

Optimizing dialog policy with large action spaces using deep reinforcement learning

Manisha Thakkar, Nitin Pise

Department of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, India

Article Info

Article history:

Received Feb 23, 2024

Revised May 19, 2024

Accepted Jun 5, 2024

Keywords:

Deep reinforcement learning

Dialogue policy

Imitation learning

Proximal policy optimization

Task-oriented dialogue system

ABSTRACT

Dialogue policy is responsible to select the next appropriate action from the current dialogue state to accomplish the user goal efficiently. Present commercial task-oriented dialogue systems are mostly rule-based; thus, they are not easily scalable to adapt multiple domains. To design an adaptive dialogue policy, user feedback is an essential parameter. Recently, deep reinforcement learning algorithms have been popularly applied to such problems. However, managing large state-action space is time consuming and computationally expensive. Additionally, it requires good quality and a reliable user simulator to train the dialogue policy which takes additional design efforts. In this paper, we propose a novel approach to improve the performance of dialogue policy by accelerating the training process by using imitation learning for deep reinforcement learning. We utilized proximal policy optimization (PPO) algorithm to model dialogue policy using a large-scale multi-domain tourist dataset MultiWOZ2.1. We observed a remarkable performance of dialogue policy with 91.8% task success rate, and an approximate 50% decrease in the average number of turns required to complete tasks without using user simulator in the early phase of training cycles. This approach is expected to help researchers to design computationally efficient and scalable dialogue agents by avoiding training from scratch.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Manisha Thakkar

Department of Computer Engineering and Technology

Dr. Vishwanath Karad MIT World Peace University

Pune, India

Email: manishathakkar@gmail.com

1. INTRODUCTION

Recently, dialogue systems have become an integral part of human life, and they are a vibrant field of ongoing research. These systems are broadly classified into chit-chat systems and task-oriented dialogue (TOD) systems. Chit-chat systems, also known as open-domain dialogue systems, typically do not have a specific goal to complete; these systems converse on any arbitrary topic with the user. On the other hand, TOD systems are grounded in an ontology and help users accomplish specific goals in minimal dialogue turns by proactively generating questions. For instance, finding a restaurant, reserving a taxi, booking a hotel room, and navigating to a certain destination. In our research, we explored TOD systems to enhance their capabilities.

Users often find themselves unsure about their choices, requiring TOD systems to engage in multiple dialogue turns to understand and identify user intention from natural language input. Often, commercial TOD systems are designed in a pipeline approach due to the ease of implementation and higher reliability. In the traditional TOD pipeline, the first component is natural language understanding (NLU),

There are different approaches to designing dialogue policies, which include rule-based, machine learning and deep learning-based approaches. Currently, commercial TOD systems use handcrafted rules to select the next action in dialogue policy by designing a fixed dialogue flow [3]. Although rule-based dialogue policies perform well, they are not scalable enough to support new tasks or domains easily. The dialogue flow rules must be redesigned with the help of domain experts to meet the quality and correctness requirements, which is tedious and costly. Also, present TOD systems are mostly single domain with limited task coverage in the domain under consideration [4]. To design TOD system adaptive to multi-domain scenarios, several researchers have explored supervised machine learning to design dialogue policy by approaching it as a multi-class classification problem. This approach requires large amounts of annotated data for training which is not feasible for many practical scenarios. Additionally, creating such data is tedious and costly. Also, supervised approach is not suitable for dynamic environments where user feedback is an essential parameter for the dialogue agent to understand and improve upon experience.

Reinforcement learning (RL) is the preferred choice for modeling dialogue policy in TOD systems to increase the generalization capability by taking the user conversation towards success [5]. In RL-based dialogue systems, user feedback is considered in terms of positive or negative rewards for the actions chosen by the dialogue agent, similar to the gaming environment. As depicted in Figure 3, the RL-based TOD systems are trained in a user-agent setting using a trial-and-error method, and the dialogue agent improves its performance from earlier experience and tries to maximize its rewards.

It is challenging to use RL algorithms in practice because of the following reasons: i) the requirement of large computing power; ii) the complexity of handling continuous sample space; iii) the requirement of a good quality and reliable user simulator for training RL-based agent; and iv) the large unexplored territories and sparse rewards in the RL approach. With deep learning advancements, neural approaches have been combined with RL (DRL) to manage large sample spaces efficiently. DRL algorithms are categorized into model-free and model-based algorithms. In model-free RL algorithms, a dialogue agent employs a trial-and-error method to learn the optimal action selection by using either policy iteration methods or value-iteration methods. The value-iteration methods estimate the value, i.e., expected rewards of being in the given state, by constructing a Q-function and greedily searching for the best policy. On the other hand, policy-iteration methods search for the optimal policy directly by adjusting the parameters of the policy using gradient updates to maximize rewards. In the hybrid approach, the actor-critic method combines both value and policy iteration methods, in which the actor (policy-function) learns from the feedback given by the critic (value-function). Algorithms such as the temporal difference (TD), deep Q-network (DQN), Monte Carlo algorithm, and C51 utilize value-based approaches, and algorithms based on policy iteration approaches include policy gradient (PG) and actor-critic structure-based algorithms such as the phasic policy gradient (PPG), trust region policy optimization (TRPO), proximal policy optimization (PPO), asynchronous advantage actor-critics (A3C), and actor-critics with experience replay (ACER) [6].

Model-based algorithms create a simulated model for each condition to learn and predict compact policies without runtime optimization. The few model-based algorithms are imagination-augmented agents (I2A), model-based priors for model-free (MBMF), and model-based value expansion (MBVE) [7]. The integration of model-free and model-based methods is also emerging, the Dyna-Q algorithm has its place in this category. In this paper, we focused on model-free algorithms for the survey of dialogue policy. After the remarkable success of DRL in simulated gaming environments such as the Atari, chess game go and other robotic tasks [8], various DRL algorithms have been studied for modeling dialogue policy [6], [9]–[11]. Q-learning-based algorithms are widely studied in single-domain dialogue scenarios [12], [13], and it is observed that tabular-based algorithms such as DQNs [14], double DQNs [15], and dueling DQNs [16], which utilize greedy exploration strategies, are not easily scalable for handling continuous action spaces of multiple domains; these algorithms are more suitable for limited action spaces [17].

Training RL-based dialogue agent by awarding rewards or assigning penalty is often time-consuming as rewards are often given manually, [18] studied dialogue policy learning using a hybrid approach, initially they used supervised learning to minimize the cross-entropy losses for system action prediction. In a later stage, the REINFORCE algorithm [19] was applied to optimize the RL agent to improve from the rewards received. A positive reward of 15 for successful completion, zero reward is assigned in case of failure, and a small penalty of -1 is assigned to motivate the dialogue agent to complete the task faster. Zhao *et al.* [20] proposed a latent action framework in which action spaces are treated as latent variables, and they performed unsupervised learning to capture the action space. In their work, if the user target returned by the user simulator appears in the top 5 results (i.e., $R=5$), the dialogue is treated as successful, and the reward is calculated as actual rank of the target as, $\max(0, 2(1-(r-1)/R))$, where r is the actual rank of the target. In the case of failure, the agent receives a negative reward of -1. The maximum length of dialogue allowed is 10; if the result is more than the maximum length, it is treated as a failure. To motivate the agent to complete quickly, a negative reward of -0.1 is assigned at each turn. Peng *et al.* [21] studied the hierarchical RL

algorithm, in which a top policy focuses on higher-level goals, while sub-policies focus on smaller goals responsible for fine control. They assigned a positive reward of $2 \times \text{max}$ turns for success, a negative reward of $-\text{max}$ turns for failure and a small negative reward of -1 at the end of each turn to speedy completion.

In their work, [13], [22] used the deep Dyna-Q (DDQ) algorithm by using DQN with a positive reward of $2 \times \text{maximum}$ turns for success, a negative reward of $-\text{maximum}$ turns for failure, and a negative reward of -1 at each turn for fast task convergence. In the work proposed by [23], variational auto-encoders are used to optimize dialogue policy and generate responses in an end-to-end setting. Several policy-iteration algorithms, such as the TRPO, PPO, and ACER algorithms, have been studied for continuous action spaces in TOD systems and are active areas of research. Weisz *et al.* [24] studied the ACER algorithm for optimizing dialogue policy in the restaurant domain and used rewards at the dialogue level as follows, $20 \times \text{maxturns}$ for a successful dialogue with maximum turns limited to 25. Ohashi and Higashinaka [25] utilized PPO policy to improve the performance of dialogue policy by performing post processing of the output of each module in the pipeline architecture. They assigned positive rewards of $2 \times \text{maximum}$ turns where maximum turns are allowed until 20, and a small negative reward of -1 is assigned to speed up the dialogue agent.

It is observed that, in many research studies rewards are given manually only after successful completion of the entire dialogue, and no rewards are assigned for the completion of sub-tasks towards achieving final goal. To learn from scratch by observing the environment, user simulators are often used to train RL-based dialogue agents that interact on behalf of real user to achieve stable dialogue policy as real users may feel frustrating to train these agents [26]–[28]. Most of the time, user simulators are handcrafted; they use a stack-like data structure (agenda) for maintaining goals as slot-value pairs. The quality of the user simulator impacts the performance of the dialogue policy. It is time consuming to develop a user simulator for each domain. Many researchers have focused on developing dialogue systems and used existing user simulators. As a result of learning by the trial-and-error approach, the user simulator and the dialogue agent generate thousands of poor-performing interactions, specifically in the initial phase of training. This process of learning from scratch requires considerable time for a dialogue agent to establish a stable policy; additionally, it increases the computational load.

To address these issues, inverse reinforcement learning (IRL) and imitation learning (IL) approaches have been studied. Li *et al.* [29] and Takanobu *et al.* [30] studied the IRL for precise reward signals; this approach infers from exerting behavior and generating rewards to optimize the reward function. However, IL directly facilitates the learning of complex tasks with minimal knowledge by mimicking expert demonstrations without any explicit reward generation [31]. In our work, two research objectives are as follows: i) to design an adaptive dialogue policy for a large continuous sample space in multi-domain environment using the DRL approach and ii) to speed up the learning of a dialogue agent without using user simulators in the initial phase of training by leveraging IL for the DRL-based dialogue agent. We used a two-step approach. Initially, we applied IL to mimic expert behavior in supervised manners without learning from scratch or through sparse and manually given rewards. In the later step, we use the PPO (proximal policy optimization) algorithm, which is a type of model-free DRL algorithm, to select the optimal system action from the current dialogue state in a dynamic environment. Finally, we evaluate the performance of the dialogue policy in a multi-domain setting. The remainder of the paper is arranged as follows: the methodology section describes the formulation of dialogue policy as a Markov decision process using PPO objective function, proposed two-step algorithm to use IL with PPO, details of the large scale multi-domain dataset, toolkit and model parameters used for experiment setup; further, the result and discussion section describes the comparative analysis of results obtained; and finally, the conclusion section summarizes all the findings, discuss the limitations and future research directions.

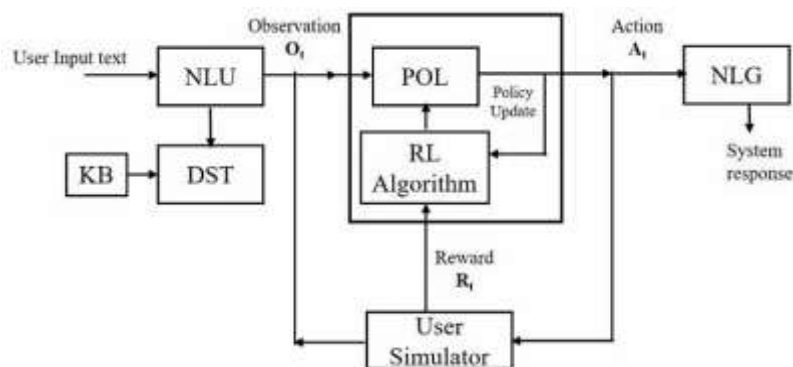


Figure 3. RL-based POL in a TOD system

2. METHOD

RL algorithms have been successfully applied in deciding the next action in simulated gaming environments for playing Atari games [8], playing chess games [32], and various robotic tasks. Similarly, dialogue policy learning is treated as a sequence decision-making problem and formulated as an MDP (Markov decision process) [33], [34]. The RL methodology offers a robust framework for training dialogue agents within dynamic environments, utilizing feedback in the form of rewards to enhance learning. In the succeeding sub-section, we detail the formulation of a markov decision process (MDP) designed specifically for RL-based dialogue policy learning.

2.1. Learning dialogue policies as markov decision processes

A multi-domain dialogue (MDD) corpus with a collection of human dialogues that may be single or multiple turns spanning across domains is represented as $MDD = \{T1, T2, T3, T4, \dots\}$, where T_i represents a trajectory of state and action pairs $\{s_{usr}^0, a_{usr}^0, s_{sys}^0, a_{sys}^0, s_{usr}^1, a_{usr}^1, s_{sys}^1, a_{sys}^1, \dots\}$, where (s_{usr}, a_{usr}) represents the user's state-action and (s_{sys}, a_{sys}) denotes the system's state-action. A trajectory is generated by sampling actions according to the dialogue policy formulated as a partially observable Markov decision process (POMDP) with tuples $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho_0)$, where \mathcal{S} denotes the set of states, \mathcal{A} denotes the set of system actions, and $\mathcal{P}: \mathcal{S} \times \mathcal{A}$ is the transition probability function given as $(s_{t+1} | s_t, a_t)$, where state $s_t \in \mathcal{S}$ and action $a_t \in \mathcal{A}$ at time step t . The expected immediate reward function is represented as $\mathcal{R}(s_t, a_t)$, and the discount factor is denoted as $\gamma \in (0, 1)$, which is used to represent the effect of future rewards. The initial state distribution is denoted as ρ_0 in the tuple.

An optimal policy implies that the dialogue agent will take the most accurate action to maximize the cumulative reward \mathcal{R}_t when the context, i.e., the current dialogue state along with the dialogue history, is given to it [33]. The policy gradient algorithm stochastically samples appropriate action and adjust the policy parameters θ (weights and biases of the neural network) through the gradient ascent method to obtain the maximum cumulative reward [35]. For instance, consider a gradient stochastic policy, as shown in (1), and the cumulative reward \mathcal{R}_t at time step t , as shown in (2).

$$a = \pi(a | s; \theta) \quad (1)$$

$$\mathcal{R}_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}, r_t = \mathcal{R}(s_t, a_t) \quad (2)$$

The objective is to learn an optimal dialogue policy π^* to adapt in multi-domain setting using the DRL algorithm to maximize cumulative rewards. The loss function is defined as shown in (3).

$$\mathcal{L}(\pi) = \mathbb{E}_{r \sim \pi} [R_t | \pi(\cdot; \theta)] \quad (3)$$

The state-value function of π is defined as:

$$\mathcal{V}^\pi(s) = \mathbb{E}_{r \sim \pi} [R_t | s_0 = s] \quad (4)$$

The state-action value function of π is given as:

$$\mathcal{Q}^\pi(s, a) = \mathbb{E}_{r \sim \pi} [R_t | s_0 = s, a_0 = a] \quad (5)$$

Using (4) and (5), the advantage function of π is given as:

$$\mathcal{A}^\pi(s, a) = \mathcal{Q}^\pi(s, a) - \mathcal{V}^\pi(s) \quad (6)$$

Whether the action is better or bad than the policy's default behavior is measured by the advantage function $\mathcal{A}^\pi(s, a)$, as shown in (6). Therefore, we need to choose the gradient term $\pi_t \nabla_{\theta} \log \pi_{\theta}(a_t | s_t)$ to point in the increased direction of $\pi_{\theta}(a_t | s_t)$ which implies, $\mathcal{A}^\pi(s_t, a_t) > 0$

Designing a reliable user simulator is not trivial; it is as complex as building a good dialog agent. We propose an efficient two-step approach for modeling adaptive dialogue policy in a multi-domain environment. To reduce the RL training duration and computational overhead, we use imitation learning (IL) using behavior cloning (BC) to learn the optimal policy by mimicking expert demonstrations directly in a supervised manner, and no explicit rewards are generated in this step. Given the database of expert trajectories as, $D_{\text{Expert trajectories}} = \{ \text{state}_n, \text{action}_n, \text{reward}_n, \text{next_state}_n \}_{n=1}^N$, the dialogue agent acquires the state-action pairs from \mathcal{D} ; then, a regressor or classifier is estimated to mimic the expert behavior as

described in (7). The policy network $\pi_{\theta,old}(\cdot)$ is trained by minimizing the mean squared error or cross entropy error and calculates advantage estimates \mathcal{A}^π by calculating rewards.

$$\theta^* = arg \min_{\theta} \sum_{i=1}^N L_{BC} (\pi_{\theta} (state_i, action_i)) \tag{7}$$

This no-scratch learning approach does not learn from sparse rewards or manually assigned rewards, but it observes and mimics experts’ demonstrations. This approach mitigates the need for large amounts of data for adapting to new tasks or domains. The IL approach accelerates the RL training process by serving as a warm-up. In the second step, the best performing policy weights from BC are used as pretrained weights. In this step, we utilized the policy gradient algorithm PPO using a constant clipping mechanism as the soft constraint for optimization [36]. PPO is an on-policy reinforcement learning algorithm based on the trust region policy optimization (TRPO) [36] algorithm with strong performance and simple implementation compared to TRPO. This process requires user feedback and ensures that the policy update at each epoch is not too large, increasing the training stability. To achieve the same, the ratio of the current and next policy is represented as $r_t(\theta)$, as defined in (8).

$$r_t(\theta) = \left(\frac{\pi_{next}(a|s)}{\pi_c(a|s)} \right) \tag{8}$$

Then, clipping is applied to this ratio to alter the range $[1 - \epsilon, 1 + \epsilon]$ to ensure that policy updates occur within this range only. Thus, the goal of the PPO algorithm is to maximize the surrogate objective while placing the constraint that the next policy is close to the current policy. This is accomplished by using the following clipped surrogate objective function objective at every policy update, as shown in (9).

$$\mathcal{L}_c^{PPO}(\pi) = \mathbb{E}_{r \sim \pi_c} [\min (r_t(\theta)) \mathcal{A}^{\pi_c} (s, a), \text{clip} (r_t(\theta), 1 - \epsilon, 1 + \epsilon) \mathcal{A}^{\pi_c} (s, a))] \tag{9}$$

In the second term of the objective function in (9), PPO is designed to limit the discrepancy between successive policies by negating the potential for the probability ratio $\left(\frac{\pi_{next}(a|s)}{\pi_c(a|s)} \right)$ to surpass the boundaries of the clipping range $[1-\epsilon, 1+\epsilon]$. Here, ϵ is a hyper-parameter that helps us to define this clip range. The following subsection describes the algorithm for modeling DRL-based dialogue policy by utilizing IL with the BC approach.

2.2. Algorithm

In this sub-section, we explore a detailed algorithm designed to enhance dialogue policy, leveraging PPO as the core reinforcement learning method for stable and efficient policy updates within predefined constraints as shown in Algorithm 1. Additionally, the integration of Imitation Learning with PPO aims to empower the agent with enhanced dialogue management capabilities in dynamic environments, combining the robustness of reinforcement learning with valuable insights from human expertise.

2.3. Dataset

A large scale MultiWOZ2.1 dataset, which contains the human-to-human conversational dialogues of a tourist company with its customers is utilized in this study. The dataset is annotated for different TOD tasks, such as dialogue states, system dialogue acts, and user goals, in different domains of trip information settings (hotel, train, hospital, taxi, police, postcode, restaurant) [37], [38]. MultiWOZ2.1 is a comprehensive dataset encompassing more than 3K single-domain dialogues and 7K multi-domain dialogues, with each multi-domain dialogue spanning at least two to five domains. Approximately 70% of the dialogues consist of more than ten turns, introducing complexity and replicating real-time scenarios [39].

2.4. Toolkit

Numerous tools have been incorporated and merged into an integrated development environment (IDE). These tools include rule-based dialogue management (DM) and built-in natural language understanding (NLU) capabilities, but do not provide user simulators for training and evaluation. Among various open-source frameworks, ConvLab [20] is a popular open-source framework in the research community that facilitates various state-of-the-art models for each component of TOD systems, it also supports the end-to-end design. Various datasets and user simulators are available for experimental setup and evaluation of dialogue systems [21]. Recently released ConvLab-3 provides sophisticated RL features within this toolkit, which enables seamless incorporation of RL-based algorithms via the ‘vectorizer’ class, as

depicted in Figure 4. We used an agenda-based user simulator (ABUS) for automatic evaluation of the dialogue policy provided in the toolkit for training in the second step.

```

Require: Multi-domain multi-turn conversation Dataset, User Simulator,
Expert Dataset  $D_{expert\_trajectories} = \{state_n, action_n, reward_n, next\_state_n\}$ 

for iteration in range (1 to N):
    # Collect on-policy data with current policy in
     $D_{expert\_trajectories} = \{state_n, action_n, reward_n, next\_state_n\}$ 

    for actor in range (actors):
         $state_n, action_n, reward_n, next\_state_n = run\_policy(\pi_{\theta_{old}}, environment, T)$ 
        # Calculate advantage estimates
         $\mathcal{A}^{\pi} = advantage(rewards, next\_states, \pi_{\theta_{old}})$ 
        # Combined loss with Imitation Learning using BC:
         $\mathcal{L}_c^{PPO} = 0$ 
    # PPO loss with clipped surrogate objective in current state:
    for  $s, a, \mathcal{A}^{\pi}, s+1$  in (states, actions,  $\mathcal{A}^{\pi}$ , next_states):
        # calculate transition probabilities with current and next state
         $\pi_{new} = \pi_{\theta_{new}}(a | s)$ 
         $\pi_{old} = \pi_{\theta_{old}}(a | s)$ 
        # Find the ratio of old and new probabilities
         $r_t(\theta) = \pi_{new} / (\pi_{old} + \epsilon)$  # avoid divide by zero, add small quantity
        # Use Clipped surrogate objective with KL divergence:
         $kl = clip\_by\_value(r_t(\theta), 1 - \epsilon, 1 + \epsilon) * \mathcal{A}^{\pi}$ 
         $\mathcal{L}_c^{PPO}(\pi) = -minimum(kl, r_t(\theta), * \mathcal{A}^{\pi})$ 
        # BC loss (mean squared error):
         $\mathcal{L}_{BC} = (action - \pi_{\theta_{new}}(action | state))^{*2}$ 
        # Combine losses with adjustable weight:
         $w = weight\_function\_adjust(iteration)$ 
         $\mathcal{L}_{combined} += w * \mathcal{L}_c^{PPO} + (1 - w) * \mathcal{L}_{BC}$ 
        # Update policy with combined loss:
        Optimize ( $\mathcal{L}_{combined}$ ,  $\pi_{\theta_{new}}$ , epochs=K, batch_size=N)
        # Policy update:
         $\pi_{\theta_{old}} = \pi_{\theta_{new}}$  # Replace old policy with updated one
    # Final policy:  $\pi_{\theta_{old}}$ 
    
```

Algorithm 1. Enhancing dialogue policies with PPO and imitation learning

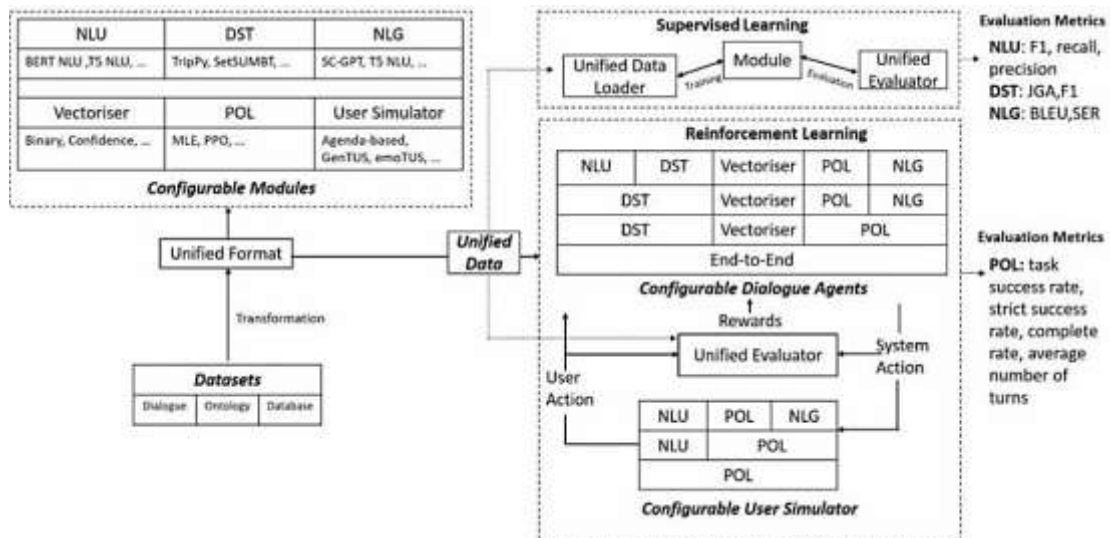


Figure 4. ConvLab-3 framework [40]

2.5. Model parameters

In the first step, we utilized supervised learning to train a multilayer perceptron (MLP) for implementing IL with BC using the maximum likelihood estimator (MLE) policy. The parameter configuration and hyper-parameter settings are passed through a JSON file (config.json) to train the MLE policy, are shown in Table 1. In this step, a classifier is utilized to replicate expert behavior as depicted in Figure 5. The parameter configuration file, database name and seed value are specified as argument for training MLE policy. At this step, no explicit rewards are generated for the dialogue agent.

Table 1. Imitation learning model parameters

Parameter	Value
Batch size	32
Epochs	24
Learning rate	0.0001
Hidden layer size (h_dim)	256
No of hidden layers of size h_dim	100
Path to save Policy	best/MLE

```

python /content/ConvLab-3/convlab/policy/mle/train.py --dataset_name=multiwoz21 --seed=20200202
INFO:root:Visible device: cpu
Visible device: cpu
INFO:root:Seed used: 20200202
Seed used: 20200202
INFO:root:Vectorizer: Data set used is multiwoz21
Vectorizer: Data set used is multiwoz21
Load actions from file..
Dimension of system actions: 288
Dimension of user actions: 79
State dimension: 361
Load processed data file
INFO:root:Start training
Start training
INFO:root:Epoch: 0
Epoch: 0
INFO:root:Precision: 0
Precision: 0
INFO:root:Recall: 0
Recall: 0
INFO:root:<<dialog policy>> epoch 23: saved network to mdl
<<dialog policy>> epoch 23: saved network to mdl
INFO:root:Best Precision: 0.42528588359652775
Best Precision: 0.42528588359652775
INFO:root:Best Recall: 0.7878891257995735
Best Recall: 0.7878891257995735
INFO:root:Best F1: 0.5283194960237989
Best F1: 0.5283194960237989
    
```

Figure 5. Evaluation of MLE policy

The ‘vectorizer’ class interprets the semantic actions that are received by DST and converts them into vectorized representations based on the domain ontology. This class serves as a flexible method for experimenting with various RL strategies for dialogue policy. In the next step, the best performing MLE policy is used as a pretrained model for the PPO algorithm. The PPO policy is trained for 10 epochs to evaluate its adaptability by evaluating the policy after 500 and 1,000 dialogues. The batch size is configured as 32 and hidden layers as 100 with the dimensions of 50. The hyper-parameter settings used for training the PPO dialogue policy using IL and the model configuration parameters are contained in the JSON file as described in Table 2.

Table 2. PPO Dialogue policy model configuration and hyper-parameters

Parameter name	Value
Batch size	32
Gamma	0.99
Epsilon (clipping)	0.2
Learning rate (policy)	0.0001
Learning rate (value)	0.00005
Hidden layer size (h_dim)	50
No. of hidden layers of size h_dim	100
Load path	Path to pretrained model (best/MLE)

We configure the dialogue agent and user simulator, as shown in Figure 6, and pass the configuration file (RuleUser-Semantic-RuleDST.json) as an argument while training PPO policy along with database name and seed value. The path for best performing MLE policy which is found in earlier step is specified using 'load_path' parameter and used as a starting point to train PPO policy instead of training it from scratch. The evaluation takes place after every 500 dialogues, and the evaluation frequency is set to 5.

Inside the 'vectorizer_sys' section within the same configuration file, rule-based DST is specified using the 'dst_sys' parameter, which implies that the POL component receives a dialogue state from DST which is modeled using a rule-based approach. The current dialogue state that also incorporates dialogue history serves as a context for selecting appropriate action in a particular dialogue turn. We utilize an existing user simulator configured using the 'policy_usr' parameter, which employs rule-based policy, i.e., an agenda-based user simulator (ABUS) for automatic evaluation of proposed hybrid dialogue policy. As we are interested in POL performance, other components, such as the NLU and NLG for the dialogue agent and the NLU, DST and NLG for user simulators, are not configured for this specific experiment. However, these parameters can be configured to design and evaluate the overall performance of the dialogue system in a pipeline architecture. The efficacy of IL for modeling scalable multi-domain dialogue policy using PPO algorithm is evaluated using agenda-based user simulator and results are discussed in next section.

```
{
  "args": {"config_name": "RuleUser-Semantic-RuleDST", "seed": 0, "mode": "info", "save_eval_dials": false},
  "config": {"model": {"load_path": "../best/MLE",
    "use_pretrained_initialisation": false, "pretrained_load_path": "",
    "num_train_dialogues": 100, "seed": 0, "epoch": 10, "eval_frequency": 5,
    "process_num": 4, "sys_semantic_to_usr": false, "num_eval_dialogues": 500},
  "vectorizer_sys": {
    "uncertainty_vector_mul": {"class_path": "convlab.policy.vector.vector_binary.VectorBinary",
    "ini_params": {"use_masking": true, "manually_add_entity_names": false, "seed": 0}},
    "nlu_sys": {}, "sys_nlg": {},
    "dst_sys": {"RuleDST": {"class_path": "convlab.dst.rule.multiwoz.dst.RuleDST"}},
    "nlu_usr": {},
    "dst_usr": {},
    "policy_usr": {"RulePolicy": {"class_path": "convlab.policy.rule.multiwoz.RulePolicy",
    "ini_params": {"character": "usr"}}, "usr_nlg": {}}, "policy_config": null}
}
```

Figure 6. Configuration of agent and user simulator

3. RESULTS AND DISCUSSION

We assess the accuracy of identifying expert actions by utilizing the precision (P), recall (R), and F1-score (F1) at initial step. We recorded the best policy with approximate values as follows: P=42%, R=70%, and F1=52% after running 24 epochs with seed values of 0 and 20200202. Further, pretrained weights of MLE policy from its checkpoint are used to train PPO policy and evaluated the performance of dialogue policy with different seed values to validate the results for 5 runs. As demonstrated in Figure 7, we observed a remarkable boost in the success rate, strict success rate and complete rate of dialogue compared to learning the dialogue policy from scratch. As shown in the Figures 7(a) and 7(b), we observed 88% of success rate and strict success rate in first 500 dialogues and approximately 92% after 1,000 dialogues this indicates fast learning of dialogue agent for goal completion directly from expert demonstrations without using user simulator. The complete rate as demonstrated in Figure 7(c) is 88.6% after evaluating the first 500 dialogues and 93.2% after 1,000 dialogues signifies that dialogue agent has improved adaptation to multi-domain tasks by fulfilling more user's requests. The Figure 7(d) indicates a greater than 50% reduction in the average number of turns needed to accomplish the task i.e., fast convergence, it is one of the essential measures as any TOD system is required to accomplish the task with minimum conversation steps for enhanced user experience or else user may quit feeling frustrated. The results indicate manually given rewards and training dialogue agent from scratch can be avoided by using the pretrained model by adopting computationally efficient approach thereby elevating the load during initial learning phase of the dialogue agent. Additionally, dependence on user simulators is reduced. This method may be helpful to train scalable conversational systems in computationally efficient manner.

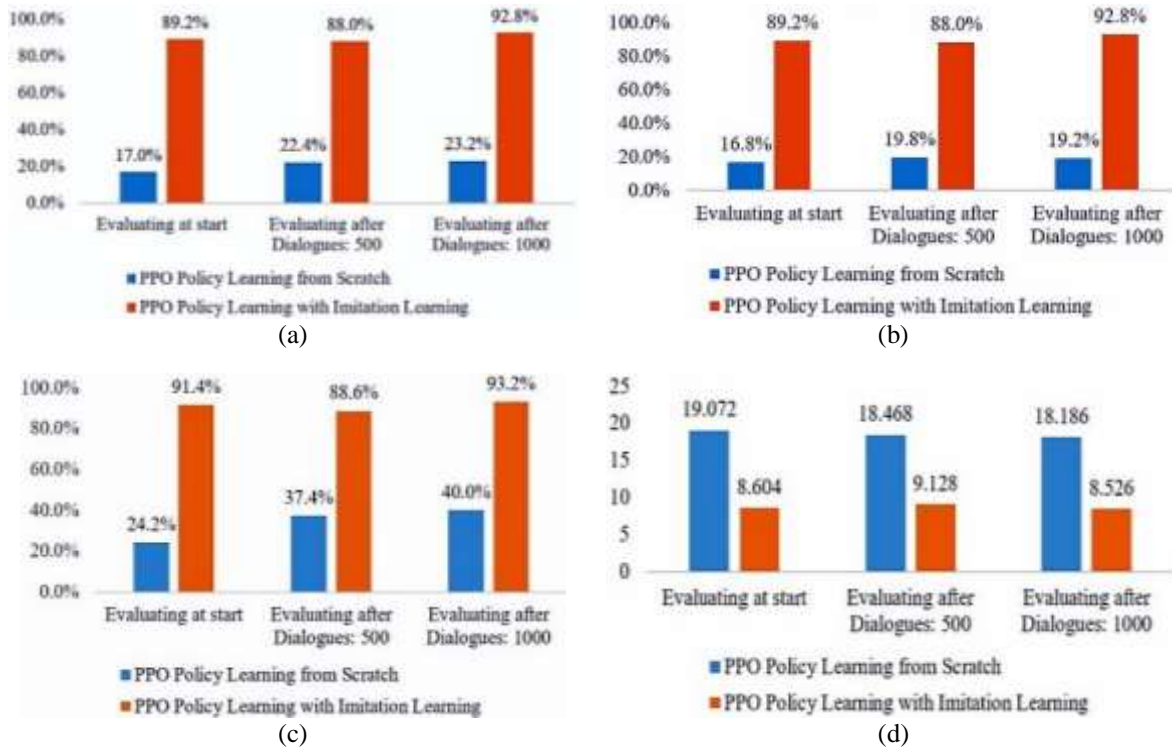


Figure 7. Evaluation of the Dialogue policy: (a) success rate, (b) strict success rate, (c) complete rate, and (d) average number of successful turns

As NLG is not configured, the simulation of user and system agent interaction is demonstrated by capturing changes in user and system actions which are recorded in generated log for each dialogue turn as follows.

```
!python conllab/policy/po/train.py --config_name=RuleUser-Semantic-RuleDST --seed=20200202
```

```
User: [['inform', 'attraction', '', '']]
System: [['inform', 'attraction', 'choice', '10'], ['request', 'attraction', 'area', '?'], ['request', 'attraction', 'type', '?']]
User: [['inform', 'attraction', 'type', 'museum'], ['inform', 'attraction', 'area', 'west'], ['inform', 'attraction', 'choice', 'any']]
System: [['inform', 'attraction', 'area', 'west'], ['inform', 'attraction', 'choice', '7'], ['inform', 'attraction', 'entrance fee', 'free'],
['inform', 'attraction', 'name', 'cafe jello gallery'], ['inform', 'attraction', 'type', 'museum'], ['recommend', 'attraction',
'name', 'cafe jello gallery']]
User: [['request', 'attraction', 'phone', ''], ['request', 'attraction', 'postcode', ''], ['request', 'attraction', 'address', ''], ['request', 'attraction', 'entrance fee', '']]
System: [['inform', 'attraction', 'address', 'cafe jello gallery, 13 magdalene street'], ['inform', 'attraction', 'entrance fee', 'free'],
['inform', 'attraction', 'name', 'cafe jello gallery'], ['inform', 'attraction', 'phone', '01223312112'], ['inform', 'attraction', 'postcode', 'cb30af'], ['reqmore', 'general', 'none', 'none']]
User: [['inform', 'restaurant', '', ''], ['inform', 'restaurant', 'food', 'indian'], ['inform', 'restaurant', 'price range', 'moderate'],
['inform', 'restaurant', 'area', 'west']]
System: [['inform', 'restaurant', 'area', 'west'], ['inform', 'restaurant', 'choice', '1'], ['inform', 'restaurant', 'food', 'indian'],
['inform', 'restaurant', 'name', 'meghna'], ['inform', 'restaurant', 'price range', 'moderate']]
User: [['bye', 'general', '', '']]
System: [['bye', 'general', 'none', 'none'], ['welcome', 'general', 'none', 'none']]
```

We compared performance of POL component using different policy optimization algorithms for the same NLU, DST and NLG in the pipeline. This study utilized MLE, PG, GDPL and PPO which employed behavior cloning (BC). After evaluating 1,000 dialogues, Table 3, demonstrates PPO algorithm is more robust with significant improvement in complete rate and success rate compared to other approaches MLE, PG and GDPL. GDPL algorithm depicted improved convergence speed when compared with PPO, but the observed success rate is not even 50%. MLE and PG algorithms demonstrated poor performances in dialogue completion.

Recently, neural based E2E design of dialogue systems are preferred which jointly optimize more than two or all the components of dialogue system using a shared model. When compared with E2E system performances, significant improvement in complete rate is observed with lesser dialogue turns to accomplish the goal. Though the better success rate is observed in E2E neural approaches using GPT-based models for all TOD tasks, large number of parameters are required, indicating the increased computational overload for

maintaining the common model with shared parameters across all tasks. Therefore, pipeline approach using PPO algorithm can be investigated further for designing scalable systems in scarce resource environment using imitation learning and inverse reinforcement learning approaches.

Table 3. Comparison of proposed approach with other configurations in pipeline and E2E design for multi-domain dialogue system [41], [42]

NLU	DST	Configuration POL	NLG	Complete rate (%)	Success rate (%)	Average no. of success turns	No of parameters in POL model (approx.)
With only dialogue policy component optimized using BC in pipeline architecture							
BERT	Rule	Supervised learning (MLE)	Template	52.6	48.4	12.45	
BERT	Rule	Policy gradient (PG)	Template	47.9	44.3	14.69	
BERT	Rule	Guided dialogue policy learning using adversarial inverse RL (GDPL)	Template	57.9	48.5	11.47	65K for all configurations in pipeline design
BERT	Rule	Proximal policy optimization (PPO)	Template	81.7	76.4	14.1	
All components NLU, DST, NLG optimized jointly using off-policy model free algorithms POL in E2E architecture							
UBAR-distilGPT-2 based model, utilize belief states, database results and generated act and responses as context				74.3	79.8	14.2	82 M
Critic-regularized regression (CRR) - GPT-2 based model, performs weighted behavior cloning of offline dataset				72.6	78.2	13.6	≥124 M
Decision transformer - GPT-2 based model				75.30	81.3	14.81	≥124 M
GPT-critic - GPT-2 based model				77.7	84.3	16.3	≥124 M

4. CONCLUSION

Dialogue policy is an essential component of a dialogue system that determines the flow of a conversation to achieve user goals efficiently. In line with objectives raised, model-free deep reinforcement learning algorithm using PPO demonstrated a promising approach for handling large number of state-action spaces in multi-domain conversation environments. Often dialogue agents are assigned sparse rewards at dialogue level which result in huge number of poor performing interactions between user simulator and dialogue agent during initial learning phase. To establish a robust dialogue policy in a computationally efficient manner, the proposed approach leverages imitation learning using behavior cloning in the initial stage. In the consequent step, DRL-based dialogue agent utilizes these pretrained weights to establish scalable dialogue policy in multi-domain environment. The proposed approach demonstrated significant improvement in the performance of dialogue policy and overall performance of dialogue system when compared with other systems. It may be helpful for designing scalable dialogue systems to support new tasks without using user simulators by cutting down the training time significantly. However, the following limitations of the behavior cloning approach need to be considered; mistakes in expert trajectories will be replicated by the dialogue agent as well therefore error free expert dataset is required. Also, expert biases may lead to suboptimal dialogue policies or sometimes better generalization may not be achieved because the user behavior is not taught in the in the given expert trajectories, causing overfitting. Recent advancements using large language models (LLMs) utilize task-specific prompts to adapt new tasks or domains. By default, LLMs are not multi-turn, moreover, they are trained on a large general-purpose corpus, exploring the efficacy of LLMs for modeling prompt-based adaptation of new tasks in TOD systems is left for future work.

REFERENCES





- [1] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: a survey and new perspectives," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–38, Jan. 2020, doi: 10.1145/3285029.
- [2] K. Qian and Z. Yu, "Domain adaptive dialog generation via meta learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2639–2649, doi: 10.18653/v1/P19-1253.
- [3] A. Rastogi, D. Hakkani-Tur, and L. Heck, "Scalable multi-domain dialogue state tracking," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017 - Proceedings*, 2017, vol. 2018-Janua, pp. 561–568, doi: 10.1109/ASRU.2017.8268986.
- [4] Y. Lu *et al.*, "Goal-oriented end-to-end conversational models with profile features in a real-world setting," in *Proceedings of the 2019 Conference of the North*, 2019, pp. 48–55, doi: 10.18653/v1/N19-2007.
- [5] P.-H. Su *et al.*, "On-line active reward learning for policy optimisation in spoken dialogue systems," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 2431–2441, doi: 10.18653/v1/P16-1230.
- [6] C. Shyalika, T. Silva, and A. Karunananda, "Reinforcement learning in dynamic task scheduling: a review," *SN Computer Science*, vol. 1, no. 6, p. 306, Nov. 2020, doi: 10.1007/s42979-020-00326-5.

- [7] P.-H. Su, M. Gašić, and S. Young, "Reward estimation for dialogue policy optimisation," *Computer Speech & Language*, vol. 51, pp. 24–43, Sep. 2018, doi: 10.1016/j.csl.2018.02.003.
- [8] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," *arXiv preprint*, pp. 1–9, 2013.
- [9] Y. J. Zhao, Y. L. Li, and M. Lin, "A review of the research on dialogue management of task-oriented systems," *Journal of Physics: Conference Series*, vol. 1267, no. 1, p. 012025, Jul. 2019, doi: 10.1088/1742-6596/1267/1/012025.
- [10] W.-C. Kwan, H.-R. Wang, H.-M. Wang, and K.-F. Wong, "A survey on recent advances and challenges in reinforcement learning methods for task-oriented dialogue policy learning," *Machine Intelligence Research*, vol. 20, no. 3, pp. 318–334, Jun. 2023, doi: 10.1007/s11633-022-1347-y.
- [11] J. Zhu, F. Wu, and J. Zhao, "An overview of the action space for deep reinforcement learning," in *2021 4th International Conference on Algorithms, Computing and Artificial Intelligence*, Dec. 2021, pp. 1–10, doi: 10.1145/3508546.3508598.
- [12] Z. Lipton, X. Li, J. Gao, L. Li, F. Ahmed, and L. Deng, "BBQ-networks: efficient exploration in deep reinforcement learning for task-oriented dialogue systems," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 2018, pp. 5237–5244, doi: 10.1609/aaai.v32i1.11946.
- [13] Z. Zhang, X. Li, J. Gao, and E. Chen, "Budgeted policy learning for task-oriented dialogue systems," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3742–3751, doi: 10.18653/v1/P19-1364.
- [14] H. Cuayahuitl, S. Yu, A. Williamson, and J. Carse, "Scaling up deep reinforcement learning for multi-domain dialogue systems," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 3339–3346, doi: 10.1109/IJCNN.2017.7966275.
- [15] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, pp. 2094–2100, Mar. 2016, doi: 10.1609/aaai.v30i1.10295.
- [16] Z. Wang, T. Schaul, M. Hessel, H. V. Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," in *33rd International Conference on Machine Learning, ICML 2016*, 2016, vol. 4, pp. 2939–2947.
- [17] R. Rosalina, A. Sengkey, G. Sahuri, and R. Mandala, "Generating intelligent agent behaviors in multi-agent game AI using deep reinforcement learning algorithm," *International Journal of Advances in Applied Sciences*, vol. 12, no. 4, pp. 396–404, Dec. 2023, doi: 10.11591/ijaas.v12.i4.pp396-404.
- [18] B. Liu, G. Tur, D. Hakkani-Tur, P. Shah, and L. Heck, "End-to-end optimization of task-oriented dialogue model with deep reinforcement learning," *arXiv preprint*, pp. 1–6, 2017, [Online]. Available: <http://arxiv.org/abs/1711.10712>.
- [19] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," in *Advances in Neural Information Processing Systems*, 1999, vol. 12, [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.
- [20] T. Zhao, K. Xie, and M. Eskenazi, "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*, 2019, pp. 1208–1218, doi: 10.18653/v1/N19-1123.
- [21] B. Peng *et al.*, "Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2231–2240, doi: 10.18653/v1/D17-1237.
- [22] Y. Wu, X. Li, J. Liu, J. Gao, and Y. Yang, "Switch-based active deep Dyna-Q: efficient adaptive planning for task-completion dialogue policy learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Jul. 2019, vol. 33, no. 01, pp. 7289–7296, doi: 10.1609/aaai.v33i01.33017289.
- [23] N. Lubis *et al.*, "LAVA: latent action spaces via variational auto-encoding for dialogue policy optimization," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 465–479, doi: 10.18653/v1/2020.coling-main.41.
- [24] G. Weisz, P. Budzianowski, P.-H. Su, and M. Gasic, "Sample efficient deep reinforcement learning for dialogue systems with large action spaces," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 11, pp. 2083–2097, Nov. 2018, doi: 10.1109/TASLP.2018.2851664.
- [25] A. Ohashi and R. Higashinaka, "Post-processing networks: method for optimizing pipeline task-oriented dialogue systems using reinforcement learning," in *SIGDIAL 2022 - 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue, Proceedings of the Conference*, 2022, pp. 1–13, doi: 10.18653/v1/2022.sigdial-1.1.
- [26] X. Zhao, L. Xia, L. Zou, H. Liu, D. Yin, and J. Tang, "UserSim: user simulation via supervised generative adversarial network," in *Proceedings of the Web Conference 2021*, Apr. 2021, pp. 3582–3589, doi: 10.1145/3442381.3450125.
- [27] W. Shi, K. Qian, X. Wang, and Z. Yu, "How to build user simulators to train RL-based dialog systems," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1990–2000, doi: 10.18653/v1/D19-1206.
- [28] X. Luo, Z. Liu, S. Xiao, X. Xie, and D. Li, "MINDSim: user simulator for news recommenders," in *Proceedings of the ACM Web Conference 2022*, Apr. 2022, pp. 2067–2077, doi: 10.1145/3485447.3512080.
- [29] Z. Li, J. Kiseleva, and M. De Rijke, "Dialogue generation: from imitation learning to inverse reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Jul. 2019, vol. 33, no. 01, pp. 6722–6729, doi: 10.1609/aaai.v33i01.33016722.
- [30] R. Takanobu, H. Zhu, and M. Huang, "Guided dialog policy learning: reward estimation for multi-domain task-oriented dialog," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 100–110, doi: 10.18653/v1/D19-1010.
- [31] D. Silver *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017, doi: 10.1038/nature24270.
- [32] D. Silver *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint*, pp. 1–19, 2017, [Online]. Available: <http://arxiv.org/abs/1712.01815>.
- [33] S. Young, M. Gasic, B. Thomson, and J. D. Williams, "POMDP-based statistical spoken dialog systems: a review," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1160–1179, May 2013, doi: 10.1109/JPROC.2012.2225812.
- [34] N. Roy, J. Pineau, and S. Thrun, "Spoken dialogue management using probabilistic reasoning," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics - ACL '00*, 2000, pp. 93–100, doi: 10.3115/1075218.1075231.
- [35] L. Zhou, K. Small, O. Rokhlenko, and C. Elkan, "End-to-end offline goal-oriented dialog policy learning via policy gradient," *arXiv preprint*, 2017, [Online]. Available: <http://arxiv.org/abs/1712.02838>.
- [36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint*, pp. 1–12, 2017, [Online]. Available: <http://arxiv.org/abs/1707.06347>.
- [37] M. Eric *et al.*, "MultiWOZ 2.1: a consolidated multi-domain dialogue dataset with state corrections and state tracking baselines," in *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*, 2020, pp. 422–428.
- [38] M. Thakkar and N. Pise, "Survey of available datasets for designing task oriented dialogue agents," in *2019 International Conference on Mechatronics, Remote Sensing, Information Systems and Industrial Information Technologies (ICMRSISIT)*, Dec. 2019, pp. 1–10, doi: 10.1109/ICMRSISIT46373.2020.9405898.





- [39] M. Thakkar and N. N. Pise, "Leveraging transformer-based pretrained language model for task-oriented dialogue system," *International Journal of Computers*, vol. 8, 2023.
- [40] Q. Zhu *et al.*, "ConvLab-3: a flexible dialogue system toolkit based on a unified data format," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2023, pp. 106–123, doi: 10.18653/v1/2023.emnlp-demo.9.
- [41] M. Rohmatillah and J.-T. Chien, "Revise the NLU: a prompting strategy for robust dialogue system," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2024, pp. 10956–10960, doi: 10.1109/ICASSP48485.2024.10446138.
- [42] M. Rohmatillah and J.-T. Chien, "Hierarchical reinforcement learning with guidance for multi-domain dialogue policy," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 748–761, 2023, doi: 10.1109/TASLP.2023.3235202.

BIOGRAPHIES OF AUTHORS



Manisha Thakkar     is pursuing Ph.D. in Computer Engineering from Department of Computer Engineering and Technology at MIT World Peace University, Pune, India. With a robust background in artificial intelligence, she has developed significant expertise in dialogue systems, question-answering systems, and recommendation systems. She has completed a post graduate program in data science conducted in association with Purdue University. She can be contacted at email: manishathakkar@gmail.com.



Dr. Nitin Pise     received his Ph.D. in Computer Engineering. He currently works as Professor at the School of Computer Engineering and Technology, MIT World Peace University, Pune, India. He has 26 years of teaching experience and 2 years of industrial experience. His areas of interest are artificial intelligence, data analytics, high-performance computing, and algorithms. He has published more than 60 papers in national, international conferences, journals, and completed a post graduate program in data science by KPMG. He can be contacted at email: nitin.pise@mitwpu.edu.in.