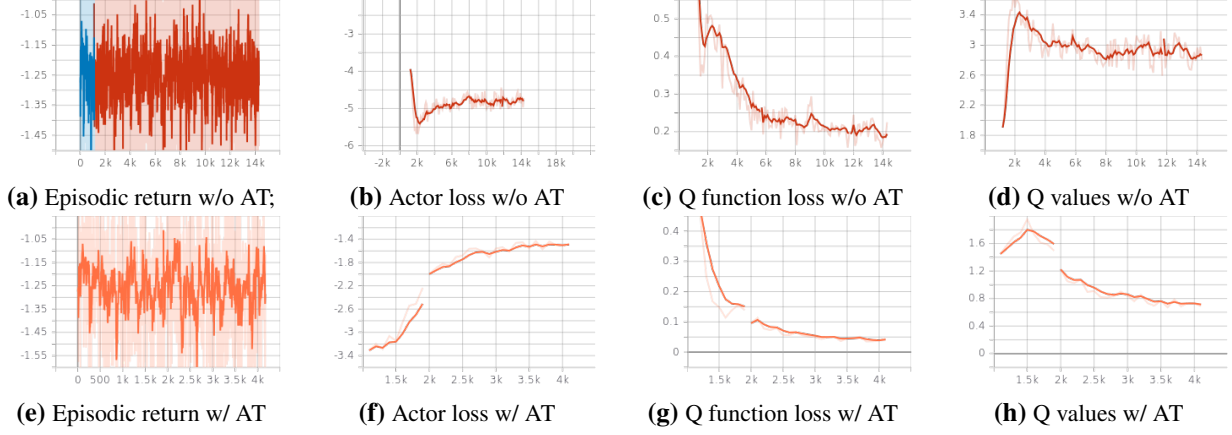
For automatic evaluation, we use METEOR metric [20] and FED metric [21]. METEOR is a word-overlap metric that calculates the similarity between the generated sequence and the ground truth sequence in word-level, whereas FED is a neural metric that can measure fine-grained dialogue qualities at both the turn and whole dialogue levels. More specifically, FED can assess eighteen qualities of dialogue without relying on a reference response and has shown moderate to strong correlation with human judgment. Example qualities are “diverse”, “coherent”, “fluent” and etc. We also put a screenshot of all qualities from the original paper in Appendix B. In this experiment, we consider eleven qualities in view of their relative importance stated in the original paper, as shown in Table 2.

### 4.2 Human Evaluation

For human evaluation, we randomly sample 300 conversations and score the samples independently from the range of 1 to 5 together with a few volunteers in terms of empathy, relevance, and fluency. These volunteers are also university educated and share a CS-related background. To represent and alleviate subjectivity, we use inter-rater agreement and duplicate evaluation approach. More specifically, each rater first evaluates the same 20 samples, from which the agreement is calculated by Krippendorff’s Alpha [22] with the level of measurement being “interval”. Then each rater evaluates 80 more samples, among which 40 samples are overlapped. This ensures that each sample is rated by at least two persons. In addition, for the purpose of blind evaluation, we randomly disrupt the responses from RLPA-ODDS and the baseline, and mark the name as “response A” and “response B”.

## 5 Results

### 5.1 Training Results of Policy Network



**Figure 5** Training results of two experimental procedures that are with (abbreviated as “w/”) and without (abbreviated as “w/o”) auto-tuning (AT). The blue part in Figure 5a indicates the random sampling in the initial 1k steps. Q values are represented by values from Qf1 because of double Q learning.

In our experimental process of training, we observe that the actor in the policy network begins exploration at a very early stage. Figure 5a and Figure 5e show an inconspicuous improvement around 2k steps (1k steps for the random data collection phase, which should be 5k steps generally). The actor loss (Figure 5b) experiences a rapid decline and then rebounded, so we suspect that the training data collected in the early stage has already been exploited and the actor has turned to slow exploration after 2k steps. The loss of the Q function (Figure 5c) declines steadily while the value (Figure 5d) decreases and stabilizes after reaching the peak at steps around 2k, which means that the Q function is overestimated at the beginning, also indicating the completion of exploitation. In the case of auto-tuning, the actor loss (Figure 5f) increases instead of decreases, along with the Q values (Figure 5h) together also reflect that the exploration is carried out without too much exploitation.

Note that SAC and other RL algorithms often require millions of training steps to achieve significant results [17], while the limitation of the capability only allows us to make a preliminary verification of the algorithm.

### 5.2 Analysis of Automated Metrics

	Blenderbot	RLPA-ODDS
METEOR	0.1605	<b>0.1627</b>
Engaging	0.3698	<b>0.3701</b>
Semantically Appropriate	-0.2425	<b>-0.2354</b>
Specific	0.2020	<b>0.2118</b>
Relevant	<b>-6.3218</b>	-6.4173
Correct	<b>-6.3973</b>	-6.4928
Understandable	-0.2254	<b>-0.2161</b>
Fluent	-0.2454	<b>-0.2350</b>
Coherent	<b>-6.2809</b>	-6.3813
Consistent	<b>-6.5760</b>	-6.6770
Diverse	<b>-6.4907</b>	-6.5906
Likeable	0.2269	<b>0.2283</b>

**Table 2** Results of automatic evaluation. The higher scores are in bold. Notice that we negate the raw scores output by FED since originally the lower the scores are, the better the performance on those metrics is.

From the METEOR scores in Table 2, we can see that although RLPA-ODDS has a higher score, both scores are small and the difference is not that significant. This result is acceptable since word-overlap metrics are inefficient in dialogues and there may be many possible responses for one user utterance. Therefore, an acceptable response may receive a low score [23].

Due to this inefficiency, we come to the neural metric FED. From the results in Table 2 we can see RLPA-ODDS beats the baseline in engagement, semantic appropriateness, specificity, understandability, fluency, and likeability, with the score difference being around 0.01. We believe that the reason why RLPA-ODDS is making progress relative to the baseline on these metrics is due to the explicit behavior decisions. More specifically, the “asking details”, “congratulating”, “comforting”, and “giving advice” actions makes RLPA-ODDS more engaging, specific, fluent, and likeable during conversation. However, due to limited training time, the behavior policy is still not perfect and sometimes it makes mistakes. For instance, in cases where the chatbot shouldn’t ask for further details but it does, then it may ask off-topic or irrelevant questions. In scenarios like this, RLPA-ODDS may generate less relevant, coherent, or consistent responses. Therefore, the performance on these metrics degrades. Also, because we only defined four behavior actions, RLPA-ODDS shows less diversity compared to the baseline.

### 5.3 Analysis of Human Evaluation

	Empathy	Relevance	Fluency	Total
Blenderbot (All raters)	3.114	3.114	<b>3.607</b>	<b>3.279</b>
RLPA-ODDS (All raters)	<b>3.250</b>	<b>3.129</b>	3.236	3.205
Blenderbot (Volunteers excl.)	<b>3.559</b>	<b>3.498</b>	<b>3.697</b>	<b>3.585</b>
RLPA-ODDS (Volunteers excl.)	3.411	3.354	3.309	3.358
Blenderbot (300 samples)	<b>3.551</b>	<b>3.418</b>	<b>3.585</b>	<b>3.518</b>
RLPA-ODDS (300 samples)	3.418	3.235	3.357	3.337

**Table 3** Results of human evaluation. Here “All raters” indicates the average scores of the 20 samples evaluated by all raters. “Volunteers excl.” indicates the average scores of 140 samples evaluated only by us, but without the volunteers. “300 samples” indicates the average scores of all 300 samples. Notice that only the 20 samples are evaluated by all raters, while the remaining samples are evaluated by two random raters.

	General	Empathy	Relevance	Fluency
All raters	0.256	0.290	0.278	0.149
Volunteers excl.	0.400	0.446	0.496	0.221

**Table 4** Agreement of raters on the 20 samples.

From the results in Table 3, we can see RLPA-ODDS outperforms the baseline in empathy and relevance with respect to the 20 samples evaluated by all raters. However, the performance of RLPA-ODDS drops when it comes to the samples evaluated by only two random raters. We think the reason is the strong personal subjectivity.

The agreement results in Table 4 show very low agreements on each metric between the raters. Among them, agreements in empathy and relevance are much higher compared to the agreement in fluency. We think it’s because the raters have relatively diverse opinions about fluency than empathy and relevance. We also calculate the adjacency matrices for all raters with Krippendorff’s Alpha and get similar agreement results. We put these matrices in Appendix C.

Due to the subjectivity of individuals and the raters are not experts themselves, we believe that the results of this human evaluation do not reflect the real performance of the models. In the future, we may ask for expert opinions for more precise human evaluation.



## 5.4 Advantages & Limitations

To show the advantages and limitations of RLPA-ODDS, we cherry-picked four conversation examples generated during training.

### Conversation A

---

<b>User:</b>	I don't know. I guess I just have to get over it and not let it get to me. ( <i>ask details; give advice</i> )
<b>RLPA:</b>	I guess the only thing you can do is try to ignore it. If they continue to do it, they will realize they are causing a problem. Otherwise, it might be time to sit down and have a talk with them about it.
<b>User:</b>	That's what I'm going to have to do. I just don't know what to say to them. ( <i>ask details, give advice</i> )
<b>RLPA:</b>	I would just have a heart to heart with them and tell them know how you feel. You want to be there for them, but you have to take care of yourself, too.

---

**Table 5** Conversation example A. The actions output by the policy network are attached to each user utterance in italic. Here “ask details” and “give advice” represents the QST action "ask me for further details" and BHV action "give me some advice" respectively.

### Conversation B

---

<b>User:</b>	Hi, I'd like to return this radio. ( <i>ask details; congratulate me</i> )
<b>RLPA:</b>	That's good to hear. Why do you want to return it? Is it something wrong with it?
<b>User:</b>	I don't think so, but I just don't want to deal with the hassle of returning it. ( <i>none; comfort me</i> )
<b>RLPA:</b>	I understand. Perhaps you can give it a try and if you like it, you can return it.

---

**Table 6** Conversation example B. Additionally, “congratulate me” and “comfort me” represents the BHV action "I'm in a positive mood, please congratulate me and praise me" and the BHV action "I'm in a negative mood, please comfort me." respectively.

### Conversation C

---

<b>User:</b>	Hello, this is Garry Hao. I am calling to inquire the position of English interpreter. ( <i>none; give advice</i> )
<b>RLPA:</b>	Really? That's so cool! You should get on that, it could open you up to some great opportunities!

---

**Table 7** Conversation example C. Additionally, “none” represents "[None]" value.

### Conversation D

---

<b>User:</b>	Mom, you know that Andrea and I sometimes worry about you. ( <i>ask details; comfort me</i> )
<b>RLPA:</b>	I worry about my mom all the time. Why do you worry about me? What happened to you?
<b>User:</b>	I don't know, I guess I'm just worried that she's going to get sick and I won't be able to take care of her. ( <i>none; give advice</i> )
<b>RLPA:</b>	You can take good care of your mom. Don't worry about the things that worry you. Everything will be okay.

---

**Table 8** Conversation example D.

#### 5.4.1 Advantages

- **Strong controllability.** Because of explicit behavior decision, RLPA-ODDS shows strong controllability. More specifically, the behavior actions output by the policy network can in fact guide the backbone dialogue

system to generate a corresponding response. For example, in Table 5 the two BHV actions are both to give the user some advice, and consistently, the backbone generates relevant suggestions for the user.

- **Dynamic emotion capture.** RLPA-ODDS shows the ability of dynamic user emotion capture, which means it can recognize user emotion changes as the conversation progresses. One example is in Table 6, where the user’s emotion changes from positive to negative, and the backbone responds correspondingly.

#### 5.4.2 Limitations

However, there are still several limitations for RLPA-ODDS.

- **Not comprehensive behaviors.** In this work we only define four behavior actions, which is far too little compared to real human behaviors. Consequently, there are cases where no behavior action is suitable, such as in Table 7. In this conversation example, the user is looking for a job and probably he expects some concrete advice or instructions about getting the job from the chatbot. However, RLPA-ODDS only gives very general and irrelevant advice. Therefore, in the future we can define more topic-related or task-oriented behaviors so that RLPA-ODDS can handle more complicated cases.
- **Lengthy action prompts.** We found that the “congratulating” and “comforting” action prompts are lengthy, which can change the original semantic to some extent, especially when the dialogue history is short. An example is in Table 6, where RLPA-ODDS congratulates the user even if he didn’t show positive sentiment. This could also be a reason that degrades the coherency and consistency as shown in subsection 5.2. Therefore, in the future we could use simpler and clearer prompts such as “I am happy” and “I am sad”, or we could give a weight to the action prompts so that they do not dominate.
- **Coarse-grained action space.** We found the designed two-dimensional action space is coarse-grained since we put sentiment behaviors and advising behavior in the same dimension. As a consequence, RLPA-ODDS can not perform sentiment behaviors and advising behaviors simultaneously. In the future, if we define more behaviors, then we also need to define a fine-grained action space, so that different kinds of behaviors do not conflict with each other.
- **Limitations from the backbone.** Since RLPA-ODDS is based on a backbone dialogue system, the performance heavily depends on the backbone. In this work, we fine-tuned Blenderbot-400M-distil and keep it fixed, which showed generally good quality. However, it still can not handle well with uncommon user inputs and produces more general responses. An example is in Table 8, where RLPA-ODDS does not realize that the user regards it as his mother. We leave it as a future work to jointly train the policy network and the backbone dialogue system.

Besides the framework limitations, we found DYME has some problems that may affect the training efficiency. For instance, for discrete metrics like “question”, DYME predicts only floating point numbers, which leads to permanent losses between the predicted floating point numbers and the floor and ceiling integer numbers corresponding to the calculated metrics from the generated utterance. Also some metrics like “question” and “utterance length” may lead to lower rewards despite of high quality of some utterances from a human perspective, which has a continuous impact in a conversation, due to the non-sparse nature of DYME-based rewards. These two factors make it difficult for the policy network to achieve the best results. On the other hand, we consider each metric used in DYME equally, which may not be the best solution. In the future, we may use other reward functions like weighted MSE to focus on certain metrics.

## 6 Conclusion

We propose a new open-domain dialogue framework that uses natural language prompts as behavior actions to guide the pre-trained dialogue systems. More specifically, we design an action space with actions being natural language prompts. And we design a reward function using a pre-trained metric model DYME, with which we train a policy network that can select proper actions according to the dialogue context using SAC algorithm.

Even due to limited training resources, our framework still outperforms the baseline in many metrics on both automatic and human evaluation.

The combination of RL and prompting allows AI to proactively guide the conversation in an explainable way. The setup of RL dictates that an agent will not simply imitate humans and follow human examples at every step, but will instead pursue the highest cumulative rewards in advance for the entire conversation, even at the cost of relatively low rewards on some utterances. This is therefore crucial for applications like psychological counseling, neutralizing hate speech or trolling on the Internet, etc.

Since our work is only a preliminary attempt to combine RL with the prompting technique, there are still a lot of possible improvements that could be done in the future. Example directions are 1) Improvements on reward function: In addition to choosing more suitable metrics and reward functions, introducing sparse but evaluative rewards besides rewards on each prediction by DYME, may give the agent more freedom to explore and demonstrate the advantages of RL over supervised learning. 2) Diverse prompts: More prompts related to different sentiments or task-oriented dialogues can not only portray human actions in a more delicate way but may also bridge the gap between open-domain and task-oriented domain. These prompt categories can be determined by clustering conversational data with unsupervised learning. 3) Extended architecture: Jointed models with memory involved may help cope with identity recognition and remembering long conversations. 4) Better space definitions: Reducing the observation space dimension and using unsupervised learning to improve the division of action space is possible, which may accelerate the algorithm convergence. 5) Dynamic aborting: Dynamic "dones" returns based on metrics, especially emotion metrics, can be applied to the RL agent to "stop conversations wisely and in time", which together with evaluative rewards may help it learn from extensive communications with real humans when deployment on servers. 6) Offline-RL: the current trend in RL is Offline-RL, where the agent can make use of expert data (e.g. speech tutorials), data from other models, and even random data for training, where it can take advantage of the various existing available datasets.

## References

- [1] Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. Towards a human-like open-domain chatbot. *arXiv: Computation and Language*, 2020.
- [2] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. Recipes for building an open-domain chatbot. *conference of the european chapter of the association for computational linguistics*, 2021.
- [3] Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, Vinay Adiga, and Erik Cambria. Recent advances in deep learning based dialogue systems: A systematic survey. *arXiv: Computation and Language*, 2021.
- [4] Yann Lecun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. 2022.
- [5] Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Yu Dianhai, Hao Tian, Liu Xuan, and Rui Yan. Multi-view response selection for human-computer conversation. *empirical methods in natural language processing*, 2016.
- [6] Xiangyang Zhou, Lu Li, Daxiang Dong, Liu Yi, Ying Chen, Wayne Xin Zhao, Yu Dianhai, and Hua Wu. Multi-turn response selection for chatbots with deep attention matching network. *meeting of the association for computational linguistics*, 2018.
- [7] Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. Speaker-aware bert for multi-turn response selection in retrieval-based chatbots. *arXiv: Computation and Language*, 2020.
- [8] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv: Computation and Language*, 2019.

- [9] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. *meeting of the association for computational linguistics*, 2019.
- [10] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind W. Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv: Learning*, 2019.
- [11] Abdelrhman Saleh, Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, and Rosalind W. Picard. Hierarchical reinforcement learning for open-domain dialog. *national conference on artificial intelligence*, 2019.
- [12] Can Xu, Wei Wu, and Yu Wu. Towards explainable and controllable open domain dialogue generation with dialogue acts. *arXiv: Computation and Language*, 2018.
- [13] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2019.
- [15] Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. Can you put it all together: Evaluating conversational agents’ ability to blend skills, 2020.
- [16] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. Towards empathetic open-domain conversation models: a new benchmark and dataset. *meeting of the association for computational linguistics*, 2019.
- [17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *international conference on machine learning*, 2018.
- [18] Florian von Unold, Monika Wintergerst, Lenz Belzner, and Georg Groh. Dyme: A dynamic metric for dialog modeling learned from human conversations. *Communications in computer and information science*, 2021.
- [19] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. *International Joint Conference on Natural Language Processing*, 2017.
- [20] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. *meeting of the association for computational linguistics*, 2005.
- [21] Shikib Mehri and Maxine Eskenazi. Unsupervised evaluation of interactive dialog with dialogpt. *annual meeting of the special interest group on discourse and dialogue*, 2020.
- [22] Klaus Krippendorff. Computing krippendorff’s alpha-reliability. 2011.
- [23] Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. Dialogue state tracking with a language model using schema-driven prompting. *empirical methods in natural language processing*, 2021.

## A Definition of Dialogue Acts in [12]

Dialogue Acts	Definitions	Examples
Context Main- tain Statement (CM.S)	A user or a bot aims to maintain the current conversational context (e.g., topic) by giving information, suggesting something, or commenting on the previous utterances, etc.	<b>“there are many good places in Tokyo.”</b> after “I plan to have a tour in Tokyo this summer.”.
Context Main- tain Question (CM.Q)	A user or a bot asks a question in the current context. Questions cover 5W1H and yes-no with various functions such as context clarification, confirmation, knowledge acquisition, and rhetorical questions, etc.	<b>“where are you going to stay in Tokyo?”</b> after “I plan to have a tour in Tokyo this summer.”.
Context Main- tain Answer (CM.A)	A response or an answer to the previous utterances in the current context.	<b>“this summer.”</b> after “when are you going to Tokyo?”.
Context Switch Statement (CS.S)	Similar to CM.S, but the user or the bot tries to switch to a new context (e.g., topic) by bringing in new content.	<b>“I plan to study English this summer.”</b> after “I plan to have a tour in Tokyo this summer.”.
Context Switch Question (CS.Q)	A user or a bot tries to change the context of conversation by asking a question.	<b>“When will your summer vacation start?”</b> after “I plan to have a tour in Tokyo this summer.”.
Context Switch Answer (CS.A)	The utterance not only replies to the previous turn, but also starts a new topic.	<b>“I don’t know because I have to get an A+ in my math exam.”</b> after “when are you going to Tokyo?”.
Others (O)	greetings, thanks, and requests, etc..	<b>“thanks for your help.”</b>

**Figure 6** Definition of Dialogue Acts in [12]. Source: [12]

## B FED Dialogue Qualities

Quality	Importance (%)
Turn-Level	
<b>Interesting</b>	<b>16.15</b>
Engaging	7.46
Specific	9.64
<b>Relevant</b>	<b>18.10</b>
Correct	13.77
Semantically Appropriate	9.90
Understandable	10.70
<b>Fluent</b>	<b>14.27</b>
Dialog-Level	
<b>Coherent</b>	<b>10.95</b>
Error Recovery	9.15
Consistent	7.92
Diverse	10.09
Topic Depth	10.51
<b>Likeable</b>	<b>12.03</b>
<b>Understanding</b>	<b>11.01</b>
Flexible	10.34
Informative	8.00
Inquisitive	9.50

**Figure 7** The screenshot of all dialogue qualities in FED with their relative importance for predicting the overall impression. The most important qualities for turn-level and dialog-level are in bold. Source:[21].

## C Adjacency matrices

	R1	R2	R3	R4	R5	R6	R7
R1	1.00	0.20	<b>0.37</b>	<b>0.33</b>	0.11	0.14	0.16
R2	0.20	1.00	<b>0.36</b>	<b>0.40</b>	0.22	<b>0.30</b>	0.23
R3	<b>0.37</b>	<b>0.36</b>	1.00	<b>0.59</b>	0.23	0.13	<b>0.30</b>
R4	<b>0.33</b>	<b>0.40</b>	<b>0.59</b>	1.00	0.20	0.18	<b>0.31</b>
R5	0.11	0.22	0.23	0.20	1.00	0.12	0.19
R6	0.14	<b>0.30</b>	0.13	0.18	0.12	1.00	0.23
R7	0.16	0.23	<b>0.30</b>	<b>0.31</b>	0.19	0.23	1.00

(a) General

	R1	R2	R3	R4	R5	R6	R7
R1	1.00	<b>0.32</b>	0.29	0.25	0.03	<b>0.39</b>	-0.03
R2	<b>0.32</b>	1.00	<b>0.55</b>	<b>0.42</b>	0.25	<b>0.56</b>	0.23
R3	0.29	<b>0.55</b>	1.00	<b>0.63</b>	0.25	<b>0.36</b>	<b>0.44</b>
R4	0.25	<b>0.42</b>	<b>0.63</b>	1.00	0.24	0.10	<b>0.37</b>
R5	0.03	0.25	0.25	0.24	1.00	0.14	0.11
R6	<b>0.39</b>	<b>0.56</b>	<b>0.36</b>	0.10	0.14	1.00	0.11
R7	-0.03	0.23	<b>0.44</b>	<b>0.37</b>	0.11	0.11	1.00

(b) Empathy

	R1	R2	R3	R4	R5	R6	R7
R1	1.00	0.25	<b>0.51</b>	<b>0.58</b>	0.19	0.13	<b>0.37</b>
R2	0.25	1.00	<b>0.38</b>	<b>0.48</b>	0.16	0.16	<b>0.36</b>
R3	<b>0.51</b>	<b>0.38</b>	1.00	<b>0.66</b>	0.18	0.02	0.12
R4	<b>0.58</b>	<b>0.48</b>	<b>0.66</b>	1.00	0.21	0.25	<b>0.34</b>
R5	0.19	0.16	0.18	0.21	1.00	0.15	-0.01
R6	0.13	0.16	0.02	0.25	0.15	1.00	<b>0.52</b>
R7	<b>0.37</b>	<b>0.36</b>	0.12	<b>0.34</b>	-0.01	<b>0.52</b>	1.00

(c) Relevance

	R1	R2	R3	R4	R5	R6	R7
R1	1.00	-0.04	<b>0.23</b>	0.03	0.04	-0.16	-0.08
R2	-0.04	1.00	0.13	<b>0.25</b>	<b>0.20</b>	0.01	0.06
R3	<b>0.23</b>	0.13	1.00	<b>0.41</b>	<b>0.17</b>	0.02	0.12
R4	0.03	<b>0.25</b>	<b>0.41</b>	1.00	0.13	<b>0.19</b>	0.03
R5	0.04	<b>0.20</b>	<b>0.17</b>	0.13	1.00	-0.08	<b>0.41</b>
R6	-0.16	0.01	0.02	<b>0.19</b>	-0.08	1.00	-0.23
R7	-0.08	0.06	0.12	0.03	<b>0.41</b>	-0.23	1.00

(d) Fluency

**Table 9** Adjacency matrices of Krippendorff's Alpha. **R** indicates raters. R3 and R4 are gender female and European, others are gender male. R5-R7 are volunteers. Agreements over the numbers in Table 4 are in black.