Deep Reinforcement Learning-based Dialogue Policy with Graph Convolutional Q-network

Kai Xu[1], **Zhengyu Wang**[1], **Yuxuan Long**[1], **Qiaona Zhao**[2]
[1]School of software engineering, South China University of Technology, Guangdong, Chian
[2]Research Institute of Radio Management Technology of Shandong Province, Shandong, China
sekxu@mail.scut.edu.cn, wangzy@scut.edu.cn, drlyxdsg@gmail.com, zhaoqiaonana@163.com

Abstract

Deep Reinforcement learning (DRL) has been successfully applied to the dialogue policy of task-oriented dialogue systems. However, one challenge in the existing DRL-based dialogue policy methods is their unstructured state-action representations without the ability to learn the relationship between dialogue states and actions. To alleviate this problem, we propose a graph-structured dialogue policy framework for task-oriented dialogue systems. More specifically, we use an unsupervised approach to construct two different bipartite graphs. Then, we generate the user-related and knowledge-related subgraphs based on the matching dialogue sub-states with bipartite graph nodes. A variant of graph convolutional network is employed to encode dialogue subgraphs. After that, we use a bidirectional gated cycle unit (BGRU) and self-attention mechanism to obtain the high-level historical state representations and employ a neural network for the high-level current state representations. The two state representations are joined to learn the action value of dialogue policy. Experiments implemented with different DRL algorithms demonstrate that the proposed framework significantly improves the effectiveness and stability of dialogue policies.

Keywords: dialogue policy, deep reinforcement learning, task-oriented dialogue systems

1. Introduction

Task-oriented dialogue systems aim to assist users to complete one or more specific tasks, such as movie ticket booking and restaurant reservation. The pipeline task-oriented dialogue system is one well-known fashion that consists of four components (Kwan et al., 2023; Lubis et al., 2020): natural language understanding, dialogue state tracking, dialogue policy (DP) and natural language generation. The DP selects the next system action based on the current dialogue state, which determines the entire dialogue flow (Peng et al., 2017; Kwan et al., 2023). We focus on dialogue policy learning (DPL).

DPL is mainly implemented by deep reinforcement learning (DRL), which integrates deep learning and reinforcement learning. In reinforcement learning, the dialogue agent takes an action to interact with the environment and receives a dialogue reward. While deep learning maps the dialogue state into feature engineering (Prudencio et al., 2023) to capture the important feature information. Traditional dialogue policies, such as rule-based (Varges et al., 2009) and supervised learning-based methods (Kim et al., 2014), are inefficient and lack robustness. DRL-based dialogue policies are better for maintaining large-scale dialogue state spaces and being robust to noise.

However, the task-oriented dialogue policy based on DRL remains some limitations. (i) The common weakness of current DRL-based methods for DPL is that they discretely represent the states/actions (Zhang et al., 2021; Tian et al., 2022; Huang and Cao, 2023), such that the underlying relationships (e.g., behavioral similarities) between different states (or sub-states) are not effectively explored and exploited. (ii) DRL-based dialog policies conducted by trial-and-error often require a high number of iterations to explore valuable state information (Zhang et al., 2022), and accelerating the learning of dialog policies through expert experiences (Jeon and Lee, 2022; Wang et al., 2020) is both a financially costly and laborious task. Furthermore, methods combine with supervised learning are often not available for a very large number of state spaces. (iii) Some traditional DRL-based dialogue policy methods (Chang et al., 2017; Li et al., 2020) are not sample-efficient. The dialogue agent training with limited samples does not achieve a satisfactory performance, and it is also expensive and time-consuming to obtain data by interacting with real users. The above limitations significantly impede the performance of existing methods to learn effective dialogue policy.

To overcome these challenges, we propose a universal DPL framework implemented via DRL with a graph-structured dialogue state. The framework is based on a **G**raph convolutional Q-network to learn graph-**S**tructured information for modeling **D**ialogue **P**olicy (GSDP). The Q-network is a deep neural network that is used to approximate the action-value function, also known as the Q-function. First, the user belief state, is the information extracted from the user's utterance, as a user-related graph node

LREC-COLING 2024, pages 4555–4565
20-25 May, 2024.

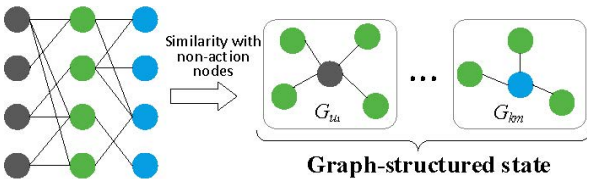


Figure 1: A toy example to illustrate how to build a graph-structured state. The gray circle ● is the user-related node U. The green circle ● is the action node. The blue circle ● is the knowledge-related node K. The left part is the raw user-related bipartite graph G_u and knowledge-related graph G_k. The right part is building a graph-structured state $s = \{G_{u_1}, ..., G_{k_j}\}$ based on a similarity with non-action nodes.

U. The corresponding action of the user belief state is another graph node. The two kinds of nodes are linked to build a user-related bipartite graph G_u. The user belief state is also used to query the database of the task-oriented dialogue system to match user requests. Then the database query results, and user belief state are combined to build another knowledge-related graph node K, with its action nodes to build a knowledge-related bipartite graph G_k. The user belief state and database query results are encoded following the existing works (Peng et al., 2018; Li et al., 2017a). In a single dialogue turn, we combine the current user belief state, and database query results to match them with the nodes of U and K, and obtain the potential actions, and generate the subgraphs of the dialogue state. Finally, the subgraphs are embedded for calculating the Q-values of agent actions. Figure 1 makes a toy example to illustrate how to build a graph-structured dialogue state for DPL. Our contributions are summarized as follows.

- This paper proposed a generic framework for DPL that predicts action values through bipartite graphs. The bipartite graphs are built via successful dialog trajectories from baseline reinforcement learning algorithms and do not require expert experience.
- We use similarity to generate dialogue subgraphs and construct a variant of graph neural network to extract information from different subgraphs and output graph-aware embeddings. The BGRU (Chung, 2014; Morchid, 2022) and self-attention mechanism (Lin et al., 2017) are employed to process the graph-aware embeddings for more dialogue features.
- Our experimental evaluations show that the GSDP framework is more robust, more efficient, and has high scalability compared to different DRL algorithms, performing excellently in the dataset of movie ticket booking.

2. Related work

With the flourishing of deep learning, deep reinforcement learning (DRL), which combines deep neural networks and RL, is also extensively used to dialogue policy. Cuayáhuitl et al. (2015) implemented DRL in the negotiation domain, their experimental results demonstrate the DRL performed significantly better than other heuristic or supervised approaches. Zhao et al. (2021) incorporated episodic memory policy and DQN policy to learn dialogue policies. Their episodic memory policy stores a large number of experiences and does not guarantee efficient storage. Tian et al. (2022) estimate the Q-value using the partial average between the predicted maximum Q-value and the minimum Q-value. A balance weight is computed by heuristics or neural networks to alleviate the overestimation Q-value problem of DQN (Tian et al., 2022). However, the heuristic algorithm can reduce the efficiency of training. The Deep Dyna-Q (DDQ) (Peng et al., 2018) is a model-based method that incorporates the world model to generate simulated experiences. Thus, both real experience and simulated experience are used to train dialogue policy. Other DDQ-based DPLs can reference (Zhang and Shinozaki, 2022; Huang and Cao, 2023).

Graph-based RL algorithms had been successfully applied in different domains, such as recommendation systems (Lei et al., 2020), epidemic control (Yang; et al., 2021), city services (Gammelli et al., 2022), etc. However, there are relatively few studies are graph-based dialogue policy. Tu and Wang (2022) proposed a hierarchical information-aware conversation recommender for conversation recommendation systems, they constructed a hierarchical relationship graph by customer, item, and attributes to enrich the item and attribute representation based on similar users. Then the DQN is used to predict items. However, they are not available in the dialogue policy domain. Chen et al. (2019) proposed a multi-agent graph dialogue policy model, where each graph node corresponds to an agent, the graph contains slot-independent nodes and slot-dependent nodes, the slot-dependent nodes consist of information slots, and slot-independent nodes indicate database search results. Their graph is constructed based on information slots, which is an isomorphic graph, lacking consideration of the relationship between nodes and potential actions.

We propose to construct bipartite graphs for dialogue policy learning. This work is the first attempt to consider the relationship between dialogue states and potential actions. We implement different RL methods based on the proposed framework to demonstrate its excellent performance and robustness.

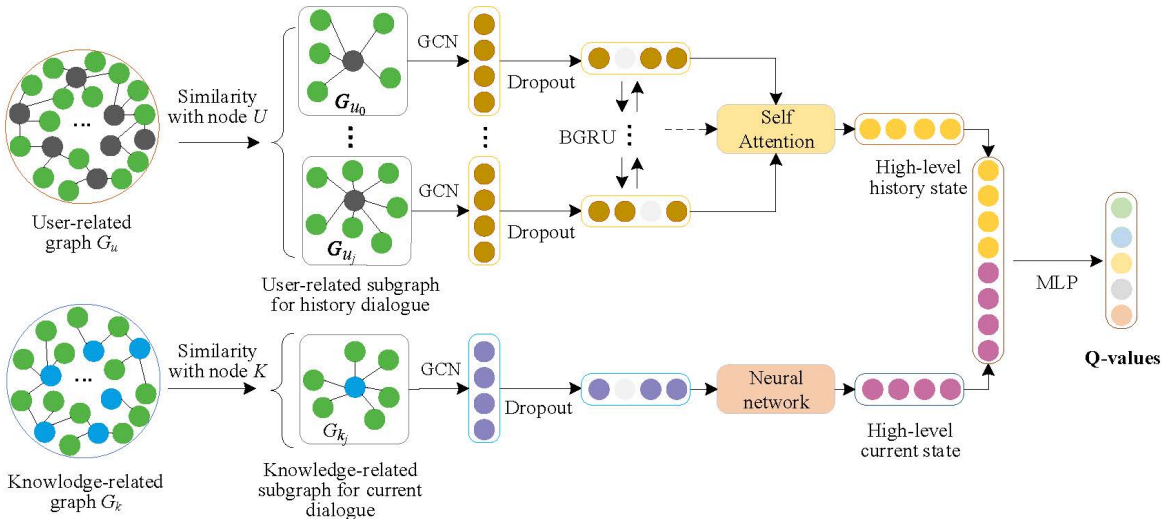


Figure 2: Illumination of proposed dialogue policy framework GSDP. It is defined by two modules. The top module receives the graph-structured history dialogue information which is represented by subgraph G_{u_i}, and then encoded to a high-level history state. The bottom module receives one knowledge-related subgraph which is represented by a G_{k_j}, and then encoded to a high-level current state.

3. Method

In this section, we introduce our proposed GSDP framework for dialogue policy learning. First, describe the structure of the proposed GSDP in general. Then, we show the details of the framework implementation of different modules. The training and pseudo-code of the GSDP model is described in the end.

3.1. Overall Architecture

The proposed GSDP is based on the structured dialogue state to learn dialogue policy. We build a user-related bipartite graph via user belief states and dialogue actions, which are user utterance-related. A knowledge-related bipartite graph is built with added features of database query results (see Figure 1 for details). Then we generate subgraphs via the current dialogue state, and a graph convolutional network (GCN) is employed to learn the graph-structured state representation of subgraphs. The GSDP can grasp the relationship between user utterances and actions and improves the training efficiency of dialogue policy because it mines the graph-structured features.

Figure 2 depicts the architecture of the GSDP framework which includes two modules. The top module encodes a high-level historical state representation based on the subgraph G_{u_i}. The bottom module encodes a high-level current state representation based on subgraph G_{k_j}. Specifically, different G_{u_i} and G_{k_j} are encoded into different subgraph embeddings using the GCN. Then, a dropout layer is employed to weaken the embedding of the subgraph that generates via similarities with non-action nodes. The bidirectional gated cycle unit (BGRU) is used to extract the temporal information of sequences after the dropout layer. The output of BGRU at each moment is incorporated into a high-level historical state representation by a self-attention mechanism. Meanwhile, A high-level current state representation is generated using a neural network following the dropout layer after GCN learning of G_{k_j}. The neural network is implemented as same as the deep Q-network (DQN) (Peng et al., 2018) and deep recurrent Q-network (DRQN) (Zhao and Eskenazi, 2016). We will demonstrate the performance of the different implementation results in the experimental part. Finally, the high-level historical state and the high-level current state are stitched together as the input of a multilayer perceptron to predict the Q-values. In the following, we present the details of the proposed GSDP framework.

3.2. Implementation of GSDP

Dialogue state representation: In DPL, the user utterance corresponding to the belief state can be converted into three parts: user intent, information slot and request slot. The embedding vector of user intent is $E_c \in \mathbb{R}^{d_c}$, the information slot embedding vector $E_i \in \mathbb{R}^{d_i}$ and request slot embedding vector $E_r \in \mathbb{R}^{d_r}$. E_c, E_i, and E_r are the one-hot encoding reference papers (Peng et al., 2018; Li et al., 2017a), which can retain the raw feature information. The user-related node embedding is the $E_u = [E_c, E_i, E_r]$, where $[\cdot, \cdot, \cdot]$ is the concatenation operation. The embedding of database query results is the multi-hot encoding $E_d \in \mathbb{R}^{d_r+1}$, which satisfies the request slots set to 1, otherwise to

0. The $+1$ is an encoding position to determine whether all the request slots are satisfied. In the graph G_k, the knowledge-related node embedding is $E_k = [E_c, E_i, E_r, E_d]$.

Subgraph: We build the bipartite graphs G_u and G_k based on a baseline RL model which can be DQN, it does not rely on any expert experience. Thus the action corresponding to the state may be in the successful episodes or the failed episodes. We select the state-action pairs with a higher success probability to build the bipartite graphs G_u and G_k. E_u is the embedding of node U, E_k is the embedding of node K. However, we do not build a large-scale bipartite graph even though the embedding sets of the E_u and E_k are enormous. Thus, we merely build small-scale bipartite graphs for convenience of storage. In experiment section, we demonstrate that even using small-scale graph data can improve the effectiveness of DPL.

The subgraphs are generated based on the G_u and G_k. In a single dialogue turn, a user current belief state representation is $s^u_{u_i}$, and the current knowledge-related dialogue representation $s^u_{k_j}$. The $s^u_{u_i}$ and $s^u_{k_j}$ are matched via cosine similarity with the nodes U and K, and obtain the action nodes to generate subgraph G_{u_i} and G_{k_j}. The action representation corresponding to G_{u_i} and G_{k_j} are obtained are obtained via weighted similarity, respectively.

$$s^a_{u_i} = \sum_{n=1}^{N} sim_n(s^u_{u_i}, E_u)E^a_u \tag{1}$$

$$s^a_{k_j} = \sum_{n=1}^{N} sim_n(s^u_{k_j}, E_k)E^a_k \tag{2}$$

where the sim_n is the function to calculate the similarity, N is the number of the most similar ones. E^a_u is representation of the action nodes corresponding to node U (E_u is the node embeddings), and E^a_k is the representation of action nodes corresponding to node K (E_k is the node embedding). In the light of above, the subgraph $G_{u_i} \sim (s^u_{u_i}, s^a_{u_i})$ and the subgraph $G_{k_j} \sim (s^u_{k_j}, s^a_{k_j})$.

GCN layer: We construct a variant of the GCN model based on (Lei et al., 2020; Hamilton et al., 2017) to encode the graph-structured states. For each subgraph G_{u_i} or subgraph G_{k_j}, the main goal of variant GCN is to produce a graph-aware representation. Notice that the E_u and E_k are replaced by $s^u_{u_i}$ and $s^u_{k_j}$ during dialog interactions. We use x_{u_i} to represent the G_{u_i}, and x_{k_j} to represent the G_{k_j},

$$x_{u_i} \leftarrow relu(W_{u_i}[s^u_{u_i}, s^a_{u_i}] + b_{u_i}) \tag{3}$$

$$x_{k_j} \leftarrow relu(W_{k_j}[s^u_{k_j}, s^a_{k_j}] + b_{k_j}) \tag{4}$$

where W_{u_i} and W_{k_j} are the trainable weights, b_{u_i} and b_{k_j} are the trainable biases. We can obtain the embedding of each subgraph G_{u_i} and G_{k_j} via the above operation (3) and (4). For the dialogue sequence of history, the graph-aware representation of $\{G_{u_1}, ..., G_{u_{j-1}}\}$ is the $\{x_{u_1}, ..., x_{u_{j-1}}\}$. For the current dialogue state, the x_{k_j} is the representation of G_{k_j}.

Dropout layer: We add the dropout layer after the GCN layer because we have used similarity to obtain the latent actions, and the build of bipartite graphs does not come from real expert experience. In the dropout layer, we obtain the x_{u_i} and x_{k_j}, where

$$x_{u_i} \leftarrow dropout(x_{u_i}), \;\; x_{k_j} \leftarrow dropout(x_{k_j}) \tag{5}$$

BGRU Layer: The dialogue history is described by $\{x_{u_1}, ..., x_{u_{j-1}}\}$ as a temporal sequence. We leverage the BGRU to process $\{x_{u_1}, ..., x_{u_{j-1}}\}$, Considering the dialogue interaction behavior of the user is reversible, i.e., the pro and con order (bidirectional) of user's requests can also complete a task-oriented dialogue. We take advantage of BGRU because it has fewer parameters than long short term memory (LSTM) and has achieved advanced effects in low-complexity sequences (Colombo et al., 2020; Cahuantzi et al., 2023). The GRU has shown advantages over LSTM and simple recurrent neural networks in many dialogue tasks (Colombo et al., 2020; Cahuantzi et al., 2023). The goal of the BGRU is to process the $\{x_{u_1}, ..., x_{u_{j-1}}\}$ to the sequence of hidden vectors $\{h_{u_1}, ..., h_{u_{j-1}}\}$, the h_{u_i} is computed as

$$\begin{aligned} h_{u_i} &\leftarrow BGRU(x_{u_i}, h_{u_{i-1}}) \\ &= GRU(x_{u_i}, \overrightarrow{h}_{u_{i-1}}) \circ GRU(x_{u_i}, \overleftarrow{h}_{u_{i+1}}) \end{aligned} \tag{6}$$

where $\overrightarrow{h}_{u_{i-1}}$ and $\overleftarrow{h}_{u_{i+1}}$ are the forward and backward hidden vectors, respectively. $\circ$ is the element-wise sum. Considering that different h_{u_i} have different importance for policy learning, a self-attention mechanism (Lin et al., 2017) is constructed to distinguish its importance. Thus the weighted history state is represented as:

$$s_u = \sum_{i=0}^{j-1} \beta_i h_{u_i} \tag{7}$$

where j is the length of dialogue history, β_i is the self-attention score, which is computed by:

$$\beta_i \leftarrow \frac{exp(w_h^T tanh(W_h h_{u_i}))}{\sum_{l=0}^{j-1} exp(w_h^T tanh(W_h h_{u_l}))} \tag{8}$$

where w_h^T and W_h are the trainable weights in the self-attention. The important features can be autonomously selected from the input sequence with the attention mechanism, and help the dialogue agent capture the user's dynamic interaction preference at each turn step. A high-level historical

dialogue state is used as a portion of the final dialogue state embedding to predict the Q-values.

Neural network layer: The main goal of the neural network layer is to process the embedding of the current dialogue state. Traditional unstructured dialogue policy directly processes the dialogue state which is embedded via a multi-hot encoder, and then either a deep neural network (Wang et al., 2020) or LSTM (Zhao and Eskenazi, 2016) is employed to output the Q-values. I.e., they do not use the graph encoding that we implemented above. In this module, we reprocess the x_{k_j} obtained from graph encoding, constructing a deep neural network to map the high-level current state. $s_j \leftarrow NN(x_{k_j})$, where the NN represents the deep neural network or typical LSTM network. In the implementation, The NN is selected as the same as the neural networks of baseline reinforcement learning.

Output Layers: We stitch together the high-level historical state and the high-level current state to obtain the final state representation $s = [s_h, s_j]$, the Q-value is

$$Q(s, a) = f([s_h, s_j], a) \tag{9}$$

where j is the current state index. f is a mapping function. It is worth mentioning that different algorithms operate on s differently. For example, the DQN algorithm uses a multilayer perceptron (MLP) to map s to its corresponding Q-value. However, the Dueling network constructs both an advantage function and a Q-value function, which are combined to determine the optimal action to take. Thus f can be either MLP or operation via Dueling network. Our model is able to use different mapping operations.

3.3. Model Training

This part describes the model training based on DQN. For notational convenience, we use s_t to denote the raw dialogue state at t-turn dialogue, which includes the dialogue history and database query results, i.e. $s^u_{u_i} \in s_t$, where $i \in [0, t)$, $s^u_{k_j} \in s_t$, where $j = t$ .This means that the raw dialogue state s_t will be coded to $s^u_{u_i}$ and $s^u_{k_j}$. The loss function based on DQN is defined as follows.

$$\begin{aligned} L(\theta) &= \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim D}[(y - Q(s_t, a_t, G_u, G_k; \theta))] \\ &\quad + \lambda KL[p(a|\pi_\theta(s_t)) \,||\, q(a|s_t, G_u, G_k)] \\ &= L_D(\theta) + \lambda L_{KL}(p, q) \end{aligned} \tag{10}$$

where $L_D(\theta)$ is the traditional DQN loss function, $L_{KL}(p, q)$ is the Kullback-Leibler (KL) divergence loss of distribution p and distribution q. $p(a|\pi_\theta(s_t)$ is the action distribution of s_t under policy π_θ, $q(a|s_t, G_u, G_k)$ is corresponding mixed latent action distribution of s_t under graphs G_u and G_k. λ is a multiplier to control the disentanglement of latent dialogue actions. We add hyperparameter λ as a reference (Wang et al., 2021). More loss functions can also be used for training. For example, loss functions based on Double DQN or Dueling networks to replace the $L_D(\theta)$. Moreover, the E^a_u and E^a_k (see Eq.(1) and Eq.(2)) are latent normalized action representations that provide an initialized distribution of latent actions. Thus, we make a slight variation of the traditional ϵ-greedy policy. The variant of ϵ-greedy policy is

$$a_t = \begin{cases} arg\max_a Q(s_t, a), & \text{with probability } 1-\epsilon \\ select\ an\ action\ based\ on\ probability\ 1-q, & \text{otherwise} \end{cases} \tag{11}$$

Our model greedily selects (exploitation) the action corresponding to the largest Q-value with the probability $1 - \epsilon$, otherwise, the action is selected (exploration) with probability $1 - q$. Algorithm 1 presents the training processing of GSDP.

Algorithm 1 Training dialogue agent with GSDP

```
Initialize replay memory D, user-related graph G_u, knowledge-related graph G_k, ε-greedy exploration value
for each episode do
    Initialize raw state s_1
    for t = 1 to j do
        Obtain the user-related status encoding s_u^{u_t} and the knowledge-related encoding s_k^{u_t} according to s_t
        Obtain the action encodings s_u^{a_t} and s_k^{a_t} according to Eq.(1) and Eq.(2)
        Based on the similarity of s_u^{u_t}, s_k^{u_t} to nodes U and K, obtain the latent action probability distribution q_t
        Select an action a_t according to Eq.(11)
        Execute action a_t, receive environment rewards r_t and come into next raw state s_{t+1}
        Store (s_t, a_t, r_t, s_{t+1}) to D
        set s_t = s_{t+1}
    end for
    Sample random minibatch of (s_t, a_t, r_t, s_{t+1}) from D
    Execute a gradient descent step via Eq. (10)
    Replace target parameter θ^- ← θ after every L iterations
end for
```

4. Experiments

This section describes the performance evaluation of GSDP. The experiments are conducted on

the task-oriented movie-ticket booking (Li et al., 2017a) dataset. It contains 11 user intents, 16 user slots, and 128 user goals. This dataset is applied in recent studies (Zhao et al., 2021; Huang and Cao, 2023). We use a user simulator (Li et al., 2017b), which is frequently used in the study of dialogue policies (Zhao et al., 2020; Zhang et al., 2021; Tian et al., 2022). The evaluation metrics include dialogue success rate (*Suc*), averaged reward (*Rew*), and the number of averaged dialogue turns (*Tur*). *Suc* indicates the probability that the RL agent helps the user to complete all user goals in a limited number of dialogue turns. *Rew* is the mean of all rewards obtained by the simulated user while interacting with the dialogue agent. $Rew = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} Q(s_j, a)$, where N is the number of dialogue trajectories, and M is the length of current dialogue trajectory. Note that different dialogue trajectories may have different dialogue lengths. That is, M is not always the same. *Tur* is the average number of dialogue turns before the simulated user completes a task.

4.1. Baselines

We compared performance of GSDP with other Q-network based methods, including:

DQN (Mnih et al., 2015): A classical reinforcement learning algorithm was first proposed in the field of games, which has become a fundamental model to train dialogue policy (Wang et al., 2020).

DDQN (Xiao et al., 2016): Double DQN (DDQN) is used to deal with the Q-value overestimation of DQN that selects the maximum action in each exploitation.

DRQN (Zhao and Eskenazi, 2016): DRQN is an extension to DQN which introduces a LSTM network. DRQN performs significantly better than DQN when an agent only observes partial states in dialogue policy learning.

Dueling (Wang et al., 2016): Dueling DQN factorizes the Q-value into the sum of the advantage function and Q-function, which changes the structure of DQN.

MAXMIN (Lan et al., 2020): MAXMIN Q-learning uses the minimum of multiple maximums from different ensemble units to estimate the ground truth maximal Q-value.

DPAV (Tian et al., 2022): DPAV uses a dynamic partial average estimator to calculate the ground truth maximum action value. A partial average weight is used to balance the predicted maximum action value and minimum action value.

These models are used as baseline methods and their inputs are the structureless dialogue states. We scale them to verify the performance of the proposed GSDP. Specifically, deep reinforcement learning contains two modules, deep learning and reinforcement learning. The GSDP framework mainly modifies the deep learning part of baselines. We obtain a high-level historical state representation and a high-level current state representation by encoding the original dialogue states through bipartite graphs. The neural network module used to obtain the high-level current state representation follows the deep learning approach used in the baseline models. In the reinforcement learning part, the GSDP framework is implemented consistent with the baseline models, but the loss function of the reinforcement learning part adds a KL regularization term.

4.2. Implementation Details

All methods are implemented with the PyTorch toolkit. In the GSDP, the BGRU takes one hidden layer of size 40, and the NN takes one hidden layer of size 80. Other methods include one hidden layer of size 80 and a ReLU activation function. All methods are warm started using a rule-based policy before training, with a hyperparameter of 200 episodes. The size of the experience replay pool is set to 5000, and the discount factor for future reward is $\gamma = 0.95$. The optimizer use $RMSprop$ with a learning rate of 0.001 and a batch size of 16. The maximum number of dialogue turns is $L = 40$. The agent is rewarded with $2L$ when the user all goals are completed, $-L$ when it fails, and -1 at other turns. The MAXMIN DQN uses 5 DQNs for selecting the minimum Q-value in the maximum Q-values of different DQNs. The weight of DPAV balancing the maximum and minimum Q-values is obtained by a heuristic algorithm, and the best performance on our machine corresponds to a weight of 0.5. The value of λ in the loss function of GSDP is set to 1, and its bipartite graph G_u contains 2100 nodes of U and the bipartite graph G_k contains 170 nodes of K. The other parameters of all models are kept consistent for a fair evaluation.

4.3. Performance of GSDP with different top-N similarity

For each turn dialogue, the raw state s_t is encoded into user-related and knowledge-related representations. Thus their latent actions need to be obtained based on top-N similarity to non-action nodes in the bipartite graphs (see Eq.(1) and Eq.(2)). Considering different top-N similarities have different performances to GSDP. We implement the GSDP with different baseline DRL algorithms, and the *Suc* as a metric to verify the performance under hyperparameter N. Table 1 displays the experimental results. The GSDP(DQN) obtained the highest dialog success rate with Top-2 similarity, and its performance is better compared with the other models, because GSDP (DQN) has more reliable actions with top-2 similarity of bipartite graph non-action nodes.

	top-1	top-2	top-3	top-4	top-5	top-6	top-7	top-8	top-9	top-10
GSDP(DQN)	0.661 ± 0.042	**0.727 ± 0.038**	0.682 ± 0.041	0.670 ± 0.053	0.683 ± 0.040	0.651 ± 0.044	0.661 ± 0.042	0.678 ± 0.043	0.662 ± 0.056	0.664 ± 0.053
GSDP(DDQN)	0.644 ± 0.034	0.659 ± 0.031	**0.694 ± 0.030**	0.675 ± 0.034	0.683 ± 0.053	0.657 ± 0.029	0.670 ± 0.038	0.668 ± 0.043	0.662 ± 0.048	0.684 ± 0.051
GSDP(DRQN)	0.640 ± 0.034	0.688 ± 0.022	0.683 ± 0.032	0.679 ± 0.028	0.687 ± 0.044	0.651 ± 0.036	0.686 ± 0.026	0.67 ± 0.039	0.643 ± 0.041	**0.704 ± 0.035**
GSDP(Dueling)	0.635 ± 0.035	0.702 ± 0.029	0.689 ± 0.031	0.699 ± 0.027	0.692 ± 0.032	0.669 ± 0.033	0.668 ± 0.038	**0.703 ± 0.036**	0.692 ± 0.039	0.683 ± 0.045
GSDP(MAXMIN)	0.648 ± 0.027	0.602 ± 0.031	0.630 ± 0.055	**0.700 ± 0.042**	0.614 ± 0.059	0.614 ± 0.059	0.595 ± 0.036	0.606 ± 0.037	0.620 ± 0.042	0.604 ± 0.048
GSDP(DPAV)	0.665 ± 0.040	0.632 ± 0.040	**0.690 ± 0.033**	0.644 ± 0.034	0.668 ± 0.033	0.653 ± 0.033	0.653 ± 0.035	0.667 ± 0.033	0.661 ± 0.028	0.623 ± 0.038

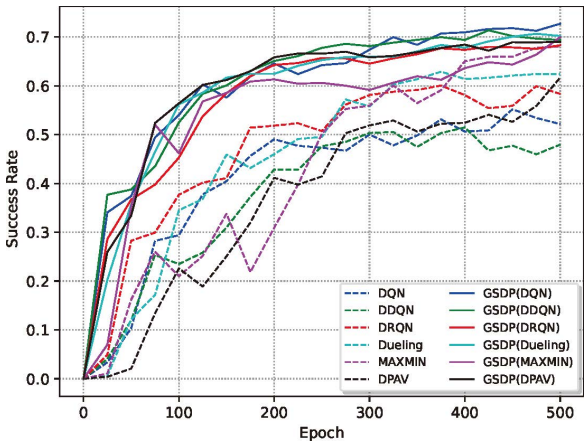
Table 1: The dialogue success rate of GSDP with Top-N similarity under different models. The bolded font indicates the best results and the underline indicates the second best results.

GSDP(DDQN) achieves the best results with Top-3 similarity. GSDP(DRQN) achieves the best results with Top-10 similarity and second-best results with Top-2 similarity. Besides, its results slightly fluctuate from Top-3 to Top-5. GSDP(Dueling) had the best results with top-8 similarity and the second-best results with Top-2 similarity, but the two results are close to each other, the result of op-2 similarity with a smaller variance. GSDP(MAXMIN) significantly fluctuates under different top-N similarities, and it achieves the best results with Top-4 similarity. The GSDP(DPAV) results are less fluctuating than the GSDP(MAXMIN) as it updates the Q function with a balance parameter. GSDP(DPAV) achieves optimal results with the Top-3 similarity. Different top-N achieve different dialogue success rates because the action distributions and high-dimensional state representation are different with distinct top-N. In the following section, we select the appropriate Top-N similarity to implement our model and compare it with the baseline models.

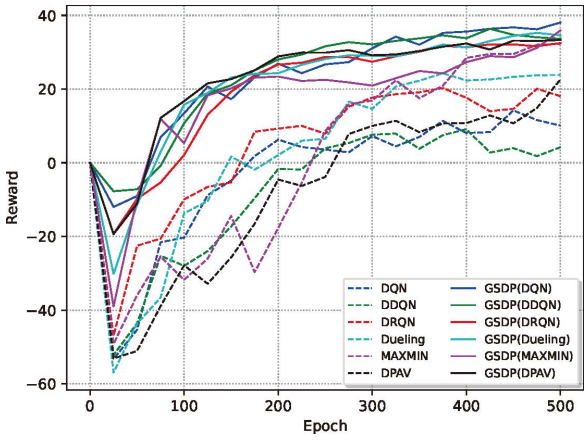
4.4. Main Results

The main results are reported in Figure 3 and Table 2. We use *Suc*, *Rew*, and *Tur* to evaluate the performance of dialog policy learning. According to the experimental results in section 3.2, we implement the GSDP(DQN), GSDP(DDQN), GSDP(DRQN), GSDP(Dueling), GSDP(MAXMIN), and GSDP(DPAV) with top-2, top-3, top-2, top-2, top-4 and top-3 similarities. These top-N values are chosen based on the *Suc* and its variance.

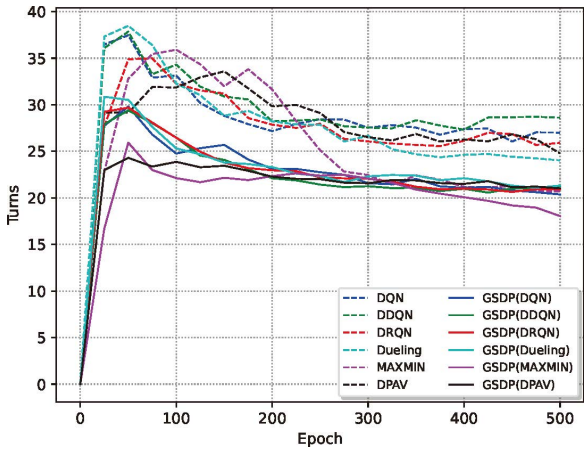
Figure 3(a) shows results of *Suc* with different Q-networks in 500 simulations. The proposed GSDP framework achieves better performance and the tendency of the Suc increase faster in the earlier simulations. The proposed framework is nearly converged at 200 epochs demonstrating its faster convergence rate. In the last epoch of the simulation, GSDP significantly outperforms the baseline models. Specifically, see Table 2 which shows that the GSDP(DQN) algorithm achieves the best performance with a Suc 72.7%. The performance of the GSDP(MAXMIN) and baseline MAXMIN algorithms are comparable, which remains a slight performance improvement. The DDQN algorithm



(a) Dialogue success rate



(b) Dialogue average reward



(c) Dialogue average turns

Figure 3: Performance demonstration of deep reinforcement learning-based graph convolutional Q-networks for dialogue policies.The X-axis is the number of training epochs and the Y-axis is the dialogue success rate.

has the most significant performance improvement (21.4%) compared with GSDP(DDQN), which fully demonstrates the superiority of our GSDP framework. Furthermore, according to the results in Table 2, the variance of experimental results of our model implemented on different Q-networks are all smaller than that of the traditional algorithms, which fully demonstrates the stability of the GSDP.

Figure 3(b) illustrates the Rew of different Q-networks over 500 simulations. It demonstrates the Rew maintains decreasing and then increasing, which is due to the sparse reward at the beginning of the dialogue. The sample data gradually increases with the agent and user interaction, while positive rewards samples are also increased, resulting in a gradual increase in dialogue rewards. Table 2 displays the detailed experimental results, in which the proposed GSDP and baseline models have a large gap with the metric of Rew at the 200th epoch. Where MAXMIN and GSDP(MAXMIN) had the largest average reward gap of 41.13. The smallest gap was between DRQN and GSDP(DRQN), which is 17.37 at the 200th epoch. At the end of the epoch, the proposed GSDP maintains a significant advantage over the baseline models, where GSDP(DQN) achieves the highest reward of 38.08.

Figure 3(c) demonstrates the results of Tur of different Q-networks. The smaller the number of dialogue turns, the less dialogue is required for the dialogue agent to complete a task, indicating the better performance of the dialogue agent. Figure 3(c) and Figure 3(b) have an inverted tendency in general, showing that the proposed GSDP framework can improve the effectiveness of different baseline Q-networks. In particular, refer to Table 2, GSDP(DDQ) reduces the average number of dialogue turns by 6.6 compared to the DDQ algorithm. Furthermore, Table 2 also displays that GSDP has lower variance in both the Tur and the Rew, which demonstrates the stability of GSDP.

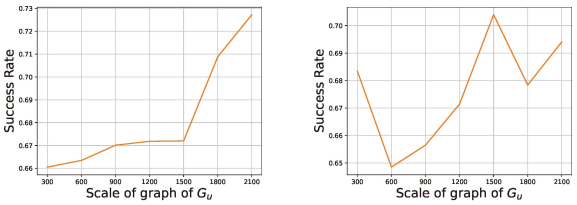
Based on the results in Figure 3 and Table 2, we conclude that the GSDP framework demonstrates superior performance when implemented on the six Q-networks. Specifically, (i) it has more efficient sampling performance: On the one hand thanks to our modification of the traditional ϵ-greedy exploration fashion (see Eq.(11)), and on the other hand owing to our incorporation of the latent action distribution into the loss function (see Eq.(10)). These two components work together to make our proposed framework more sample-efficient. (ii) It has excellent performance improvement. We suggest that the use of bipartite graphs to model dialogue policy learning contributes to the observed results, as this approach enables a more effective exploration of past interaction patterns and the relationship between user utterances and latent actions. In addition, our framework does not use expert experience modeling bipartite graphs. If we use the expert experience to build the bipartite graphs, the performance of the dialogue policy is expected to be further improved.

4.5. Sub-graphs with different Sizes

We verify the influence of the size of the subgraph to the performance of proposed GSDP framework. We only show the relationship between different sizes of G_u and dialogue success rate, ignoring the relationship between dialogue success rate and G_k, because the number of subgraphs of G_k is 170, which is already relatively small. Figure 4 shows the test results, which demonstrate that there is a positive correlation between G_u graph size and dialogue success rate. The larger of the size G_u have a higher dialogue success rate in general. The results of the dialogue success rate fluctuates with the sizes of G_u because the state encoding is based on the similarity to non-action nodes of G_u, whereas the experiments are randomly selected subgraphs of G_u to calculate the similarity. In most cases, the implementations with smaller size Gu have been able to achieve decent dialogue success rates that exceed their corresponding baseline models. This is because we use the graph G_u as a master, and then construct structured states based on similarity. The experiments verify that even small-scale graphs G_u can effectively facilitate dialogue policy learning.

Models	Epoch 200			Epoch 500		
	Suc	Rew	Tur	Suc	Rew	Tur
DQN	0.491 ± 0.121	6.28 ± 16.02	27.17 ± 3.23	0.522 ± 0.089	10.09 ± 11.79	27.01 ± 2.40
DDQN	0.428 ± 0.164	-1.69 ± 21.23	28.22 ± 3.37	0.480 ± 0.094	4.24 ± 12.44	28.61 ± 2.51
DRQN	0.518 ± 0.099	9.28 ± 12.80	27.86 ± 2.29	0.584 ± 0.077	18.07 ± 10.20	25.90 ± 2.15
Dueling	0.460 ± 0.112	2.09 ± 14.82	28.22 ± 3.05	0.624 ± 0.067	23.85 ± 8.97	24.03 ± 2.01
MAXMIN	0.309 ± 0.138	-17.71 ± 17.93	31.65 ± 3.05	0.680 ± 0.054	32.28 ± 7.10	20.72 ± 1.56
DPAV	0.411 ± 0.115	-4.54 ± 14.57	29.84 ± 1.93	0.617 ± 0.077	22.63 ± 10.21	24.78 ± 2.19
GSDP(DQN)	**0.646 ± 0.031**	**26.91 ± 4.16**	**23.12 ± 1.07**	**0.727 ± 0.038**	**38.08 ± 6.17**	**20.37 ± 1.42**
GSDP(DDQN)	**0.651 ± 0.062**	**28.05 ± 8.04**	**22.11 ± 1.99**	**0.694 ± 0.030**	**33.65 ± 7.94**	**21.28 ± 1.41**
GSDP(DRQN)	**0.643 ± 0.056**	**26.65 ± 7.26**	**22.96 ± 1.83**	**0.688 ± 0.022**	**32.50 ± 7.47**	**20.94 ± 1.66**
GSDP(Dueling)	**0.591 ± 0.116**	**24.30 ± 7.98**	**23.31 ± 1.69**	**0.702 ± 0.029**	**34.59 ± 7.64**	**21.32 ± 1.88**
GSDP(MAXMIN)	**0.613 ± 0.023**	**23.42 ± 3.15**	**22.34 ± 1.21**	**0.700 ± 0.042**	**35.93 ± 5.32**	**18.05 ± 0.94**
GSDP(DPAV)	**0.658 ± 0.056**	**28.87 ± 7.59**	**22.28 ± 2.03**	**0.690 ± 0.033**	**33.34 ± 6.02**	**21.02 ± 1.43**

Table 2: Display of detailed experimental results. The bolded font indicates better results than its baseline model.



(a) GSDP(DQN) (b) GSDP(DDQN)

(c) GSDP(DRQN) (d) GSDP(Dueling)

(e) GSDP(MAXMIN) (f) GSDP(DPAV)

Figure 4: The dialogue success rate of GSDP with different Sizes of G_u.

5. Conclusion

In this paper, we propose a graph-structured framework for learning dialogue policy. At first, user-related and knowledge-related bipartite graphs are built using baseline RL. Then, we use these bipartite graphs to generate two kinds of subgraphs which are dialogue history based and current dialogue based. The different subgraphs are encoded to obtain the high-level state representations. Finally, the Q-values are predicted based on the concatenation of high-level state representations. Our model extracts latent actions based on cosine similarity and can take into account the relationship between states, as well as the relationship between states and actions. We implement our model on six Q-networks and the experimental results show that the proposed model is more efficient, and more stability. In future work, we will focus on the impact of different dialogue environments on the proposed model, e.g., noisy environments.

6. Bibliographical References

Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. 2023. A comparison of LSTM and GRU networks for learning symbolic sequences. *arXiv:2107.02248*.

Cheng Chang, Runzhe Yang, Lu Chen, Xiang Zhou, and Kai Yu. 2017. Affordable on-line dialogue policy learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2200–2209.

Lu Chen, Zhi Chen, Bowen Tan, Sishan Long, Milica Gasic, and Kai Yu. 2019. AgentGraph: Toward universal dialogue management with structured deep reinforcement learning. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 27(9):1378–1391.

Junyoung Chung. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Presented in NIPS 2014 Deep Learning and Representation Learning Workshop*.

Pierre Colombo, Emile Chapuis, Matteo Manica, Emmanuel Vignon, Giovanna Varni, and Chloe

Clavel. 2020. Guiding attention in sequence-to-sequence models for dialogue act prediction. In *Proceedings of the genetic 34th AAAI Conference on Artificial Intelligence (AAAI)*, 2, pages 7594–7601.

Heriberto Cuayáhuitl, Simon Keizer, and Oliver Lemon. 2015. Strategic dialogue management via deep reinforcement learning. In *Proceedings of the 28th NIPS Workshop on Deep Reinforcement Learning*.

Daniele Gammelli, James Harrison, Francisco Pereira, and Marco Pavone. 2022. Graph meta-reinforcement learning for transferable autonomous mobility-on-demand. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2913–2923.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, pages 1025–1035.

Chenping Huang and Bin Cao. 2023. Learning dialogue policy efficiently through dyna proximal policy optimization. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 396–414.

Hyunmin Jeon and Gary Geunbae Lee. 2022. DORA: Towards policy optimization for task-oriented dialogue system with efficient context. *Computer Speech and Language*, 72:101310.

D Kim, P Tsiakoulis, C Breslin, M Henderson, M Szummer, B Thomson, and S Young. 2014. Incremental on-line adaptation of POMDP-based dialogue managers to extended domains. In *Proceedings of the International Speech Communication Association*, pages 1–5.

Wai Chung Kwan, Hong Ru Wang, Hui Min Wang, and Kam Fai Wong. 2023. A survey on recent advances and challenges in reinforcement learning methods for task-oriented dialogue policy learning. *Machine Intelligence Research*, 20:318–334.

Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. 2020. Maxmin Q-learning: Controlling the estimation bias of Q-learning. In *Proceedings of the 37th International Conference on Learning Representations (ICLR)*.

Yu Lei, Hongbin Pei, Hanqi Yan, and Wenjie Li. 2020. Reinforcement learning based recommendation with gaph convolutional Q-network. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR)*, pages 1757–1760.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017a. End-to-end task-completion neural dialogue systems. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, pages 733–743.

Xiujun Li, Zachary C. Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2017b. A user simulator for task-completion dialogues. *arXiv:1612.05688*.

Ziming Li, Sungjin Lee, Baolin Peng, Jinchao Li, Julia Kiseleva, Maarten de Rijke, Shahin Shayandeh, and Jianfeng Gao. 2020. Guided dialogue policy learning without adversarial learning in the loop. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2308–2317.

Zhouhan Lin, Minwei Feng, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Mohamed Morchid. 2022. Bidirectional internal memory gate recurrent neural networks for spoken language understanding. *International Journal of Speech Technology*, 25(1):19–27.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Kam-Fai Wong, and Shang-Yu Su. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2182–2192.

Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam Fai Wong. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement

learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2231–2240.

Rafael Figueiredo Prudencio, Marcos R. O. A. Maximo, and Esther Luna Colombini. 2023. A survey on offline reinforcement learning: taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, page early access.

Chang Tian, Wenpeng Yin, and Marie Francine Moens. 2022. Anti-overestimation dialogue policy learning for task-completion dialogue system. In *Proceedings of the Association for Computational Linguistics: NAACL Findings*, pages 565–577.

Quan Tu and Bin Wang. 2022. Conversational recommendation via hierarchical information modeling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2201–2205.

Sebastian Varges, Silvia Quarteroni, Giuseppe Riccardi, Alexei V Ivanov, and Pierluigi Roberti. 2009. Leveraging POMDPs trained with user simulations and rule-based dialogue management in a spoken dialogue system. In *Proceedings of the SIGDIAL 2009 Conference*, pages 156–159.

Huimin Wang, Baolin Peng, and Kam-Fai Wong. 2020. Learning efficient dialogue policy from demonstrations through shaping. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6355–6365.

Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. 2021. Modelling hierarchical structure between dialogue policy and natural language generator with option framework for task-oriented dialogue system. In *The International Conference on Learning Representations (ICLR)*.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Frcitas. 2016. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 1995–2003.

Yuchen Xiao, Joshua Hoffman, Tian Xia, and Christopher Amato. 2016. Deep reinforcement learning with double Q-learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2094–2100.

Cheng Yang;, Hao Wang;, Jian Tang;, Chuan Shi;, Maosong Sun;, Ganqu Cui;, and Zhiyuan Liu. 2021. Full-scale information diffusion prediction with reinforced recurrent networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(5):2271 – 2283.

Haodi Zhang, Zhichao Zeng, Keting Lu, Kaishun Wu, and Shiqi Zhang. 2022. Efficient Dialog Policy Learning by Reasoning with Contextual Knowledge. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pages 11667–11675.

Mingxin Zhang and Takahiro Shinozaki. 2022. DNN-rule hybrid dyna-q for dialog policy learning. In *Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1428–1434.

Rui Zhang, Zhenyu Wang, Mengdan Zheng, Yangyang Zhao, and Zhenhua Huang. 2021. Emotion-sensitive deep dyna-Q learning for task-completion dialogue policy learning. *Neurocomputing*, 459:122–130.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *Proceedings of the Conference 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–10.

Yangyang Zhao, Zhenyu Wang, Kai Yin, Rui Zhang, Zhenhua Huang, and Pei Wang. 2020. Dynamic reward-based dueling deep dyna-Q : Robust policy learning in noisy environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9676–9684.

Yangyang Zhao, Zhenyu Wang, Changxi Zhu, and Shihan Wang. 2021. Efficient dialogue complementary policy learning via deep Q-network policy and episodic memory policy. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4311–4323.