# Rescue Conversations from Dead-ends: Efficient Exploration for Task-oriented Dialogue Policy Optimization

**Yangyang Zhao**[1,2] and **Mehdi Dastani**[2] and **Jinchuan Long**[3*] and **Zhenyu Wang**[4] and **Shihan Wang**[2*]

[1]Changsha University of Science and Technology, China
[2]Utrecht University, the Netherlands
[3]Central South University, China
[4]South China University of Technology, China

yyz@csust.edu.cn; M.M.Dastani@uu.nl; longjc1226@163.com; s.wang2@uu.nl

## Abstract

Training a task-oriented dialogue policy using deep reinforcement learning is promising but requires extensive environment exploration. The amount of wasted invalid exploration makes policy learning inefficient. In this paper, we define and argue that dead-end states are important reasons for invalid exploration. When a conversation enters a dead-end state, regardless of the actions taken afterward, it will continue in a dead-end trajectory until the agent reaches a termination state or maximum turn. We propose a **D**ead-end **D**etection and **R**esurrection (DDR) method that detects dead-end states in an efficient manner and provides a rescue action to guide and correct the exploration direction. To prevent dialogue policies from repeating errors, DDR also performs dialogue data augmentation by adding relevant experiences that include dead-end states and penalties into the experience pool. We first validate the dead-end detection reliability and then demonstrate the effectiveness and generality of the method across various domains through experiments on four public dialogue datasets.

## 1 Introduction

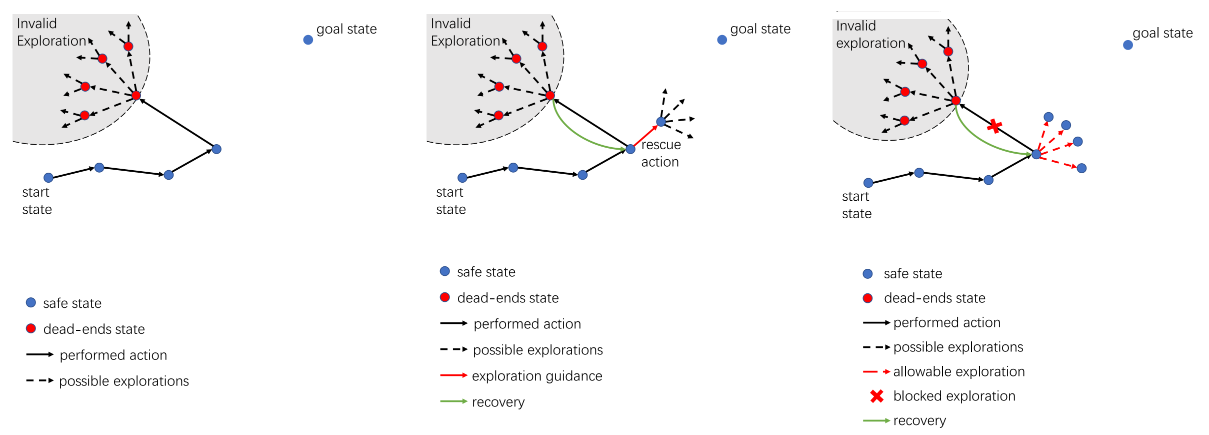
Task-oriented dialogue (TOD) systems are designed to help users perform a specific task (also referred to as the user goal), such as booking a restaurant or movie ticket (Wu et al., 2021). Typically, TOD systems are built on a structured database (DB) containing many task-related entries, where entries are potential solutions to the task at hand (Li et al., 2021). Every entry of DB is represented in terms of a set of attributes (known as slots) and their values. For example, in the movie domain, slots for the movie entry include *movie name, theater,* and *date*. In order to interact with users, TOD systems require a dialogue policy (DP) that determines how the system should reply to the user's input (Kwan et al., 2022). A well-performing DP is expected to gather as much user requirements as possible in as few dialogue turns as possible and, by using these requirements as constraints, to find matching entries in the DB to accomplish user goals (Geishauser et al., 2021).

According to TOD system construction, DP operates in two ways: through pipeline or end-to-end architectures. Pipeline architecture separates DP from Natural Language Understanding, Dialogue State Tracking, and Natural Language Generation modules (Ohashi and Higashinaka, 2022). End-to-end architecture integrates all or some of these modules into one sequence-to-sequence task but raises concerns about interpretability and controllability (Zhao et al., 2019; Kwan et al., 2022). In this paper, we focus on pipeline-based architecture with better controllability.

Partially Observable Markov Decision Process provides a natural way of modeling TOD, where the DP is often optimized using Deep Reinforcement Learning (Peng et al., 2018; Takanobu et al., 2020). However, in many real-world dialogue scenarios, transition probabilities or rewarding transitions are not known in advance, such that dialogue agents usually require extensive exploration of the environment (Li et al., 2016; Kwan et al., 2022). Some of these explorations are valid, and some are not, and the more invalid explorations there are, the less effective learning of dialogue policies becomes (Wu et al., 2019; Wu and Rasmussen, 2021). A dialogue segment constitutes an invalid exploration due to the effect of

*Corresponding authors.

Transactions of the Association for Computational Linguistics, vol. 12, pp. 1578–1596, 2024. https://doi.org/10.1162/tacl_a_00717
Action Editor: Kam-Fai Wong. Submission batch: 1/2024; Revision batch: 6/2024; Published 11/2024.



(a) Examples of an invalid exploration dialogue trajectory

(b) DDR with information gain exploration guide (DDR-IG)

(c) DDR with self-resimulation exploration guide (DDR-SE)

Figure 1: Illustration of invalid exploration (a), and our two DDR algorithms (b and c).

**Task**: When a user wants to book two tickets for the movie 'Creed' for tomorrow in the LA city and requests to know the theater that runs the movie Creed. We may get the following dialogues (where U represents the user and S represents the system).

| **Example 1** | **Example 2** |
|---|---|
| **U0**: Which theater will creed run?<br>**S1**: Which city would you like?<br>**U2**: I want to watch at LA.<br>**S3**: How many tickets do you need?<br>**U4**: I want 2 tikets please!<br><br>**S5**: Carmike summit 16 is available. (decision failures: inform the highest count in the database)<br>**U6**: Could you help me to book the tickets?<br>**S7**: Okay, you purchase of 2 tickets at carmike summit 16 for creed today at LA is confirmed.<br>**U8**: Oh, sorry! This is the wrong tickets!<br>**S9**: Yes, please!<br>**U10**: I want to set it up tomorrow.<br>**S11**: Date is not available!<br>.... | **U0**: Can I get some tickets for Creed?<br>**S1**: How many tickets do you need?<br>**U2**: I want 2 tickets, please!<br>**S3**: What date would you like to watch it?<br>**U4**: Tomorrow.<br>**S5**: What city would you like?<br>**U6**: LA (slot and value error from upstream modules: 'LA' is misrecognized as 'CA')<br>**S7**: Sorry, there is no ticket available.<br>**U8**: Can I get two tickets?<br>**S9**: Do you have any constraints on the price?<br>**U10**: I do not care.<br>**S11**: Do you want to purchase the tickets?<br>**U12**: Yes, i want.<br>**S13**: Oh sorry, there is no ticket available.<br>.... |

Table 1: Examples of two dead-end dialogues (invalid exploration starting from the blue text).

dead-ends.[1] As shown in Figure 1a, a dialogue agent can enter a dead-end due to its decision failures (Example 1 of Table 1) or accumulation of errors in upstream modules (Example 2 of Table 1). After reaching a dead-end state, the agent inevitably ends at a failed terminal state after some steps, regardless of what action it chooses. Such dialogue segments that start from a dead-end state cause invalid explorations, thus less effective learning of dialogue policies. Furthermore, such invalid explorations inherently produce a tremendously large number of training steps without getting useful information to achieve the goal state. In other words, it makes the dialogue longer by having a trajectory of unnecessary interactions, which makes it more difficult for the dialogue agent to learn which responses are bad among a lengthy dialogue.

This paper defines dead-end states and introduces a **D**ead-end **D**etection and **R**esurrection (DDR) method. DDR efficiently identifies dead-end states and provides rescue actions to guide exploration. By mitigating the adverse effects of dead-ends, DDR enhances exploration efficiency and generates more diverse training samples. Specifically, we define a reliable dead-end detection criterion for TOD systems. The state is considered a dead-end state if all trajectories starting from this state fail with probability one after a pre-defined number of dialogue turns. Once a certain dead-end is detected, a rescue module in DDR is activated, which transfers the current dead-end state to the previous safe state and provides rescue guidance. DDR comprises two types of rescue guidance: one that provides explicit rescue actions with maximum information gain (Figure 1b) and the other blocks incorrect paths, enabling diverse exploration (Figure 1c). We conducted extensive experiments to evaluate the advantages and applicability of the two types of rescue guidance.

To prevent repetitive mistakes, we employ a Dialogue Data Augmentation (DDA) module, introducing an experience pool with warning experiences, which consist of dead-end states and associated penalties. The effectiveness of DDR was validated on four public datasets, demonstrating its robustness even in noisy environments.

[1]See Section 3.1 for a formal definition.

Notably, DDR is a model-agnostic algorithm applicable to various RL-based DP methods. In summary, our contributions are as follows:

- Defining the concept of the dead-end state for dialogue policy exploration.
- Proposing and verifying a reliable dead-end detection criterion.
- Developing the DDR method to rescue conversations from dead-ends effectively.
- Demonstrating the dead-end state's impact on learning efficiency, and validating the effectiveness and generalizability of DDR across multiple datasets.

## 2 Related Work

**Learning Efficiency.** Our work connects to several research attempts to apply RL to learn dialogue policies for TOD systems. We touch on the research works aiming to make RL learning efficient by improving sample efficiency through I) generating effective exploration strategies, and II) manipulating generated training samples.

Type I: Research in the field of RL has developed many studies on general and effective exploration strategies to address the issue of sample efficiency, e.g., the $\epsilon$-greedy (Sutton, 1995), Boltzmann exploration (Kaelbling et al., 1996), Upper confidence bounds (Lai and Robbins, 1985), and Gaussian exploration (Lillicrap et al., 2015). Given that both our DDR and baseline methods have the potential to combine with these exploration strategies, resulting in a more efficient learning process, we conduct comparisons in our experiments based on the same exploration strategy for all baseline methods, rather than directly comparing with exploration strategies.

Type II: Some RL studies applied to TOD policies improve sample efficiency by manipulating the training samples. Companion learning introduced by Chen et al. (2017a,b) provides necessary guidance for dialogue policies, but it also transfers the training burden to the design of the teacher model (Zhao et al., 2021). Hindsight Experience Replay (HER) (Andrychowicz et al., 2017; Lu et al., 2019) extracts partial success segments from failed dialogues to synthesize successful artificial dialogues, but the quality of synthetic experience impacts HER's effectiveness (Cao et al., 2020). Trainable-Action-Mask (TAM) (Wu et al., 2019) learns action masks to block ineffective actions for generating more informative samples. Loop-Clipping Policy Optimisation (LCPO) (Wu and Rasmussen, 2021) clips useless trajectories to improve sample efficiency. However, both LCPO and HER may involve artificial synthesis and parameter tuning, leading to potential limitations in generating diverse and high-quality dialogues. Essentially, they only process the dialogue segments after the conversation ends, while our DDR detects and fixes the dialogue during the conversation.

Besides the above methods, there are non-deep learning-based planning methods for dialogue management (Ardissono et al., 1998; Ludwig, 2006; Teixeira and Dragoni, 2022). These approaches utilize domain knowledge to organize domain actions in a rational order, aiming to recover from misunderstandings. However, planning-based approaches, while successful in certain domains, often lack scalability and rely on expert-designed heuristics. In contrast, our method avoids these challenges, distinguishing it fundamentally from the mentioned approaches.

**Dead-end.** Kolobov et al. (2012) introduced and formalized the theory of dead-ends in Stochastic Shortest Path MDPs. Subsequently, Fatemi et al. (2019) extended this theory to the Atari gaming domain, introducing the concept of dead-ends along with corresponding safety conditions. In their work, undesirable terminal states were identified as dead-ends, representing situations where the game agent is compelled to terminate or restart before achieving its goals. Additionally, Irpan et al. (2019) introduced concepts related to dead-ends, with a primary focus on policy evaluation. They proposed the notion of feasible states—the states in which the agent has a non-zero probability of achieving success according to an optimal policy and does not face immediate defeat. Our definition of dead-ends draws inspiration from the aforementioned studies and applies it to the domain of TOD. It should be noted that due to unique characteristics in different domains, these studies are not applicable in the dialogue domain.

In the dialogue domain, Bui et al. (2004) proposed a simple solution (RDPM) to deal with dead-ends by relaxing one of the user's constraints as unknown. This approach heavily relies on manual intervention to select acceptable alternatives and cannot learn from past dead-end

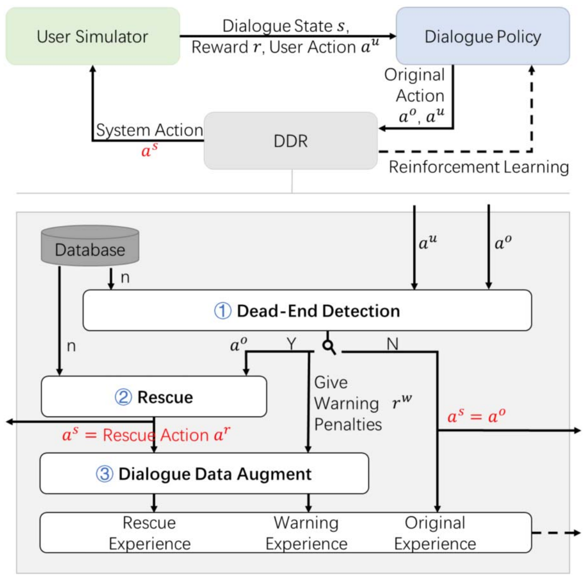


Figure 2: Illustration for dialogue policy optimization using proposed DDR method.

experiences. Consequently, it often faces recurring dead-end problems repeatedly and is limited to simple dialogue scenarios. In contrast, our approach can automatically evaluate and select appropriate recovery actions. In Hierarchical Reinforcement Learning With Guidance (HRLG), a related concept of confounded state was introduced, representing a situation within a dialogue where the same state appears three or more times (Rohmatillah and Chien, 2023). This differs from our notion of a dead-end state. We concentrate on the states where, if the dialogue continues from that state, it inevitably fails (invalid explorations), regardless of the chosen actions. In such cases, actions and states are not necessarily repeated. In other words, dead-end states cover the confounded states, but not vice versa.[2] Additionally, HRLG focuses on multi-domain dialogue tasks rather than invalid exploration issues.

## 3 Method

As illustrated in Figure 2, DDR consists of three main stages: 1) *Dead-end Detection* process automatically detects whether the current conversation with the original action $a^o$ leads to a dead-end. If so, 2) *Rescue* process transfers the current dead-end state back to the previous safe state and provides a rescue action $a^r$ as the system action $a^s$ for the response. And 3) *Dialogue Data Augment* process is performed by adding a warning experience containing this dead-end state and warning penalties into the experience buffer to prevent the dialogue policy from repeatedly making the same mistake. If $a^o$ does not lead to a dead-end state, $a^o$ is transferred directly to the user simulator as a $a^s$ without *Rescue* and *Dialogue Data Augment* stages. These stages are described in detail in the following subsections.[3]

### 3.1 Dead-end States and Their Detection

When a conversation enters a dead-end state $s$, regardless of actions taken afterward, it will continue in a dead-end trajectory until the agent reaches a failed termination state or maximum turn. Thus, if a state is a dead-end state, all possible states following this state are also dead-end states. We use dead-end to denote the trajectory from the initial dead-end state to the failed termination state (the end state of failing trajectory) of this conversation trajectory. To this end, we need to detect the initial dead-end state (referred to as dead-end detection) to avoid invalid exploration.

**Definition 1.** *(Dead-end State). A state $s$ is a dead-end state if all trajectories starting from this state fail with probability one after a pre-defined number of turns.*

Database entries and user queries can be used to assist TOD in identifying matching entries for providing the expected actions or information. The purpose of the dialogue agent is to obtain the user's constraints through interactions and find entries from the database that meet these constraints to accomplish the user goal. The number of matching entries, denoted as $n$, changes during the conversation as the user provides more constraints. Therefore, $n$ serves as an important indicator to detect whether a dialogue action will lead to a dead-end.[4] When $n$ disappears (i.e.,

[2]From our study, we found that the occurrence of repeated states within overall dead-ends is rare. For instance, within 10K randomly sampled dialogues with dead-ends, only about 17.6% are related to repeated states.

[3]When describing the relationship with policies, we implicitly employ deep Q-networks (DQN) as the foundation for implementing the policy. However, it should be noted that our approach is generalizable and can be extended to other RL-based algorithms, as discussed in Section 4.8.

[4]The number of matching entries $n$ is also used as a feature of the dialogue state, obtained by querying the database. The dialogue state serves as the input for the dialogue policy, acquiring $n$ easily attainable.

$n_{t-1} \neq 0$ and $n_t = 0$ and dialogue_end_flag = False), it indicates that the dialogue has entered a dead-end. This situation arises because the user goal has transitioned from a solvable state to an unsolvable one, preventing the dialogue policy from finding entries that meet the user's needs. In this way, our dead-end detection enables the identification of dead-ends in real-time during the dialogue, without requiring human intervention.[5]

## 3.2 Rescue

In the rescue phase, we consider two approaches of exploration guidance: *Information Gain Rescue* and *Self-resimulation Rescue*. The former directly provides an explicit exploration guidance, while the latter only blocks wrong exploration paths, allowing explore diverse paths.

### 3.2.1 Information Gain Rescue

The main objective of TOD systems is to help users accomplish their specified tasks, of which acquiring information through dialogue interactions is the first important aspect (Geishauser et al., 2021). Therefore, optimal dialogue policies are expected to gain maximal information on the user's needs by asking targeted questions. Information gain (IG) is a promising measure for evaluating the system actions as it determines the amount of information that can be obtained by a system action if it is performed in a particular state (Kent, 1983). Inspired by the usage of IG in dialogue systems (Padmakumar and Mooney, 2021; White et al., 2021), we construct an information gain rescue approach that provides explicit exploration guidance on the database (Figure 1b).

Specifically, IG allows a better assessment of the amount of information carried or gained by knowing a certain slot-value. Prioritizing queries for slots with more IG can help the dialogue policy accomplish user goals efficiently. Formally, the IG at the current state for each query action $a$ of the form request($slot$) is:

$$\begin{aligned} IG(N,a) &= H(N) - H(N|V_a) \quad (1)\\ &= H(N) - \sum\nolimits_{v\in V_a} p(N^v)H(N^v|V_a = v) \end{aligned}$$

$$H(N) = -\sum\nolimits_{n_i\in N} p(n_i)log\ p(n_i) \quad (2)$$

[5]While our dead-end concept broadly encompasses all dead-end scenarios, this paper focuses on those that the dialogue policy can resolve. For more details, please refer to Section 5.

where $N$ denotes a set of entries in the database, $n_i$ denotes the $i$th entry in $N$, $V_a$ is a variable whose values $v$ are the possible answers that the user can provide after executing the system action $a$, and $N^v$ is a subset of $N$ filtered by $v$. $p(N^v)$ denotes the probability that an entry in $N$ contains the value $v$, while $p(n_i)$ represents the probability of entry $n_i$ appeared in $N$. $H(N)$ and $H(N|V_a)$ denote the information entropy and conditional entropy before and after executing action $a$, respectively. The difference between these two denotes the information gain $IG(N,a)$.

Since other types of actions (e.g., *inform, bye, hello*, etc.) tend to be less helpful in filtering entries in DB, the information gains of those types of action are always more minor compared to the *request* type. It leads to the possibility that the dialogue agent is biased toward the *request* action, which is obviously not reasonable. To correct this bias, the type of action needs to be chosen first based on the number of matching entries $n$ and the user's action before computing IG.

When the constraints provided by the user are challenging to target specific entries in the database, the system needs to guide the user in providing more constraints through a request, denoted as $a_s$ = request($slot$). The specific slot for the request is determined by selecting the one with the maximum IG according to Eq. (1). When user constraints are sufficient to filter specific entries in the database, the system can provide the information the user wants to know ($a_s$ = inform($slot$)), or perform the action the user has asked for, for example, to book a ticket ($a_s$ = booking_confirmed).

Compared with the work by Geishauser et al. (2021), our information gain rescue considers the variation of the probability distribution over all slots, which covers complete information gain. Moreover, previous studies that utilize information gain or maximum entropy are often focused on other types of dialogue systems such as question-answering systems (one-turn) (Padmakumar and Mooney, 2021; White et al., 2021), and recommender systems (lack of clear user goal) (Lei et al., 2020; Deng et al., 2021; Kim et al., 2023), which are outside the scope of our focus area (multi-turn and specific user goal).

### 3.2.2 Self-resimulation Rescue

As shown in Figure 1c, rather than directly providing an explicit exploration direction, this rescue guidance shields dialog policy from wrong paths and allows them to explore more diverse directions.

In each step, the dialogue policy observes a state $s$ and selects an action $a^s$ using an $\epsilon$-greedy policy, where a random action is selected with $\epsilon$ probability, or otherwise, a greedy action is selected, $a^s = \text{argmax}_a Q(s, a; \theta_Q)$, where $Q(s, a; \theta_Q)$ is the approximated value function that is implemented as a Multi-Layer Perceptron with parameter $\theta_Q$. It should be noted that even if the dead-end state has been detected and restored to a safe state, the direct use of the initial DQN without additional training will always select the same action. Instead, we set the probability of selecting the action, which leads the conversation to a dead-end to 0. We then perform a self-resimulation by using the dialogue policy to re-select from the action space, excluding the action that causes the dead-end state. It is worth noting that the used self-resimulation differs from the dialogue rollout (Lewis et al., 2017). Dialogue rollout does lookahead for all unexecuted actions in the candidate set, while our approach focuses on the already executed actions. It eliminates the need to evaluate candidate actions one by one (and therefore also does not have evaluation errors caused by that). Moreover, it blocks a dead-end path and allows dialogue agents to explore more freely than the information gain-based exploration guidance.

### 3.3 Dialogue Data Augmentation

Self-reflection aids in averting recurring errors among humans (Farthing, 1992). Inspired by it, we added a Dialogue Data Augmentation module to prevent dialogue policies from continually making the same mistakes. This is done by introducing an experience replay buffer that contains three types of experiences: Original experience, rescue experience, and warning experience. Leveraging these experiences, we can improve the value functions $Q(\cdot)$ of DDR agents by minimizing the mean-squared loss function.

For state $s$, the dialogue agent performs the (original) action $a^o$, observes the updated state $s'$, and receives the reward $r$. If state $s$ is detected as an dead-end state, the rescue experience $E^r \leftarrow \{s, a^r, s^r, r^r\}$ and the warning experience $E^w \leftarrow \{s, a^o, s', r + r^w\}$ are stored in the experience replay buffer. In the rescue experience, the original action $a^o$ is replaced with a rescue action $a^r$, and their corresponding reward and next state update are adjusted accordingly. In the warning experience, the original action $a^o$ is retained, but a penalty $r^w$ is added. Otherwise, only the original experience $E^o \leftarrow \{s, a^o, s', r\}$ is stored.

During the random sampling process, if the sampled experience includes a dead-end state, both the dead-end experience and its corresponding warning experience will be sampled simultaneously. If the sampled experience does not contain a dead-end state, only the original experience will be sampled.

## 4 Experiments

Our experiments have five objectives: I) Verifying the effectiveness of our DDR in simulated (Section 4.4) and human (Section 4.9) experiments; II) Investigating the advantages and applicability of two rescue modules (Section 4.5); III) Analyzing the efficiency of the dead-end detection (Section 4.6); IV) Evaluating the individual contribution of the different modules in our DDR (Section 4.7), and V) Validating the generality of our DDR (Section 4.8).[6]

### 4.1 Datasets

The experiments were performed on the MS Dialogue Challenge,[7] which provided three standard datasets, user simulators, and a unified experimental environment for collaboration and benchmarking in the dialogue research community (Li et al., 2016, 2018). Further validation was conducted on a complex multi-domain dataset, MultiWOZ 2.1 (Budzianowski et al., 2018) provided by the ConvLab-2 platform.[8] The number of slots in user goals affects dataset difficulty, with more domains leading to more slots. ConvLab-2's default user goals involve 1–3 domains, similar to the Restaurant dataset. To highlight differences among the four datasets, we increased the MultiWOZ dataset's difficulty by generating user goals with 5–7 domains, introducing more challenging features. Simulated evaluation employs three publicly available and widely accepted criteria: success rate, average turns, and average reward. The human evaluation utilized established metrics consistent with our study's datasets and commonly

[6]https://github.com/zhaoyangyangHH/DDR/tree/main.

[7]https://github.com/xiul-msr/e2e_dialog_challenge/tree/master.

[8]https://github.com/thu-coai/ConvLab-2.

used ones (e.g., Liu et al., 2021; Wang et al., 2020; Tang et al., 2018): success rate (SR) and average scores (AS) ranging from 1 to 5. These metrics assess naturalness, coherence, and task completion capability.

To simulate realistic dialogue scenarios, all datasets include a minority of unresolvable user goals, where the user's request genuinely cannot be fulfilled due to the absence of matching entries in the database. MultiWOZ datas et allows providing alternative *inform_slot* values to find alternative solutions. We ensured that the proportion of unresolvable user goals was consistent across the different datasets. Additionally, we introduced similar scenarios in human evaluations.

### 4.2 Baselines

We compare our approach with different dialogue agents that make RL effective and efficient by similarly manipulating training samples as baselines:[9]

**DQN** agent learned with one standard Deep Q-network (Mnih et al., 2015). **RDPM** agent conducts dialogues based on the designed global dialogue flow management (Bui et al., 2004). When encountering a dead-end, it relaxes one of the user's constraints as unknown. **HER** agent is learned from synthesized successful dialogues, which are created by extracting successful dialogue segments from failed dialogues (Lu et al., 2019). **LCPO** agent is learned from clipped dialogue trajectories (Wu and Rasmussen, 2021).[10] **DDR-IG** agent is learned with a DQN that incorporates proposed dead-end detection, information gain rescue, and dialogue data augmentation. **DDR-SE** agent is similar to DDR-IG except that the rescue module is based on self-resimulation.

### 4.3 Settings

For a fair comparison, all dialogue agents are trained using RL's typical representative DQN, except for the non-RL based RDPM method. It is worth noting that our method is general, so in the General Evaluation section, we substituted other RL algorithms for the training network architecture. Each policy network consists of one hidden layer with 80 nodes. Common parameters are standardized across all models, while unique optimal parameters are selected for each model.[11] The learning rate is 0.001, and the batch size is 16. The optimizer is Adam. $\epsilon$-greedy is always applied for exploration with 0.1 and decayed to 0.01 during training. All baselines employ $\epsilon$-greedy to select actions, whereas our DDR-$^*$ utilizes either IG or SE to select recovery actions when encountering a dead-end. For other non-recovery cases, the actions of both DDR-$^*$ are selected through the default $\epsilon$-greedy strategy (same as the baselines). The buffer size $D$ is set to 10000, and the number of warm start epochs is 120. The reward function provides a large bonus of $2L$ for success and a penalty of $-L$ for failure. Additionally, a fixed penalty of $-1$ is given for each turn to encourage short interactions. All agents are limited to the same number of turns ($L = 30$ for single domains and $L = 40$ for multi-domains), and their discount factor is set to 0.95.

All results are averaged from ten runs of 1,000 dialogues each, using different random seeds, following training on one dialogue.

### 4.4 Effectiveness Evaluation

This section validates our methods with simulated experiments and the case study.

#### 4.4.1 Simulated Experiments

To verify the effectiveness of our method, we conduct experiments on three single-domain datasets in a normal environment (without noise). Table 2 reports the main result of different agents. The results show that both our DDR methods outperform other baselines with a statistically significant margin. While HER shows significant improvement in the movie and restaurant domains compared to DQN, it performs poorly in the taxi domain. The complexity of the taxi domain resulted in fewer successful dialogues early on, leading to fewer synthesized experiences in HER. Additionally, some synthesized experiences lacked coherence,

[9]It is worth noting that we do not consider methods that use additional human intervention since an important advantage of our approach is that it does not require any human intervention (Ardissono et al., 1998; Ludwig, 2006; Teixeira and Dragoni, 2022).

[10]Since Wu and Rasmussen (2021) demonstrated the superiority of the LCPO compared to TAM (Wu et al., 2019), we only use the better-performing LCPO as the baseline model.

[11]Please be aware that for the *dead-end management threshold* in RDPM, we did not choose the value that was optimal for the simulation experiments. Instead, we selected the optimal value of 10 based on a combination of simulated and human experiments. Even if we had chosen the best parameter of RDPM from the simulation experiments, our DDR-IG still outperforms RDPM. Detailed experimental results are provided in Appendix A.

| Domain | Agent | Epoch $= 150$ | | | Epoch $= 350$ | | | Epoch $= 500$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Success ↑ | Reward↑ | Turns↓ | Success↑ | Reward↑ | Turns ↓ | Success↑ | Reward↑ | Turns ↓ |
| Movie | DQN | 0.1546 | −25.51 | 20.34 | 0.5402 | 10.15 | 18.42 | 0.5746 | 13.30 | 18.32 |
| | RDPM | 0.3788 | −8.57 | 29.60 | 0.7003 | 25.03 | 20.01 | 0.7679 | 30.74 | 18.80 |
| | HER | 0.4490 | 0.38 | 21.57 | 0.6388 | 20.05 | 16.38 | 0.7104 | 27.57 | 14.24 |
| | LCPO | 0.4108 | −1.22 | 21.33 | 0.6024 | 19.85 | 16.12 | 0.7258 | 29.74 | 14.35 |
| | DDR-IG | 0.4731 | 3.02 | **18.04** | **0.7567** | **31.92** | **12.30** | **0.8021** | **36.37** | **11.34** |
| | DDR-SE | **0.5329** | **8.53** | 20.36 | 0.7115 | 27.27 | 15.02 | 0.7585 | 32.17 | 13.70 |
| Restaurant | DQN | 0.0530 | −39.30 | 29.75 | 0.1971 | −24.22 | 25.92 | 0.4110 | −3.52 | 23.02 |
| | RDPM | 0.0974 | −35.66 | 26.46 | 0.4993 | 2.02 | 24.63 | 0.7166 | 24.86 | 22.88 |
| | HER | 0.1022 | −33.70 | 27.79 | 0.4840 | 4.01 | 21.10 | 0.6664 | 22.10 | 17.75 |
| | LCPO | 0.0903 | −35.65 | 27.64 | 0.4831 | 3.74 | 21.30 | 0.6949 | 27.69 | 16.94 |
| | DDR-IG | 0.1038 | −33.26 | **26.22** | **0.6230** | **17.32** | **17.79** | 0.7582 | 30.74 | **15.44** |
| | DDR-SE | **0.1064** | **−32.79** | 26.73 | 0.4970 | 4.82 | 21.83 | **0.7716** | **32.25** | 16.38 |
| Taxi | DQN | 0.0548 | −37.5 | 26.85 | 0.2707 | −16.05 | 22.82 | 0.6300 | 18.31 | 18.78 |
| | RDPM | 0.0532 | −37.96 | 27.50 | 0.3894 | −6.24 | 24.41 | 0.6625 | 19.24 | 20.77 |
| | HER | 0.0359 | −40.51 | 29.47 | 0.1786 | −25.27 | 24.64 | 0.5003 | 8.49 | 25.15 |
| | LCPO | 0.0674 | 36.65 | 27.79 | 0.3565 | 5.38 | 21.90 | 0.6404 | 19.69 | 19.01 |
| | DDR-IG | **0.0984** | **−33.96** | **25.14** | **0.5299** | **9.55** | **16.83** | **0.7821** | **32.90** | **14.85** |
| | DDR-SE | 0.0790 | −35.57 | 27.35 | 0.3974 | −3.21 | 19.95 | 0.6654 | 21.64 | 18.49 |

Table 2: Results of different agents on three datasets in the normal environment. The highest performance in each column is highlighted. The difference between results of all agent pairs evaluated at the same epoch is statistically significant with a t-test ($p < 0.05$). These specific epochs were chosen to demonstrate the model's performance at different training stages (150 = early training stage, 350 = mid-training stage, 500 = performance after convergence).

resulting in poor-quality dialogues. Similarly, LCPO demonstrated improvements in the movie and restaurant domains but had limited impact in the taxi domain. Dialogues in LCPO were concise, leading to shorter average dialogue turns, but dead-end dialogues still struggled to be resolved. Although RDPM could alleviate dead-ends by relaxing one of the user's constraints, it fails to learn from past dead-ends, often encountering the same dead-ends repeatedly, leading to dialogue failures due to excessive turns. In contrast, our two DDR methods effectively and diversely explore by promptly detecting and recovering from dead-ends and learning from past dead-end experiences.

Further validation was also performed on the complex multi-domain MultiWOZ dataset, where each agent was trained for 1000 epochs. Results in Table 3 demonstrate better performance achieved by our two DDR methods, consistent with the single-domain results. Thus, our DDR methods are widely applicable in various scenarios.

| Agents | Success ↑ | Reward ↑ | Turns ↓ |
|---|---|---|---|
| DQN | 0.0726 | −48.83 | 37.08 |
| RDPM | 0.1166 | −44.56 | 38.63 |
| HER | 0.0862 | −47.04 | 36.76 |
| LCPO | 0.1008 | −40.71 | 34.53 |
| DDR-SE | 0.1793 | −33.85 | 32.72 |
| DDR-IG | **0.2010** | **−31.01** | **32.27** |

Table 3: Results of different agents on the MultiWOZ dataset.

### 4.4.2 Case Study

Table 4 shows how our DDR-IG method handles the same dead-end dialogue issues encountered by DQN agents. For a comparison with DQN's dialogue, refer to Example 1 in Table 1. In the DQN example, the agent preemptively provides the highest-count theater value (inform_theater = Carmike summit 16) in 'S5' based on available constraints and ignoring the subsequent date constraint. Consequently, none of the movie items in the database matching all the constraints specified in the user goal were provided after a new constraint was added ($n = 0$). In contrast, our DDR-IG method, as shown in Table 4, detected a conflicting action when obtaining the date constraint ''date=tomorrow'' and identified a conflict with the constraint ''theater=carmike summit 16'',

| **Task:** |
|---|
| A user wants to book two 'creed' tickets for tomorrow in the LA city and request to know the theater that the show is playing in. |
| **DDR-IG** |
| *U0*: Which theater will creed play? |
| *S1*: Which city would you like? |
| *U2*: I want to watch at LA. |
| *S3*: How many tickets do you need? |
| *U4*: I want 2 tikets please! |
| *S5*: Carmike summit 16 is available. |
| *U6*: Could you help me to book the tickets? |
| *S7*: Okay, you purchase of 2 tickets at carmike summit 16 for creed today at LA is confirmed. |
| *U8*: Oh, sorry! This is the wrong tickets! |
| *S9*: Which data would you like? |
| *U10*: I want to set it up tomorrow. |
| *S11*: Science Park is available. |
| *U12*: Could you help me to book the tickets? |
| *S13*: Okay, you purchase of 2 tickets at science park for creed tomorrow at LA is confirmed. |
| *U14*: Thanks, bye. |
| *S15*: Bye. |

Table 4: Examples of dialogue encountering dead-end problems by DDR-IG, where we use *U* and *S* to indicate user and system turns respectively.

despite this value having the highest count in the database. The DDR-IG method detected that the conversation had reached a dead-end and restored the dialogue to the turn that produced the conflicting action, 'S5'. It selected the action with the maximum information gain, ''request(date)'', i.e., asking for the date, and then provided the booking information, thus resolving the dead-end dialogue.

## 4.5 Robustness Evaluation

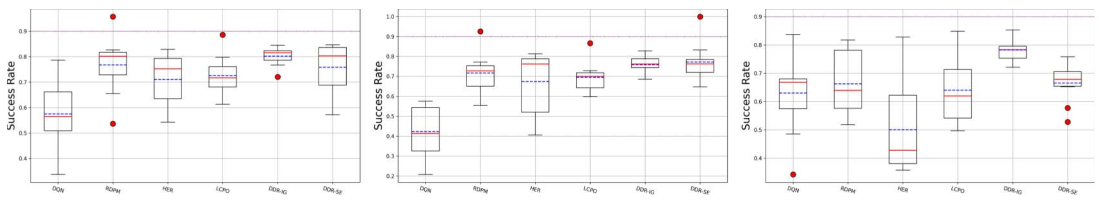
As illustrated by Example 2 in Table 1, real-life dialogue environments often feature noise from upstream modules and ambiguous user utterances, exacerbating dead-end issues in dialogues (Zhao et al., 2020). A dialogue policy should sustain conversations despite these inaccuracies. To evaluate the robustness of our methods in addressing dead-ends caused by noisy environments, we conducted experiments with slot and value error rates of 0%, 5%, 15%, and 25% within the error model for simulating noise.

The results, shown in Figure 3–Figure 6, reveal that as noise levels increase, the performance of all agents declines, but to varying degrees.[12] Notably, DQN and HER exhibit significant drops and unstable performance, especially in the more challenging Taxi dataset. Despite LCPO's performance being relatively stable compared to HER, it still shows a significant decline. RDPM shows substantial performance improvement in noise-free and low-noise environments, particularly in the simpler Movie dataset, but its performance advantage diminishes as noise increases. This may be due to noise exacerbating dead-end problems, highlighting RDPM's difficulty in learning from dead-end experiences. Additionally, in more complex datasets with numerous database entries, the number of entries after releasing a user constraint often exceeds the dead-end management threshold. Consequently, RDPM frequently releases random user constraints, which may be satisfactory to the user, leading to dialogue failure due to exceeding the maximum number of turns. In contrast, our DDR-SE maintains the best performance in noisy environments through exploration and data augmentation, despite a decrease in success rate. DDR-IG performs best in noise-free environments but shows a marked performance decline under noisy conditions, similar to baseline models. We attribute this disparity to the utilization of information gain for action selection, which helps DDR-IG avoid suboptimal actions, enhancing its efficiency in noise-free conditions. However, DDR-IG's usability is constrained by its inability to estimate information gains in the presence of noise accurately. Conversely, DDR-SE exhibited greater adaptability in handling noise effects through exploration, allowing it to mitigate the impact of noise effectively. This flexibility in navigating noisy environments enabled DDR-SE to outperform DDR-IG in such conditions.

## 4.6 Effectiveness of DDR for Dead-ends

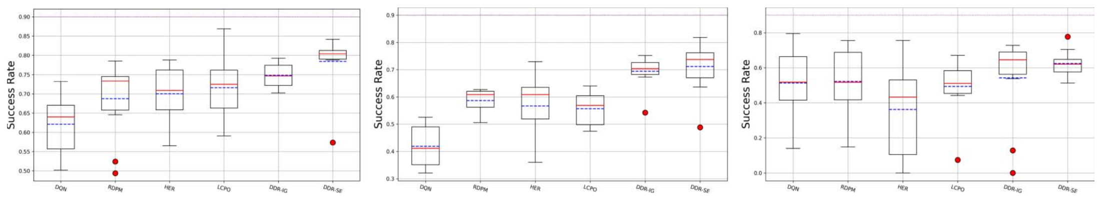
We examined the percentage of dead-end conversations among the failed ones for different

[12]Results of the robustness experiments for all error rates including those of 0%, 5%, 10%, 15%, 20%, 25%, and 30% are shown in Appendix B. Since all methods tend to fail to learn above a 30% error rate, we limited our tests to these values.



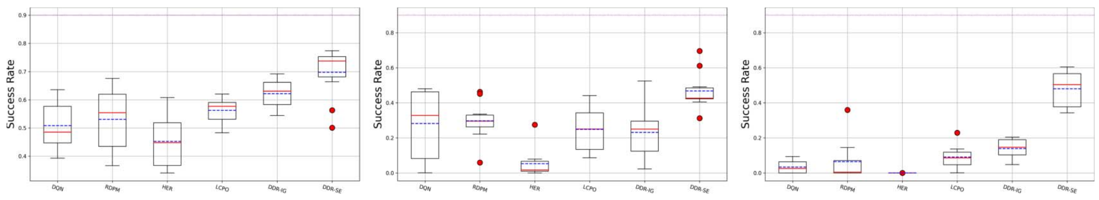
(a) Movie (b) Restaurant (c) Taxi

Figure 3: Results of different agents on three datasets in environment with $0\%$ noise.



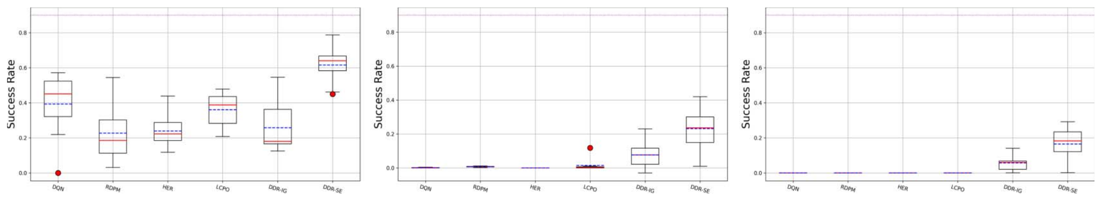
(a) Movie (b) Restaurant (c) Taxi

Figure 4: Results of different agents on three datasets in environment with $5\%$ noise.



(a) Movie (b) Restaurant (c) Taxi

Figure 5: Results of different agents on three datasets in environment with $15\%$ noise.



(a) Movie (b) Restaurant (c) Taxi

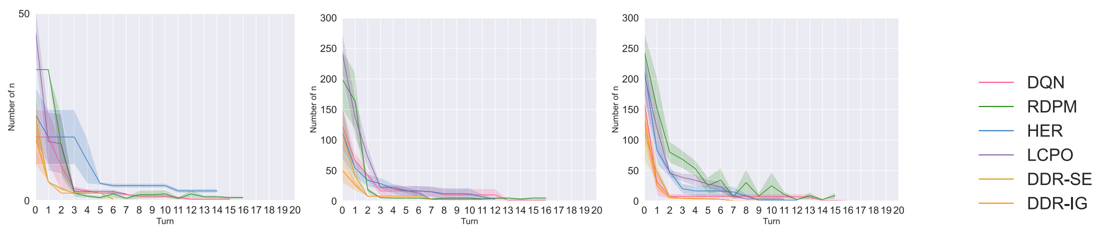
Figure 6: Results of different agents on three datasets in environment with $25\%$ noise.

dialogue agents. Higher percentages indicate a higher generation of low-quality conversations with dead-ends. As shown in Table 5, both DDR methods significantly alleviate this problem. This demonstrates that our two DDR methods accurately identify dead-ends with high probability and rescue them instead of continuing meaningless conversations.

Furthermore, to further interpret the effectiveness of DDR for dead-ends, we randomly selected three user goals in the movie domain to visualize the changes in the average value of $n$ for each agent over different dialogue turns during testing, as depicted in Figure 7. In the ideal case, $n$ rapidly diminishes to a non-zero value and eventually disappears. This indicates the dialogue agent efficiently targets the matching entries through effective interactions, achieving the user goal. When

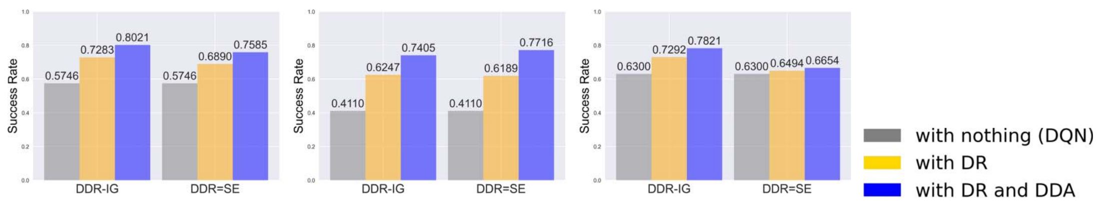
| Agent | Movie | Restaurant | Taxi | MultiWOZ |
|---|---|---|---|---|
| DQN | 64.29% ± 11.06% | 82.58% ± 5.06% | 88.36% ± 5.58% | 73.41% ± 6.55% |
| RDPM | 24.91% ± 5.31% | 36.24% ± 6.96% | 31. 05% ± 7.57% | 35.20% ± 12.63% |
| HER | 37.35% ± 9.88% | 56.19% ± 10.33% | 71.28% ± 9.18% | 60.73% ± 14.25% |
| LCPO | 36.22% ± 8.09% | 44.50% ± 4.26% | 50.88% ± 5.57% | 49.32% ± 8.66% |
| DDR-SE | 23.25% ± 6.34% | 30.04% ± 3.31% | 29.22% ± 2.09% | 26.259% ± 7.26% |
| DDR-IG | **3.32% ± 0.75%** | **7.40% ± 1.82%** | **8.21% ± 1.15%** | **16.80% ± 5.76%** |

Table 5: Statistics on the percentage of failed conversations with dead-ends for different agents in three domains. ± is the mean standard deviation of the average of 10 runs with different seeds.



(a) User goal 34 (b) User goal 97 (c) User goal 121

Figure 7: Visualization of the variation of $n$ with turns of interaction for different user goals. All results are the average values from 10 runs with different random seeds.



(a) Movie (b) Restaurant (c) Taxi

Figure 8: The ablation experiment of our DDR methods in Movie, Restaurant, and Taxi domains.

the value of $n$ remains 0 for an extended period while the conversation continues, it indicates that the dialogue has entered a dead-end. Taking Figure 7a as an example, for DQN, $n$ drops to 0 at turn 12, but the conversation continues until turn 15, indicating the dialogue encountered a dead-end during this period. In contrast, for HER, $n$ decreases to a certain range at the turn 11, but the conversation extends until turn 14, indicating that although the conversation did not reach a dead-end, it possibly encountered repetitive states between turns 11 and 14. This is a common occurrence in HER due to the inclusion of repetitive dialogues in its synthesized successful dialogues, leading to verbosity. For RDPM, although the dialogue often enters a dead-end by turn 7 and is promptly rescued, the significant fluctuations in $n$ could result in the deletion of valuable information during the dialogue, prolonging the process before achieving success. LCPO effectively avoids repetitive state dilemmas like HER by trimming dialogues. Our approaches (yellow and orange) consistently demonstrate a faster understanding of user needs and more effective conversations, successfully targeting matching entries to accomplish user goals by turns 4 and 6, respectively. As a result, their average performance does not exhibit the fluctuations seen in RDPM.

Those results are consistent with the case study analysis in Section 4.4.2, further showcasing the effectiveness of our approach in dealing with the dead-end issue.

## 4.7 Ablation Evaluation

To evaluate the impact of the *Dialogue Data Augmentation* (DDA) module and the *Dead-end Detection and Recovery* (DR) module individually in our DDR-* methods, we conducted an ablation study. The results of the ablation study for the two DDR methods on three domains are presented in Figure 8. We set up three versions of DDR: 1) with nothing (gray) actually refers to DQN, which lacks both the DDA and DR modules. 2) with DR (yellow) is the DDR-* method without DDA, meaning it only detects and recovers from dead-ends but does not augment dialogue data based on dead-end experiences. 3) with DR and DDA (blue) represents the DDR-* method proposed in the paper, incorporating both DDR and DR modules. As the experience utilized for DDA was a byproduct of the DR process, we did not include the DDA module as a standalone component. We found that both the DR and DDA modules had a positive impact on the DDR methods, with the DR module showing a more pronounced improvement in performance. The most significant decrease in performance occurred when both modules were removed. Combining both modules resulted in the highest overall performance for DDR-* methods.

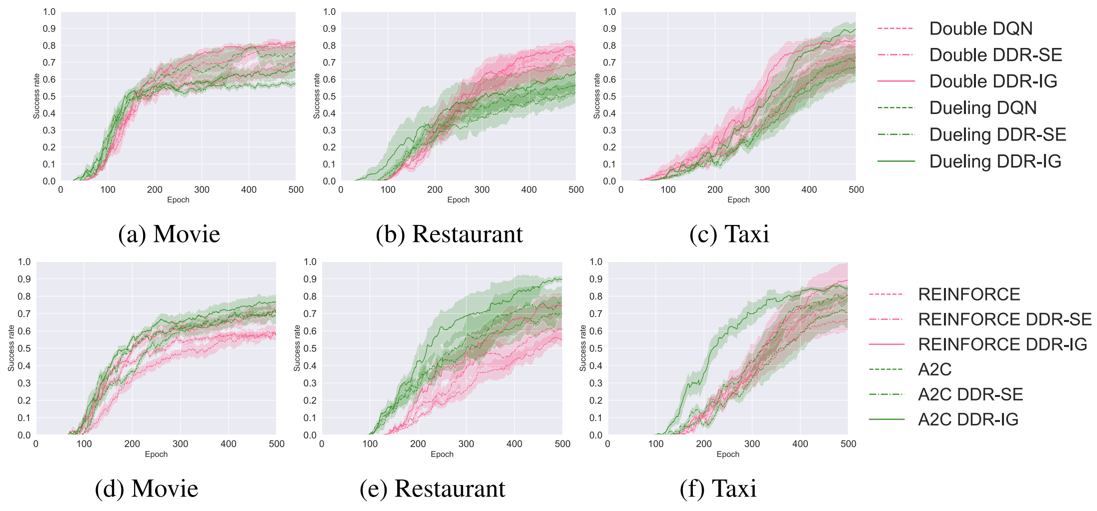


Figure 9: Incorporating different RL algorithms (Double DQN (van Hasselt et al., 2015), Dueling DQN (Wang et al., 2016), REINFORCE (Williams, 1992), A2C (Fatemi et al., 2016)) with our two methods on three datasets. The standalone results obtained are shown in Appendix D.

In summary, our ablation study confirms the importance of each module in the DDR methods and highlights the necessity of integrating both modules to achieve optimal performance.

To further validate these findings, we integrated our two modules into two other methods (Dueling DQN and A2C) and found similar trends. The results are provided in Appendix C.

## 4.8 Generality Evaluation

In previous experiments, all baseline and DDR agents were trained using a typical representative of reinforcement learning, DQN, for a fair comparison. We aim to verify that our DDR methods are valid not only on the DQN, but also on other RL-based methods. Thus, we use four other commonly used RL algorithms in combination with our two DDR methods in this generality evaluation. Figure 9 shows their learning curves in three domains. It is evident that integrating our DDR methods improves their learning efficiency to varying degrees. This observation allows us to conclude that our approach generalizes effectively to different RL-based dialogue policy algorithms.

## 4.9 Human Evaluation

Since user simulators do not fully capture the complexity of human interactions, we evaluated the feasibility of DDR-* methods in real-world scenarios. Based on the publicly available human evaluation settings provided by the dataset platform, we employed 68 human evaluators for human evaluation. Each agent underwent evaluation with users randomly selecting five goals without knowing the agent's identity. To approximate realistic dialogue scenarios, four goals directly matched database entries, while the remaining goal initially had no match, allowing the system to provide alternative answers to meet the user's alternative requirements. Users could terminate the dialogue at any time if it was deemed unproductive. Each agent collected 340 valid dialogues (5 per user) per domain. Human evaluation results in Table 6 showed superior performance of our methods over the baselines, consistent with simulated experiment outcomes. DQN is often terminated early due to a lack of progress, leading to a lower SR and AS. RDPM and HER received low AS ratings due to excessive dialogue length. Although LCPO was more concise than the previous agents, it still struggled with dead-ends. In contrast, our DDR-* methods, particularly DDR-IG, excelled by actively resolving dead-ends, demonstrating outstanding performance.

| **Model** | Movie | | Restaurant | | Taxi | | MultiWOZ | |
|---|---|---|---|---|---|---|---|---|
| | **SR** | **AS** | **SR** | **AS** | **SR** | **AS** | **SR** | **AS** |
| **DQN** | 0.3764 | 2.31 | 0.2724 | 2.16 | 0.3494 | 2.41 | 0.0824 | 1.54 |
| **RDPM** | 0.4206 | 2.25 | 0.3181 | 1.95 | 0.3390 | 2.14 | 0.0647 | 1.29 |
| **HER** | 0.4059 | 2.30 | 0.3599 | 2.35 | 0.2484 | 1.95 | 0.0676 | 1.31 |
| **LCPO** | 0.4235 | 2.60 | 0.3328 | 2.44 | 0.3527 | 2.51 | 0.1088 | 1.72 |
| **DDR-IG** | **0.5000** | **3.14** | **0.3913** | **2.71** | **0.4412** | **2.91** | **0.1794** | **2.04** |
| **DDR-SE** | **0.4824** | **2.90** | **0.3639** | **2.52** | **0.3711** | 2.50 | **0.1265** | **1.75** |

Table 6: Human evaluation of different agents in different domains. The metrics for human evaluation (SR and AS) are described in Section 4.1.

## 5 Conclusion and Future Work

This paper identifies the dead-end as a critical source for invalid exploration of task-oriented dialogue policies and proposes a novel dead-end detection and rescue method (DDR). The method effectively rescues conversations from dead-ends. Dead-end detection in DDR introduces an efficient criterion for detecting dead-ends, and its effectiveness has been experimentally verified. Once a certain dead-end is detected, rescue in DDR provides rescue exploration guidance for avoiding invalid exploration and thus generates more high-quality and diverse conversations. In addition, the experience containing dead-ends is added for DDA to prevent the same mistakes from occurring in the dialogue policy. The extensive results demonstrate the superiority and generality of our approach.

Although our DDR algorithm is efficient and very general to be used with different RL algorithms on TOD systems, it has two main limitations: 1) our method is challenging to use for other types of dialogue systems where the database is not accessible. Even though this paper focuses on TOD systems that work on the database, it is undeniable that other types of dialogue systems, such as chatbots or recommendation systems, may also have dead-end problems. Future efforts will extend our method to accommodate these inaccessible database scenarios. 2) The scope of dead-end rescue addressed in this paper pertains to the category of dead-ends resolvable by dialogue policies, while our defined dead-end concept can generally cover all dead-end situations. It struggles to handle dead-ends arising from reasons beyond dialogue policies, such as cases where user requests cannot be fulfilled due to absence from the database. While our DDR techniques effectively mitigate many dead-ends, some persist, as illustrated in Table 5. Future research aims to bolster the system's ability to handle these situations more effectively.

## Acknowledgments

We sincerely thank the reviewers for their valuable feedback and constructive comments, which greatly improved the paper. We also acknowledge the volunteers from the South China University of Technology who assisted with our human experiments. This work was supported by Hunan Provincial Natural Science Foundation of China (grant no. 2024JJ6062).

## References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. *NeurIPS*, 30.

Liliana Ardissono, Guido Boella, and Rossana Damiano. 1998. A plan-based model of misunderstandings in cooperative dialogue. *International Journal of Human-Computer Studies*, 48(5):649–679. `https://doi.org/10.1006/ijhc.1997.0185`

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *EMNLP*, pages 5016–5026. `https://doi.org/10.18653/v1/D18-1547`

Trung H. Bui, Martin Rajman, and Miroslav Melichar. 2004. Rapid dialogue prototyping methodology. In *Text, Speech and Dialogue, 7th International Conference, TSD 2004, Brno, Czech Republic, September 8–11, 2004, Proceedings*, volume 3206 of *Lecture Notes in Computer Science*, pages 579–586. Springer. `https://doi.org/10.1007/978-3-540-30120-2_73`

Yan Cao, Keting Lu, Xiaoping Chen, and Shiqi Zhang. 2020. Adaptive dialog policy learning with hindsight and user modeling. *arXiv preprint arXiv:2005.03299*. `https://doi.org/10.18653/v1/2020.sigdial-1.40`

Lu Chen, Runzhe Yang, Cheng Chang, Zihao Ye, Xiang Zhou, and Kai Yu. 2017a. On-line dialogue policy learning with companion teaching. In *EACL*, pages 198–204. `https://doi.org/10.18653/v1/E17-2032`

Lu Chen, Xiang Zhou, Cheng Chang, Runzhe Yang, and Kai Yu. 2017b. Agent-aware dropout dqn for safe and efficient on-line dialogue policy learning. In *EMNLP*, pages 2454–2464. https://doi.org/10.18653/v1/D17-1260

Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. 2021. Unified conversational recommendation policy learning via graph-based reinforcement learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11–15, 2021*, pages 1431–1441. ACM. https://doi.org/10.1145/3404835.3462913

G. William Farthing. 1992. *The Psychology of Consciousness*. Prentice-Hall, Inc.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. In *Proceedings of the SIGDIAL 2016 Conference, The 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 13–15 September 2016, Los Angeles, CA, USA*, pages 101–110. The Association for Computer Linguistics. https://doi.org/10.18653/v1/W16-3613

Mehdi Fatemi, Shikhar Sharma, Harm van Seijen, and Samira Ebrahimi Kahou. 2019. Dead-ends and secure exploration in reinforcement learning. In *ICML*, volume 97.

Christian Geishauser, Songbo Hu, Hsien-chin Lin, Nurul Lubis, Michael Heck, Shutong Feng, Carel van Niekerk, and Milica Gašić. 2021. What does the user want? Information gain for hierarchical dialogue policy optimisation. In *ASRU*. https://doi.org/10.1109/ASRU51503.2021.9687856

Alexander Irpan, Kanishka Rao, Konstantinos Bousmalis, Chris Harris, Julian Ibarz, and Sergey Levine. 2019. Off-policy evaluation via off-policy classification. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8–14, 2019, Vancouver, BC, Canada*, pages 5438–5449.

Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285. https://doi.org/10.1613/jair.301

John T. Kent. 1983. Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173. https://doi.org/10.1093/biomet/70.1.163

Heeseon Kim, Hyeongjun Yang, and Kyong-Ho Lee. 2023. Confident action decision via hierarchical policy learning for conversational recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 – 4 May 2023*, pages 1386–1395. ACM. https://doi.org/10.1145/3543507.3583536

Andrey Kolobov, Mausam, and Daniel S. Weld. 2012. A theory of goal-oriented mdps with dead ends. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, August 14–18, 2012*, pages 438–447. AUAI Press.

Wai-Chung Kwan, Hongru Wang, Huimin Wang, and Kam-Fai Wong. 2022. A survey on recent advances and challenges in reinforcement learning methods for task-oriented dialogue policy learning. *arXiv preprint arXiv:2202.13675*.

T. L. Lai and Herbert Robbins. 1985. Asymptotically efficient adaptive allocation rules. *Advanced in Applied Mathematics*, vol. 6. https://doi.org/10.1016/0196-8858(85)90002-8

Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive path reasoning on graph for conversational recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23–27, 2020*, pages 2073–2083. ACM. https://doi.org/10.1145/3394486.3403258

Mike Lewis, Denis Yarats, Yann N. Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? End-to-end learning for negotiation dialogues. *CoRR*, abs/1706.05125. https://doi.org/10.18653/v1/D17-1259

Xiujun Li, Zachary C. Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

Xiujun Li, Yu Wang, Siqi Sun, Sarah Panda, Jingjing Liu, and Jianfeng Gao.

2018. Microsoft dialogue challenge: Building end-to-end task-completion dialogue systems. *arXiv preprint arXiv:1807.11125*.

Yunhao Li, Yunyi Yang, Xiaojun Quan, and Jianxing Yu. 2021. Retrieve & memorize: Dialog policy learning with multi-action memory. In *Findings of the ACL*, pages 447–459. https://doi.org/10.18653/v1/2021.findings-acl.39

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Sihong Liu, Jinchao Zhang, Keqing He, Weiran Xu, and Jie Zhou. 2021. Scheduled dialog policy learning: An automatic curriculum learning framework for task-oriented dialog system. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1091–1102. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.findings-acl.94

Keting Lu, Shiqi Zhang, and Xiaoping Chen. 2019. Goal-oriented dialogue policy learning from failures. In *AAAI*, volume 33, pages 2596–2603. https://doi.org/10.1609/aaai.v33i01.33012596

Bernd Ludwig. 2006. Tracing actions helps in understanding interactions. In *Proceedings of the SIGDIAL 2006 Workshop, The 7th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 15–16 July 2006, Sydney, Australia*, pages 60–67. The Association for Computer Linguistics. https://doi.org/10.3115/1654595.1654609

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. https://doi.org/10.1038/nature14236

Atsumoto Ohashi and Ryuichiro Higashinaka. 2022. Post-processing networks: Method for optimizing pipeline task-oriented dialogue systems using reinforcement learning. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2022, Edinburgh, UK, 07–09 September 2022*, pages 1–13. Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.sigdial-1.1

Aishwarya Padmakumar and Raymond J. Mooney. 2021. Dialog policy learning for joint clarification and active learning queries. In *AAAI*, volume 35, pages 13604–13612. https://doi.org/10.1609/aaai.v35i15.17604

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Yun-Nung Chen, and Kam-Fai Wong. 2018. Adversarial advantage actor-critic model for task-completion dialogue policy learning. In *ICASSP*, pages 6149–6153. https://doi.org/10.1109/ICASSP.2018.8461918

Mahdin Rohmatillah and Jen-Tzung Chien. 2023. Hierarchical reinforcement learning with guidance for multi-domain dialogue policy. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 31:748–761. https://doi.org/10.1109/TASLP.2023.3235202

Richard S. Sutton. 1995. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *NeurIPS*, 8.

Ryuichi Takanobu, Runze Liang, and Minlie Huang. 2020. Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. In *ACL*, pages 625–638. https://doi.org/10.18653/v1/2020.acl-main.59

Da Tang, Xiujun Li, Jianfeng Gao, Chong Wang, Lihong Li, and Tony Jebara. 2018. Subgoal discovery for hierarchical dialogue policy learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 – November 4, 2018*, pages 2298–2309. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1253

Milene Santos Teixeira and Mauro Dragoni. 2022. A review of plan-based approaches for dialogue management. *Cognitive Computing*, 14(3):1019–1038. https://doi.org/10.1007/s12559-022-09996-0

Hado van Hasselt, Arthur Guez, and David Silver. 2015. Deep reinforcement learning with double q-learning. *CoRR*, abs/1509.06461.

Huimin Wang, Baolin Peng, and Kam-Fai Wong. 2020. Learning efficient dialogue policy from demonstrations through shaping. In *ACL*, pages 6355–6365. https://doi.org/10.18653/v1/2020.acl-main.566

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19–24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1995–2003. JMLR.org.

Julia White, Gabriel Poesia, Robert X. D. Hawkins, Dorsa Sadigh, and Noah D. Goodman. 2021. Open-domain clarification question generation without question examples. In *EMNLP*, pages 563–570. https://doi.org/10.18653/v1/2021.emnlp-main.44

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256. https://doi.org/10.1007/BF00992696

Guanlin Wu, Wenqi Fang, Ji Wang, Jiang Cao, Weidong Bao, Yang Ping, Xiaomin Zhu, and Zheng Wang. 2021. Gaussian process based deep dyna-q approach for dialogue policy learning. In *Findings of the ACL*, pages 1786–1795. https://doi.org/10.18653/v1/2021.findings-acl.156

Yen-Chen Wu and Carl Edward Rasmussen. 2021. Clipping loops for sample-efficient dialogue policy optimisation. In *NAACL*, pages 3420–3428. https://doi.org/10.18653/v1/2021.naacl-main.267

Yen-Chen Wu, Bo-Hsiang Tseng, and Carl Edward Rasmussen. 2019. Tam: Using trainable-action-mask to improve sample-efficiency in reinforcement learning for dialogue systems. In *NeurIPS*.

Tiancheng Zhao, Kaige Xie, and Maxine Eskénazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers)*, pages 1208–1218. Association for Computational Linguistics.

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 *AAAI*, pages 9676–9684. https://doi.org/10.1609/aaai.v34i05.6516

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 *EMNLP*, pages 4311–4323. https://doi.org/10.18653/v1/2021.emnlp-main.354

## A The Impact of Different *dead-end management threshold* on RDPM

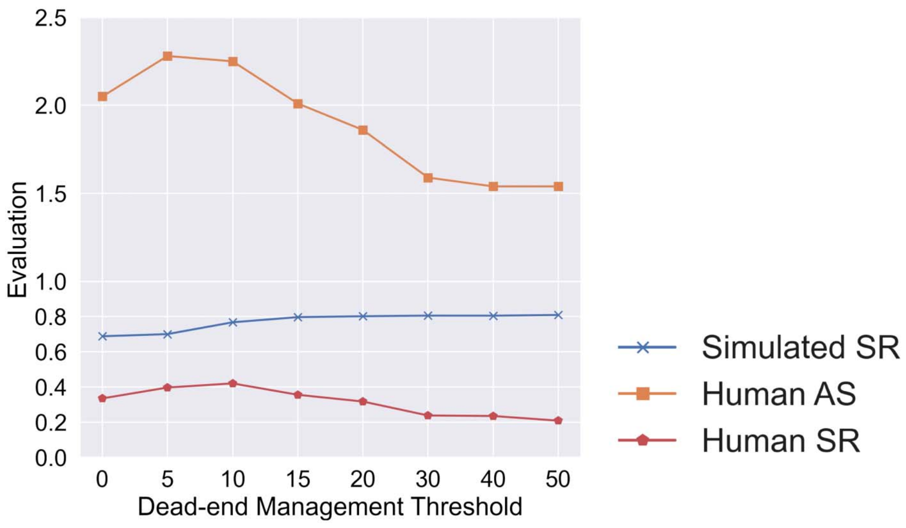
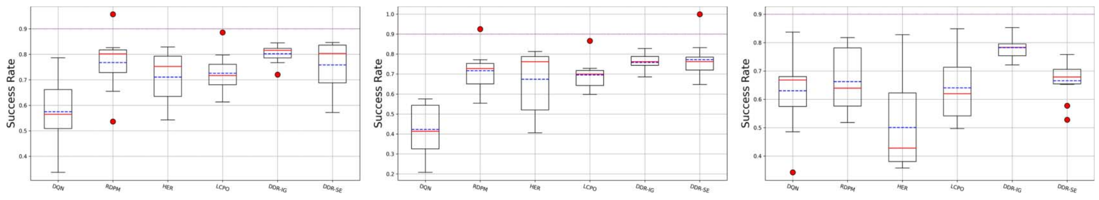


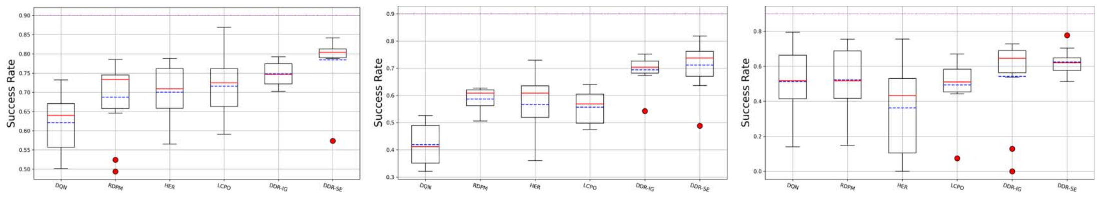
Figure 10: Results of PDRM with different dead-end management thresholds.

## B Robustness Evaluation under All Error Rates



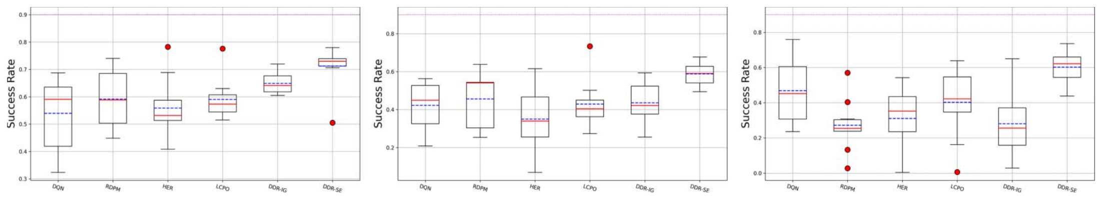
(a) Movie (b) Restaurant (c) Taxi

Figure 11: Results of different agents on three datasets in environment with 0% noise.



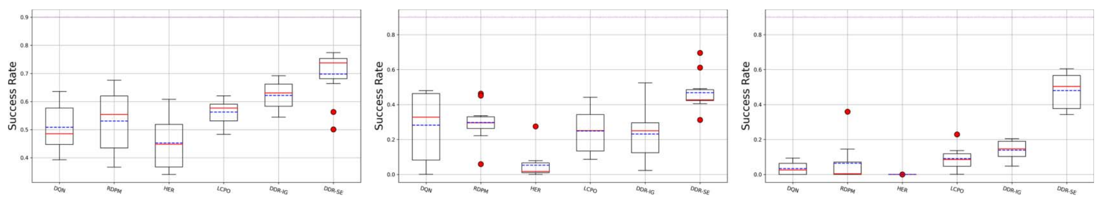
(a) Movie (b) Restaurant (c) Taxi

Figure 12: Results of different agents on three datasets in environment with 5% noise.



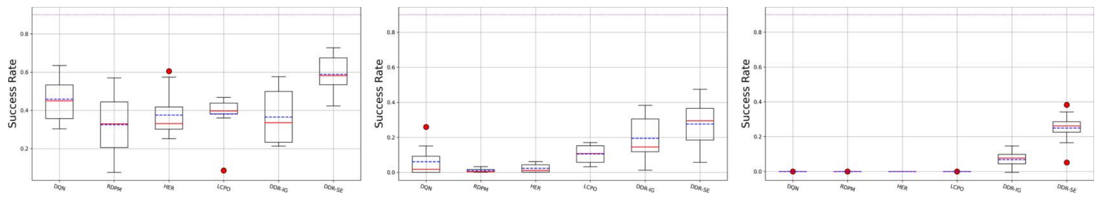
(a) Movie (b) Restaurant (c) Taxi

Figure 13: Results of different agents on three datasets in environment with 10% noise.



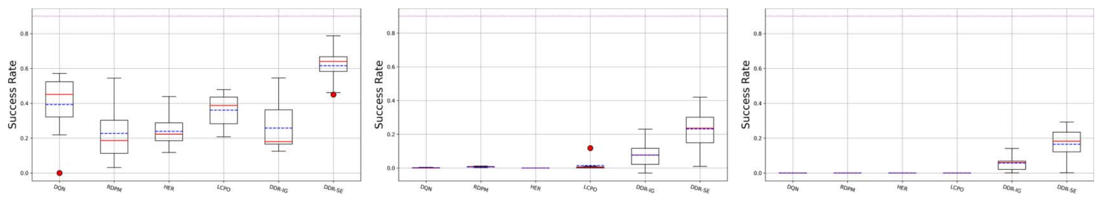
(a) Movie (b) Restaurant (c) Taxi

Figure 14: Results of different agents on three datasets in environment with 15% noise.



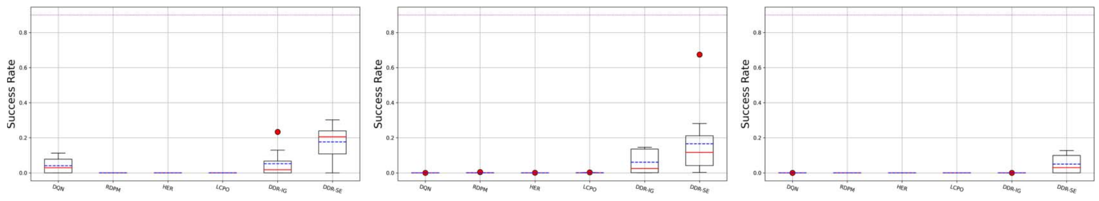
(a) Movie (b) Restaurant (c) Taxi

Figure 15: Results of different agents on three datasets in environment with 20% noise.



(a) Movie (b) Restaurant (c) Taxi

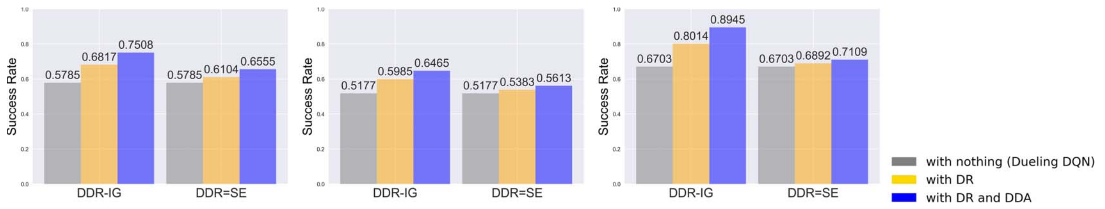
Figure 16: Results of different agents on three datasets in environment with 25% noise.



(a) Movie (b) Restaurant (c) Taxi

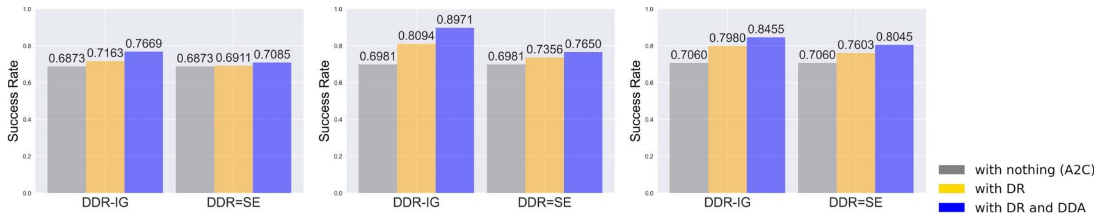
Figure 17: Results of different agents on three datasets in environment with 30% noise.

## C Ablation Evaluation based on Other Methods



(a) Movie (b) Restaurant (c) Taxi

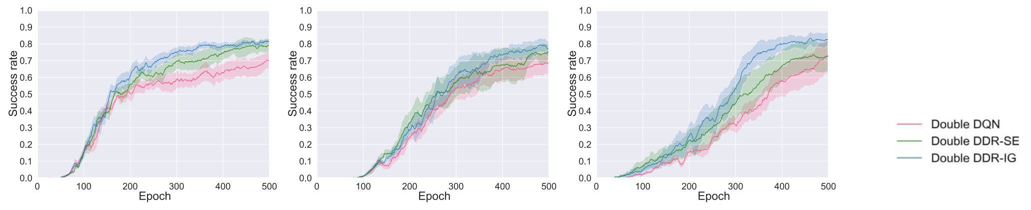
Figure 18: The ablation experiment of Dueling DDR-* methods in Movie, Restaurant, and Taxi domains.



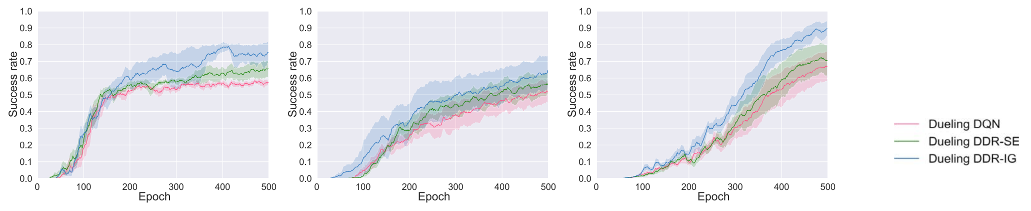
(a) Movie (b) Restaurant (c) Taxi

Figure 19: The ablation experiment of A2C DDR-* methods in Movie, Restaurant, and Taxi domains.

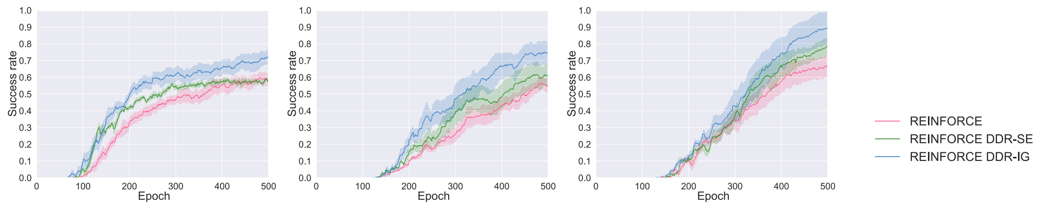
## D Generality Evaluation



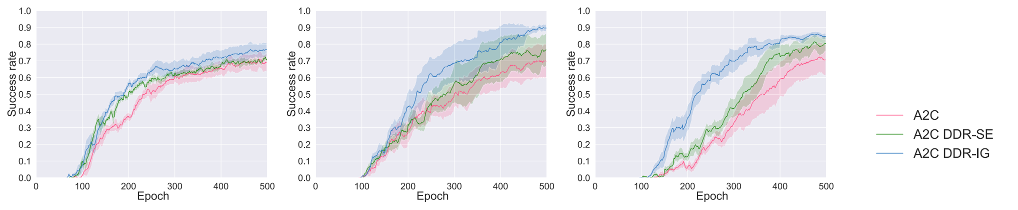
(a) Movie (b) Restaurant (c) Taxi



(d) Movie (e) Restaurant (f) Taxi



(g) Movie (h) Restaurant (i) Taxi



(j) Movie (k) Restaurant (l) Taxi

Figure 20: Incorporating different RL algorithms (Double DQN (van Hasselt et al., 2015), Dueling DQN (Wang et al., 2016), REINFORCE (Williams, 1992), A2C (Fatemi et al., 2016)) with our two methods on three datasets.