SRTTU

Journal of Electrical and Computer Engineering Innovations (JECEI)

Journal homepage: http://www.jecei.sru.ac.ir

JECEI

Research paper

Actor Double Critic Architecture for Dialogue System

Y. Saffari, J. S. Sartakhti *

Department of Electrical and computer engineering, University of Kashan, Kashan, Iran.

Article Info

Article History:
Received 26 November 2022
Reviewed 28 December 2022
Revised 08 January 2023
Accepted 01 March 2023

Keywords:
Dialogue system
Actor-critic
Double DQN
Task-based

*Corresponding Author's Email Address: *salimi@kashanu.ac.ir*

Abstract

Background and Objectives: Most of the recent dialogue policy learning methods are based on reinforcement learning (RL). However, the basic RL algorithms like deep Q-network, have drawbacks in environments with large state and action spaces such as dialogue systems. Most of the policy-based methods are slow, cause of the estimating of the action value using the computation of the sum of the discounted rewards for each action. In value-based RL methods, function approximation errors lead to overestimation in value estimation and finally suboptimal policies. There are works that try to resolve the mentioned problems using combining RL methods, but most of them were applied in the game environments, or they just focused on combining DQN variants. This paper for the first time presents a new method that combines actor-critic and double DQN named Double Actor-Critic (DAC), in the dialogue system, which significantly improves the stability, speed, and performance of dialogue policy learning.
Methods: In the actor critic to overcome the slow learning of normal DQN, the critic unit approximates the value function and evaluates the quality of the policy used by the actor, which means that the actor can learn the policy faster. Moreover, to overcome the overestimation issue of DQN, double DQN is employed. Finally, to have a smoother update, a heuristic loss is introduced that chooses the minimum loss of actor-critic and double DQN.
Results: Experiments in a movie ticket booking task show that the proposed method has more stable learning without drop after overestimation and can reach the threshold of learning in fewer episodes of learning.
Conclusion: Unlike previous works that mostly focused on just proposing a combination of DQN variants, this study combines DQN variants with actor-critic to benefit from both policy-based and value-based RL methods and overcome two main issues of both of them, slow learning and overestimation. Experimental results show that the proposed method can make a more accurate conversation with a user as a dialogue policy learner.

Introduction

Task-based dialogue systems aim to interact with users to achieve their goals, such as movie ticket booking [1], restaurant reservation [2], and booking flights [3]. Dialogue systems, based on their structure of training, are divided into two categories: end-to-end and pipeline. The end-to-end ones directly map the user's action in a natural language way to the agent's response using a sequence-to-sequence model with supervised learning [4] Pipeline methods separate the system into some interdependent units, mainly natural language understanding (NLU), natural language generation (NLG), and dialogue management unit [5]. The dialogue management unit has the dialogue state tracker (DST) and the agent's policy, mostly represented by a neural network.

The goal of a dialogue manager is to learn the dialogue policy. The policy decides which action should to choose based on the input state. In other words, the agent is trained in the dialogue management unit to learn the dialogue policy, so that it can interact directly with the knowledge base to create the appropriate action for each input from the user or user simulator [4].

Supervised learning and RL are used to train a task-based dialogue system [6]. Based on the task, the input space can be very large and make the DST misunderstand the input from the user, cause of errors in the NLU unit. Therefore, to overcome this issue and insufficient available annotated data, Reinforcement learning (RL) can be used to learn the policy automatically using interacting with a user simulator **Error! Reference source not found.**Then learning the dialogue policy, which can be viewed as a Markov decision process (MDP) [8], [9], [10], is often formulated as an RL problem [11].

Two main categories of model-free RL algorithms are policy-based and value-based learning algorithms [12]. Deep Q-networks (DQN) is one of the most successful value-based RL algorithms [13]. As DQN has a maximization of overestimated action values, it tends to prefer overestimated to underestimated values. This overestimation makes DQN learn unrealistically high action values [14]. Double DQN, is one of the varies of DQN that solves the issue of overestimation in DQN by decoupling the action selecting function and Q value estimator [15].

Dialogue policy learning for tasks with large state-action space, such as booking a movie ticket, needs to take a lot of dialogue turns to be able to explore in such a large space, leading to a long trajectory and finally delayed and sparse reward signals. To deal with reward sparsity, it's possible to employ intrinsic reward after each action to guide the exploration. Another class of RL algorithms takes advantage of value-based and policy-based RL algorithms methods, called actor-critic architecture. The critic is used to approximate the value and the actor tried to learn the policy, while the critic helps to update the actor by determining how much the action selected by the agent is good [16].

Many of the sequential decision-making problems, such as game playing, use DQN and its extension of it to learn the policy [17], [18]. DQN has been used in the context of dialog policy learning too [4], [19] but there is less effort in studying the extensions of DQN, especially in combination with other categories of RL algorithms such as actor-critic which is a combination of value and policy-based algorithms.

This study presents a new model for dialogue policy learning in a task-based dialogue system that combines the double DQN and actor-critic approaches to take advantage of both.

According to our information, this study proposes the first method that combines double DQN with the actor-critic method in a dialogue system environment.

The main contributions of this paper are as follows:

- The whole structure of the proposed method is actor-critic. The critic used to employ intrinsic reward after each action to guide the exploration and makes the learning process more stable in high dimensional state-action space.
- To overcome the overestimation in DQN, double DQN is employed in the actor unit.
- To have a smoother and faster update in the agent, the minimum of two errors was calculated based on the critic, and double DQN was obtained for optimizing the actor.

The proposed model is evaluated on the movie ticket booking dataset. This dataset was collected via Amazon Mechanical Turk, with annotations provided by domain experts [4], [20]. The evaluation results show that the proposed model operates well in the dialogue system tasks with high dimensional state-action space and the training process is more efficient than the basic algorithms, double DQN, and actor-critic.

The rest of the paper is organized as follows. First, briefly, researches that have investigated DQN methods in the dialogue system or have presented a new architecture of the combination of methods are reviewed. Then is provided some necessary background knowledge of the double DQN and actor-critic methods. The next section involves the main contribution of this study, which discusses the presented method which combines double DQN with the actor-critic network, and presents the details of the method and components of the dialogue system. In the results and discussion section, the experimental results of the proposed method on the movie ticket booking dataset are presented. Finally, the conclusion section, concludes the paper and makes suggestions for future research.

Related Work

There are many DRL-based approaches that are configured for dialogue management, but most of the complex DRL-based models that combine the variants of DQN or other types of RL algorithms, have been investigated in a game environment.

Many researchers had studied deep reinforcement learning (DRL) algorithms (a combination of RL and deep learning) to improve their training performance. However, the use of RL faces several problems such as the requirement for a very large number of train data, the instability of learning, slow learning, and convergence problems caused by the overestimation.

The first DRL method is DQN which for the first time proposed to play Atari games by Google Deep Mind [17],

[13]. Using a convolutional neural network (CNN), DQN extracts features from a screen of the game and then uses Q-learning [21] to learn how to play the game. A lot of research has been conducted on versions of DQN, to improve it.

Normal policy-based methods are slow as they have to estimate the value of each action through multiple episodes, and then normalize the sum of the future discounted rewards for each action. Actor-critic is a combination of value and policy-based methods. The critic gives directions to the actor, which means that the actor can learn the policy faster. Actor-critic needs less computation too because the policy is explicitly stored, and actor-critic methods can learn the optimal probabilities of selecting various actions. There is a lot of research that proposed models based on the actor-critic in all environments such as asynchronous advantage actor-critic (A3C) [22], A2C [23], actor-critic using Kronecker-factored trust region (ACKTR) [24], and soft actor-critic.

The actor-critic method has achieved superior performance on sequential decision-making problems [16], [19], [25]. Recently, some actor-critic algorithms are applied for dialogue policy learning, such as A2C [22], eNAC [26], and ACER [27]. ACER is the most efficient off-policy actor-critic method that, unlike basic actor-critic methods, it uses experience replay and some other methods to reduce the bias and variance of estimators. Su et al. employed the actor-critic model in dialogue policy optimization and showed that it can make convergence faster and more stable than other RL methods such as DQN [14].

In some of the complex stochastic environments, DQN leads to a suboptimal policy because of the overestimation of action values. This overestimation happens because of the noise on the approximations due to the generalization [14]. Hasselt et al. [15] presented a double-Q estimator for value-based RL methods to decrease the overestimated Q values. This leads to a more stable learning process and improves performance. While the double-Q estimator is the most popular method to solve the overestimation problem, there are other methods to solve this problem such as dropout techniques on DQN [28], cross DQN algorithm [29], and decreasing learning rate [30].

In the following, some popular research that tried to improve DQN in dialogue systems has been described. Firstly, studies that investigated DQN variants with different setups in a dialogue system environment to find the most suitable model have been described and concluded that DDQN could not outperform other variants such as dueling DQN. Then the works that investigated the actor-critic method as a combination of policy and value-based methods to overcome the problems of DQN have been described and concluded that actor-critic can help to increase the speed of learning in the dialogue system. Also, some of the comparisons between DQN variants and actor critic showed that actor critic can outperform all of them including DDQN this can be because of the large state action space in the dialogue system and the huge help of critic to overcome the slow exploration and learning. Moreover, results of related works show that double DQN is ineffective in an actor-critic because due to the slow-changing policy in an actor-critic, the current and target value estimates remain too similar to avoid maximization bias. Reference [31] developed a novel variant of double Q-learning which limits possible overestimation. And their results demonstrate that mitigating overestimation can greatly improve the performance of modern algorithms.

References [32] and [33] investigated extensions of DQN such as double DQN, distributional DQN, and dueling DQN, individually and in combination, in the task-based dialogue system to find the most suitable model in a dialogue system. Finally, they concluded that choosing a specific algorithm for a specific task is not the right thing to do, and it depends on the state-action space, type of task, and the dataset. However, they have chosen the dueling network as the best choice over other methods or combinations. It seems that the network structure of basic DQN cannot model the Q-function perfectly while dueling DQN using an extra value function performs better in a dialogue system.

Fatemi used deep policy networks which are trained with an advantage actor-critic method for statistically optimized dialogue systems. The training process is done in a two-step approach: supervised learning and batch-based learning. The main benefit of their method, which paves the way for developing trainable end-to-end dialogue systems, was a combination of supervised and RL. They showed that the RL method based on an actor-critic architecture can exploit a small amount of data and can be used to improve the convergence speed of RL in dialogue systems. They compared the results of DQN, double DQN, advanced actor-critic (A2C), and SARSA algorithms on the dialogue state tracking challenge 2 (DSTC2) [34] dataset for the restaurant domain. According to experimental results, A2C had the fastest and best performance in convergence [35].

To reach to a faster and more stable learning, Gao proposed an adversarial A2C model which could perform well in the function of the dialogue system for ordering movie tickets. The proposed model trains a discriminator through an expert data file and online experiences. Then the trained discriminator is used as an additional critic to guide policy learning [23]. Also, Yen-Chen Wu proposed an actor-double-critic model to improve the performance stability of the DRL in a voice-based dialogue system for restaurant ordering [25].

As mentioned in this section, most of the improvements in DQN and the combination of it with actor-critic methods are employed in game environments. Based on our information, there is no work that tries a combination of actor critic and double DQN in dialogue system environment. As the dialogue system environment in this study has a discrete space and usually dialogue systems in the real world suffer from slow convergence, two most popular extensions, double DQN to overcome the overestimation and actor-critic to faster convergence have been chosen to analyze and combine them to employ in this study.

Background

In this section firstly, the structure of whole dialogue systems described and after formulization the dialogue policy learning process as a Markov decision process, the two RL method that have been used in proposed new model in this study, are described.

An end-to-end dialogue system has a user or a user simulator with a user goal. A user goal represents the user's goal of the conversation at a particular task. After choosing a goal, the user’s utterance passed through the NLU unit that the output of this unit is a lower-level representation of a natural language sentence, called a semantic frame. The DST takes the user’s action and the history of the current conversation to build a state representation as input for the agent to learn the policy. The policy of the agent unit chooses an action using interaction with a database to fill the information slots. The agent's output, which is the action in the form of a semantic frame, is sent to the NLG component, and it converts the action to a natural language format for the user [23].

In a discrete space of dialogue, at each time step t, the current state $s_t \in S$ of the environment is sent to the agent. The agent responds by selecting an appropriate action $a_t \in A$. The user gets this action and, based on the affection of it in the environment, gives to the agent a signal named as reward $r_{t+1} \in R$ and new state $s_{t+1} \in S$. Formalization of this cycle as a Markov decision process (MDP) is in the form of $< S, A, R_{t+1}, \mathrm{S}_{t+1}, \gamma >$ [36].

A. *Double DQN*

Deep Q-network is the beginning of the development of RL into more complex decision-making problems. It tries to teach a network to predict the $Q(s,a)$ value of action a by receiving a state s. Using of target network trick, the loss to update the parameters of the agent at DQN with Q' as target network, formulated as (1), where γ is discount factor which $\gamma \in [0,1]$, and α is learning rate [15]:

$$\mathrm{L} = \left(\mathrm{R}_{t+1} + \gamma \max_{a'} \mathrm{Q}'\left(\mathrm{s}_{t_{+1}}, \mathrm{a}'\right) - \mathrm{Q}(\mathrm{s}_t, \mathrm{a}_t)\right) \qquad (1)$$

To solve the problem of the overestimation bias due to the maximization step in the conventional DQN, can decouple action selection and evaluation in DQN and rewrites the loss as (2) to update double DQN (DDQN):

$$L = \left(R_{t+1} + \gamma Q'(s_{t+1}, a) - Q(s_t, a_t)\right) \qquad (2)$$

$$a = argma\,x_{(a')}\, Q\left(s_{(t+1)}, a'\right) \qquad (3)$$

B. *Actor-Critic*

Policy optimization methods are divided into two main categories: policy-based methods such as policy gradient and value-based such as Q-learning. But both of these major methods have drawbacks, which, combining two methods, can be complementary. Fig. 1 shows the general architecture of an actor-critic, combining policy-based and value-based approaches. The actor unit is used to generate the action.

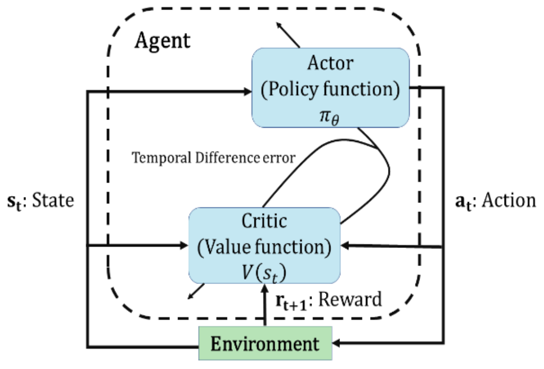

Fig. 1: Actor-Critic architecture.

The critic unit is supposed to approximate the value function and evaluate the quality of the policy used by the agent [37]. The evaluation method is the temporal difference (TD) error in (4), where V is the value function implemented by the critic [16]:

$$\delta_t = R_{(t+1)} + \gamma V\left(s_{(t+1)}\right) - V(s_t) \qquad (4)$$

The critic uses the TD error to update itself (value function’s parameters: w) in the gradient direction:

$$w \leftarrow w + \alpha\delta \nabla_w V(s) \qquad (5)$$

The actor too updates itself (the policy parameters: θ) using the evaluation from the critic:

$$\theta \leftarrow \theta + \alpha\delta\nabla_\theta \ln\pi(s, a) \qquad (6)$$

Methodology

In this study as illustrated in Fig. 2, a novel hybrid architecture, combining actor-critic and DDQN in a dialogue system, has been proposed. The overall neural network architecture for policy learner is an actor-critic that for overcoming the overestimation of Q values, DDQN is used to calculate the unbiased Q values.

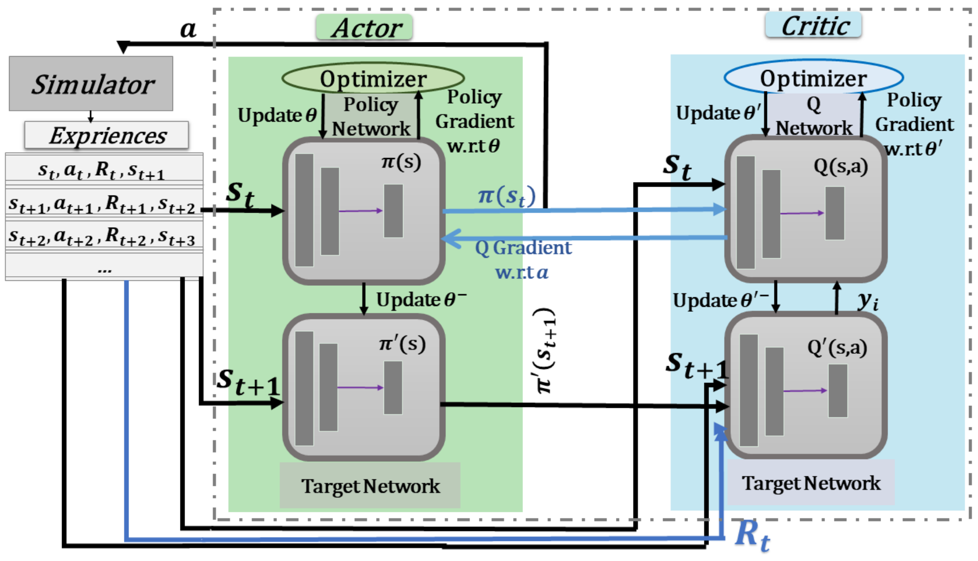

Fig. 2: Architecture of Double Actor-Critic: DAC.

The structure of the proposed policy learner model has been explained in detail in the proposed method subsection in further.

A. Dialogue System

In following firstly, the structure of the dialogue system framework and the elements of RL in this study like state and action, will be explained and then the proposed method will be explained in detail.

The dialogue framework used to apply the proposed agent is an end-to-end dialogue system that for simplicity, remains only the dialogue management component, and the NLU and NLG components have been removed. Fig. 3 shows the architecture of the dialogue system in this study.

To train end-to-end, a user simulator is needed that can automatically interact with the dialogue system. The simulator first generates a user goal while the agent does not know it and tries to accomplish that goal. Fig. 4 shows an example of the general format of a user goal. A user goal consists of two sections, inform slots and request slots. Inform slots are pairs of slot-value that express user restrictions and conditions. Requested slots are slots that the user wants to find some value for, during the conversation with the agent [4].

The inner function of the simulator is to produce action at each time step on a rule-based policy, based on the current state. The initial selected action by the user, must have at least one request slot and include the name of the movie as an information slot.

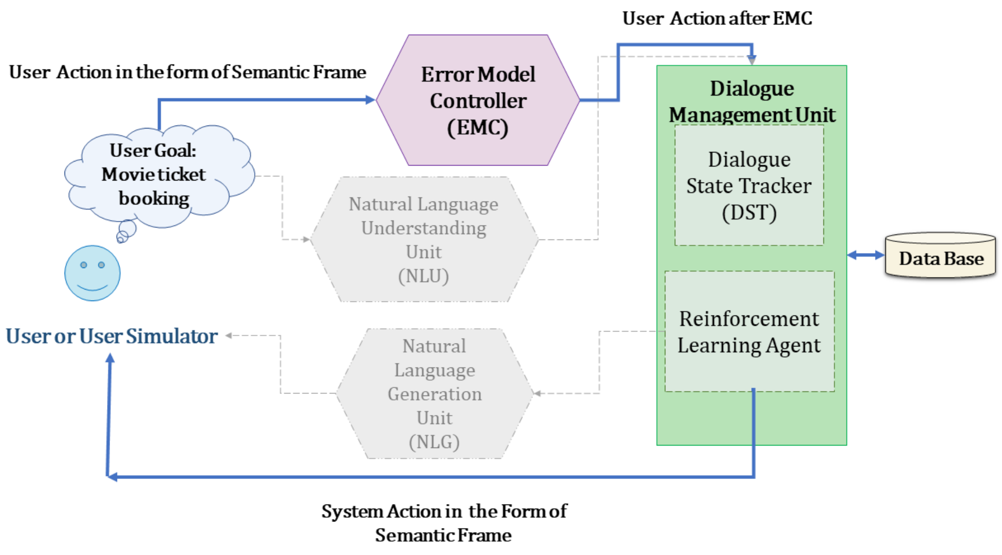
User Action after EMC
User Action in the form of Semantic Frame
Error Model Controller (EMC)
Dialogue Management Unit
User Goal: Movie ticket booking
Dialogue State Tracker (DST)
Natural Language Understanding Unit (NLU)
Data Base
User or User Simulator
Reinforcement Learning Agent
Natural Language Generation Unit (NLG)
System Action in the Form of Semantic Frame

Fig. 3: Architecture of the used End-to-End dialogue system.

$$["request-slots": ["date": "UNK", "starttime": "UNK"], "inform-slots": ["numberofpeople": "5", "moviename": "avengers"]$$

Fig. 4: Example of user goal.

Briefly, a conversation cycle between the user simulator and the agent, as shown in Fig. 3, can be expressed as follows: 1) The user simulator produces an initial action according to certain rules. 2) The action of the user simulator, after applying noise in EMC, enters the DST in dialogue management. 3) Based on the information slots received from the simulator, the history and database are updated through interaction with the database. 4) A state is generated based on information retrieved from the database and history. 5) The generated state enters to the agent in the dialogue management and, based on the policy, action is generated. 6) The generated action, if its intent is informative, returns to the DST to quantify the information slots by interacting with the database and the history information in the DST. 7) The updated action enters to the user simulator. If the goal of the simulator will be completed and the desired ticket is found, the dialogue will close, otherwise, it will either end unsuccessfully or the dialogue will continue. To optimize this policy, in this research, dialogue management has been formulated as an RL learning problem. The state of this study is made up of useful information about the current state and conversation such as the last user action, last agent action, and round-num. The round-num is encoded to let the agent know if the episode was close to its max limit number of rounds that the agent might take a match-found action. A match-found action happens when the match ticket has been found. The steps taken to reach a match found action made a negative signal for the agent. As NLG and NLU components are ignored in the proposed system, then all the actions in this study, including the actions on the user side and agent side, are in the form of a semantic frame. Action in the form of a semantic frame including an intent. The intent represents the type of action and the rest of the action is split into inform slots which contain constraints and request slots which the sender does not know. In Fig. 5 Fig. 6 the format of action and state that are used in this work have been shown.

Finally, since the NLU component has been removed in this study, an error model controller (EMC) is created to simulate the error resulting from NLU. EMC boosted the dialogue system and make it more resistant to the noise caused by the error in NLU. The error used in this study is changing slot values with a probability.

$$["intent": "request", "inform-slots": ["city" : "seattle"], "request-slots": ["date": "UNK", "starttime": "UNK"]]$$

Fig. 5: Example of Action in this study.

$$State = [user-action-rep,\ user-inform-slots-rep,\ user-request-slots-rep,\ agent-action-rep,\ agent-inform-slots-rep,\ agent-reques-slots-rep,\ current-slots-rep,\ turn-rep,\ turn-onehot-rep,\ kb-binary-rep,\ kb-count-rep]$$

Fig. 6: Format of a state in this study.

B. Proposed Method

As mentioned before, dialogue policy learning can be formalized as a Markov decision process (MDP) in the form of $< S, A, R_{t+1}, S_{t+1}, \gamma >$. The policy π is defined as a function $\pi: S \times A \to [0,1]$ that with probability $\pi(s,\ a)$ takes an action a in state s. The goal of RL is to find an optimal policy π^*, a policy that maximizes the value function which is the expected rewards of an episode in each state [36]. In this study, a hybrid deep reinforcement framework called Double Actor-Critic (DAC) was proposed, and ϵ-greedy search is employed during the training (please see Fig. 2). Using of a policy gradient method can suffer from a large variance. To overcome this issue, instead of increasing the size of batch size which causes a bad effect on sample efficiency, has been introduced a critic. The critic reduces the variance and, at the cost of bias, makes an improvement on the sample efficiency. In other words, reducing the cumulative reward using subtracting it with a baseline which here is the critic value, will make smaller gradients, and then smaller and more stable updates in policy learning will happen [14]. As the space of action-state of the studied task in this research is large, hence, it was decided to use the actor-critic architecture to learn policies better. To have more stability in training, batch learning is used too. The whole structure of DAC is actor-critic and the algorithm of DAC shows in Fig. 7. To approximate the true value function of the current policy π_θ, a critic $Q_w(s,\ a)$ was introduced. The objective function is (7):

$$J(\theta) = EQ_{\pi_\theta}(s, \pi_\theta(s)) \tag{6}$$

Using the deterministic policy gradient theorem, the gradient of the objective function is as (8):

$$\Delta_\theta J(\theta) = E\left[\Delta_\theta \pi_\theta(s) \Delta_a Q_{\pi_\theta(s,a)} \middle| a = \pi_\theta(s)\right] \tag{7}$$

Then the Bellman operator, with expectation, is taken with respect to the next state s', converted to (9) (line 5):

$$Q_\pi(s,a) = r(s,a) + \gamma E\left[Q\left(s', \pi(s')\right) \middle| s, a\right] \tag{8}$$

And the TD error as final loss at this actor-critic network becomes (10) (line 7):

$$E\left[Q_w(s,a) - r(s,a) + \gamma E\left[Q'_w(s', \pi'_\theta(s'))\middle| s, a\right]\right] \tag{9}$$

```
Algorithm 2 DAC Architecture
 1: Inputs:
        initial learning rates $\alpha_0$ and $\beta_0$ , target
        network replacement frequency $C$
 2: Initialize:
        network $Q$ with weights $(\theta, w)$ at random
        target $Q'$ with weights $(\theta', w') \leftarrow (\theta, w)$
 3: for t =1 to T do
 4:     Sample $M$ transitions $(s_i, a_i, r_i, s_{i:i+1})$ uniformly
 5:     $Y_{i_{(AC)}} = r_i + \gamma Q_{w'}(s_i, \arg\max_{b \in A} Q_{\theta'}(s_{i+1}, b))$
 6:     $Y_{i_{(DDQN)}} = r_i + \gamma Q_{\theta'}(s_i, \arg\max_{b \in A} Q_{\theta}(s_{i+1}, b))$
 7:     $\delta_w = \frac{1}{M} \sum_i \Delta_w (Y_{i_{(AC)}} - Q_w(s_i, a_i))$
 8:     $\delta_\theta = \frac{1}{M} \sum_i \Delta_\theta \pi_\theta(s_i) E[\Delta_a Q_w(s_i, a_i)]|_{a=\pi_\theta(s_i)}$
 9:     $\delta_{DDQN} = \frac{1}{M} \sum_i \Delta_\theta (Y_{i_{(DDQN)}} - Q_\theta(s_i, a_i))$
10: end for
11: update:
12: $\delta_\theta = \min(\delta_{DDQN}, \delta_\theta)$
13: $\theta \leftarrow \theta + \alpha_t \delta_\theta$
14: $w \leftarrow w + \beta_t \delta_w$
15: Every $C$ times, update target network: $(\theta', w') \leftarrow (\theta, w)$
16: Output: policy parameters $\theta$
```

Fig.7: The algorithm of DAC.

Hence, the objective function to optimize the policy in actor-critic is as (11) where (θ', w') referred to the parameters of actor and critic's target network (line 2) to stabilize learning in TD error and T is the number of steps in the episode (line 8).

$$\Delta_\theta J(\theta) = E[\textstyle\sum_{i=0}^{T-1} \Delta_\theta \log \pi_\theta (s_i) Q_w(s, a)] \quad (10)$$

One of the problems with DQN in this task is that due to the maximization operator in predicting $q(s, a)$, the Q value of some actions is likely to be overestimated. This problem led the agent to constantly learn some non-optimal actions. In this task, in Q −learning, the value function is updated with a greedy target at step $t + 1$:

$$y = r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \quad (11)$$

However, if the target is susceptible to error ϵ, then the maximum over the value along with its error will generally be greater than the true maximum [14]:

$$E_\epsilon\left[max_{a_{t+1}}(Q(s_{t+1}, a_{t+1}) + \epsilon)\right] \geq max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \quad (12)$$

To solve this problem, two separate networks are used, one to select the appropriate action and the other to calculate the value of actions, and since these two separate models are trained, it is less likely to estimate the same actions. To implement the proposed architecture, there are two similar networks called online and target as actor part, that per each some fixed number of episodes, the weights of the online network are copied to the target. In the target Q value calculation, for updating network weights, the Q value is obtained based on the selected action of the online model but through the target model. While in DQN a maximum is used to calculate the Q of the next state, but here first the best action of the next state is selected and then the Q value of that action is extracted from the target network. Finally, to have smaller gradients, the $Q(s, a)$ calculated using the online network in the actor part, at the episode $< e = s, a, r', s' >$, is subtracted by the (14) (line 6):

$$Y^{double} = r' + \gamma Q_{target}(s', A) \quad (13)$$

which A is the optimal action obtained by the online network and the result of this subtracting is assumed as a double loss (line 9). To have a smoother update in the agent, the minimum of two losses of actor-critic (10) and double was obtained for updating of actor, in each batch (line 12). The target network of DDQN updates periodically based on the weights of actor (line 15), and critic updates based on the target Q value that is calculated in the critic component. Updating of actor-critic done using stochastic gradient V with learning rates α and β respectively (lines 13,14), which adjusted are using ADAM [38[.

Results and Discussion

This study considers a task-based dialogue system for helping users to book movie tickets. During the conversation, the dialogue system gathers information about the customer's desires and ultimately books the movie tickets. The environment then assesses a binary outcome (success or failure) at the end of the conversation, based on whether or not the movie is found in a limited time. The focus of this study is on the evaluating the DQN methods and the proposed novel model, DAC.

C. Dataset

The basic conversational data were collected via Amazon Mechanical Turk, with annotations provided by domain experts. The annotated dataset that has been used in this study is for movie ticket booking [4], [20] It consists of 2890 dialogue sessions, with approximately 7.5 turns per session on average. The annotated data includes 29 dialogue actions and 11 slots, of which most of the slots are inform slots, that users can use to constrain the search, and some are request slots, of which users can ask for values from the agent.

D. Analysis

For analyzing the performance and improvements in the presented new architecture, the dialogue system of the proposed method was trained with DDQN and the proposed model DAC. Then based on the success rate factor, these models were compared and the results of these experiments are shown in follow. All implementations have been performed in Google's Colab environment. The hyperparameter settings are given in Table 1.

Table 1: The parameters of model

Parameter	Value
discount factor	0.9
max memory size	100000
DQN hidden size	80
batch size	16
learning rate	1e-3
warm up memory	7000
number of episodes	15000
train frequency	100

E. Results

In this section, the actor-critic and DDQN and the combination of them implemented and the performance of them at the move ticket booking task, have been analyzed. The metric that has been used to measure the quality of the agent was the success rate. Per each transition, a reward based on the reward function and a success score is returned. A good policy should have a high success rate. The success rate (15) is known as the fraction of dialogues that are done successfully where the user goal matches the information the system acquired during the interaction:

$$SuccessRate = \frac{Period\ Success\ Total}{fixed\ frequency\ of\ train} \qquad (14)$$

I) Double DQN

Previous works [17] have experimented with variants of DQN on Atari games. Wang [39] did research that experimented with variants of DQN in the same task and dataset in the dialogue system and reported that DDQN improves very little in most environments while performing better in restaurant and taxi domains. But it seems not to affect significantly in the movie domain. Based on the setup of learning at experiments, as shown in Fig. 8 of DQN and DDQN, the behavior of both likely is the same. However, it looks like in training for future episodes, the results get better. Then, it can be concluded that DQN does not suffer a lot from the problem of overestimation in Q value predicting in this task, hence, the improvement of DDQN here is limited. This can happen, because of some reason, an agent at this task could predict Q values near to actual Q values and non-optimal actions are not given a higher Q value than the optimal best action.

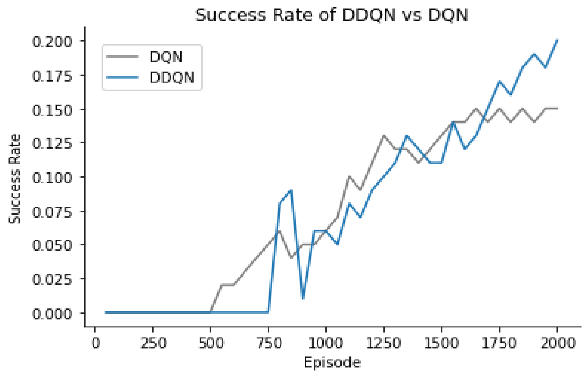

Fig. 8: Success rate of double DQN vs. DQN.

II) Actor Critic

Regular policy-gradient methods have slow convergence due to the large variances of the gradient estimates. The actor-critic methods attempt to reduce the variance using a critic network to estimate the value of the policy, which is then used to update the actor parameters [26]. As shown in Fig. 8 of DQN and actor-critic, the behavior of both likely is the same, but the actor-critic improved in stability success rate. It shows that the critic made actor decisions better and more accurately with more stability.

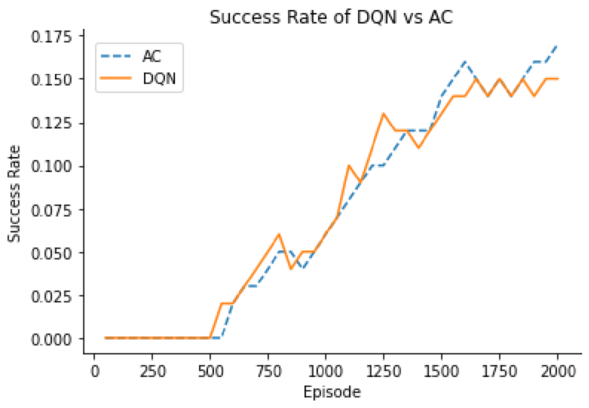

Fig. 9: Success rate of actor critic vs. DQN.

III) Double Actor Critic

In this paper, DDQN and actor-critic methods are combined to have the advantage of both. Based on the result of experiments on DDQN shows it doesn't have a significant effect to improve the DQN, but actor-critic made the agent more stable and faster. Hence, it is expected that the behavior of the proposed model, DAC, be affected by the actor-critic more and the behavior should be more stable. It is noteworthy that the proposed model reached the threshold of success rate faster too (please see Fig. 10). It can be concluded that the collaboration of the critic and target model of DDQN can result in better and solve some overestimation issues of DQN and actor-critic too. In Fig. 11, the behavior of all methods is shown.

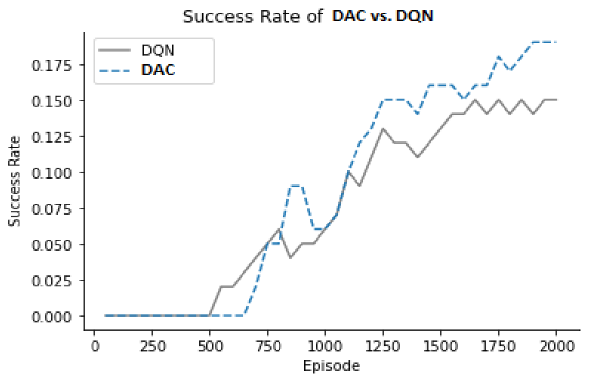

Fig. 10: Success rate of double actor critic vs. DQN.

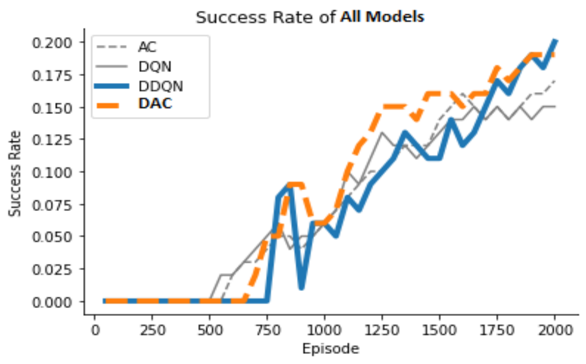

Fig. 11: Comparison of Actor-Critic, Deep Q Network (DQN), Double Deep Q Network (DDQN), and Double Actor-Critic (DAC) based on the success rate metric.

Conclusions

This study proposed a new architecture that combined two reinforcement methods, actor-critic and DDQN to have the advantage of both methods and examined it on a task-based dialogue system for booking the ticket movie task. DDQN has no significant effect on the results of DQN at this task and can conclude that this task does not suffer from notable overestimation that it can happen because of the task features. Actor-critic made more effect on the success rate and made the learning more stable and accurate. Finally, the investigated model, DAC, shows somehow comparable results. Actor-critic made it faster and more stable and can conclude that collaborating with the DDQN and actor-critic to make more accurate decisions, ends well as DDQN could solve new overestimation made by the actor-critic.

In a real application, the domain of dialogues is usually multi-domain and the diversity of actions is more and the training process is more complex. Then in this situation, learning is getting slower and more unstable. The biggest advantage of the proposed method in this study is a faster and more stable convergence in a complex domain.

In this study a user simulator with a rule-based policy has been used to interact with the agent in training. However, the best way is using a real user but it's not possible cause of the time and cost of it. This is obvious that the trained model with a user simulator is not excellent in the test step with the real user but the training with the simulator is a common and useful method. However, using a multi-agent interaction can be a good resolve that is out of the subject of this study.

The proposed idea of combining two methods and analysis from this research guide the future directions of applying more types of actor-critic methods to DQN variants for dialogue policy learning, although some methods may don't behave as well as they behave in Atari environments cause of different environmental features.

Author Contributions

Y.Saffari and JS.Sartakhti designed the experiments. Y.Saffari collected the data and carried out the data analysis. Y.Saffari and JS.Sartakhti interpreted the results and wrote the manuscript.

Acknowledgment

The author would like to thank the editor and reviewers for their helpful comments.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Abbreviations

rep	Representation
NLU	Natural Language Understanding
NLG	Natural Language Generation
DST	Dialogue State Tracker

References

[1] Z. C. Lipton, J. Gao, L. Li, X. Li, F. Ahmed, L. Deng, "Efficient exploration for dialog policy learning with deep {BBQ} networks & replay buffer spiking," arxiv preprint arxive: 1608.05081, 2016.

[2] T. H. Wen, D. Vandyke, N. Mrkšić, M. Gašić, L. M. Rojas-Barahona, P. H. Su, S. Ultes, S. Young, "A network-based end-to-end trainable task-oriented dialogue system," in Proc. 15th Conference of the European Chapter of the Association for Computational Linguistics: 438-449, Valencia, Spain, 2017.

[3] H. Cuay ahuitl, S. Renals, O. Lemon, H. Shimodaira, "Hierarchical Dialogue optimization using semi-markov decision processes," in Proc. 8th Annual Conference of the International Speech Communication Association: Interspeech: 2693-2696, 2007.

[4] X. Li, Y. N. Chen, L. Li, J. Gao, A. Celikyilmaz, "End-to-End task-completion neural dialogue systems," in Proc. Eighth International Joint Conference on Natural Language Processing, (1): 733–743, Taipei, Taiwan, 2017.

[5] H. Sun, C. Zhao, S. Liu, H. Jiang, "A pipeline dialogue system scheme," in Proc. 2nd International Conference on Machine Learning and Computer Application: 1-5, Shenyang, China, 2021.

[6] R. Fellows, H. Ihshaish, S. Battle, C. Haines, P. Mayhew, J. I. Deza, "Task-oriented dialogue systems: performance vs. quality-optima, a review," arxive preprint. arxive: 2112.11176, 2021.

[7] M. I. Bahria, Z. Yan, "Supervised machine learning approaches: A survey," Int. J. Soft Comput., (5): 946-952, 2015.

[8] R. Howard, Dynamic Programming and Markov Processes, The MIT Press, Cambridge, 1960.

[9] S. Young, M. Gasi˘ c, B. Thomson, J. D. Williams, "Pomdp-based statistical spoken dialog systems: A review," Proc. IEEE, 101(5): 1160–1179, 2013.

[10] J. D. Williams, S. Young, "Partially observable markov decision processes for spoken dialog systems," Comput. Speech Lang., 21(2): 393–422, 2007.

[11] J. Williams, A. Raux, D. Ramachandran, A. Black, "The dialog state tracking challenge," in Proc. SIGDIAL: 404–413, 2013.

[12] P. Swazinna, S. Udluft, D. Hein, T. Runkler, "Comparing model-free and model-based algorithms for offline reinforcement learning," IFAC-PapersOnLine, 55(15):19-26, 2022.

[13] V. Mnih , K. Kavukcuoglu, D. Silver, A. A. Rusu , J. Veness , M. G. Bellemare, A. Graves, M. Riedmiller, "Human-level control through deep reinforcement learning," Nature: 529-33, 26 Feb 2015.

[14] S. Thrun, A. Schwartz, "Issues in using function approximation for reinforcement learning," in Proc. 4th Connectionist Models Summer School, 1993.

[15] H. van Hasselt, A. Guez, D. Silver, "Deep reinforcement learning with double q-learning," arxive preprint arxiv:1509.06461, 2015.

[16] R. Chen, J. H. Goldberg, "Actor-critic reinforcement learning in the songbird," Curr. Opin. in Neurobiol., (65): 1-9, 2020.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. A. Riedmiller, "Playing atari with deep reinforcement learning," DeepMind Technologies, 2013.

[18] D. Silver, A. Huang, C. J Maddison, A. Guez, L. Sifre, G. V. D. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, . M. Lanc, "Mastering the game of go with deep neural networks," Nature (529): 484, 2016.

[19] B. Peng, X. Li, J. Gao, J. Liu, Y.-N. Chen, K.-F. Wong, "Adversarial advantage actor-critic model for task-completion dialogue policy learning," Int. Conf. IEEE, Acoustics, Speech and Signal Processing (ICASSP): 6149-6153, 2018.

[20] X. Li, Z. C. Lipton, B. Dhingra, L. Li, J. Gao, Y. N. Chen, "A user simulator for task-completion dialogues," arxiv preprint arxive:1612.05688, 2016.

[21] C. J. Watkins and P. Dayan, "Q-learning," Mach. Learn., (8): 279-292, 1992.

[22] V. Mnih, A. Puigdomènech Badia, "Asynchronous methods for deep reinforcement learning," arxiv preprint arxiv:1602.01783v2, 2016.

[23] J. Gao, M. Galley, L. Li, "Neural approaches to conversational ai, question answering, task-oriented dialogues and social chatbots," arxive preprint arxive:1809.08267, 2019.

[24] Y. Wu, E. Mansimov, S. Liao, R. Grosse, "Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation," arxiv preprint arxiv:1708.05144, 2017.

[25] Y. C. Wu, B. H. Tseng, M. Gas, "Actor-double-critic: incorporating model-based critic for task-oriented dialogue systems," Findings of the association for computational linguistics: EMNLP: 854–863, 2020.

[26] J. Peters, S. Vijayakumar, S. Schaal, "Natural Actor-Critic," ECML: 280–291, 2005.

[27] Z. Wang, V. Bapst, N. Hees, V. Mnih, R. Munos, K. Kavukcuoglu, N. D. Freitas, "Sample efficient actor-critic with experience replay," arxiv preprint arxive:1611.01224, 2016.

[28] M. Sabry, K. M. A. Amr , "On the reduction of variance and overestimation of deep q-learning," arxive preprint arxiv:1910.05983v1, 2019.

[29] X. Wang, A. Vinel, "Cross learning in deep q-networks," arxive preprint ariv:2009.13780v1, 2020.

[30] Y. Chen, L. Schomaker, M. A. Wiering, "An Investigation Into the Effect of the Learning Rate on Overestimation Bias of Connectionist Q-learning," in Proc. International Conference on Agents and Artificial Intelligence 2021.

[31] S. Fujimoto , H. van Hoof, D. Meger , "Addressing function approximation error in actor-critic methods," 2018.

[32] Y. A. Wang, Y. N. Chen, "Dialogue environments are different from games: Investigating variants of deep q-networks for dialogue policy," in Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU): 1070-1076, 2019.

[33] D. Vath, N. T. Vu, "To combine or not to combine? A rainbow deep reinforcement learning agent for dialog policy," University of Stuttgart, Institute for Natural Language Processing (IMS), 2019.

[34] M. Henderson, B. Thomson, J. D. William, "The second dialog state tracking challenge," in Proc. 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL), 2014.

[35] M. Fatemi, L. E. Asri, H. Schulz, J. He, K. Suleman, "Policy networks with two-stage training for dialogue systems," arxive preprint arxive: 1606.03152, 2016.

[36] H. R. Chinaei, B. Chaib-draa, L. Lamontagne, "Learning observation models for dialogue POMDPs," in Proc. Canadian Conference on Artificial Intelligence: Springer(7310), 2012.

[37] I. Grondman, L. Busoniu, G. A. D. Lopes, R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," IEEE Trans. Syst. Man Cybern. Part C Appl. Rev., 42(6): 1291–1307, 2012.

[38] D. P. Kingma, J. Ba, "Adam: A method for stochastic optimization," in 3rd Int.Conf. Learning Representations, San Diego, 2015.

[39] Z. Wang, T. Schaul, M. Hessel, H. V. Hasselt, M. Lanctot, F. De, "Dueling network architectures for deep reinforcement learning," Computer Science , Machine Learning, 2015.

Biographies

Yasaman saffari received her B.S. degree in Computer Engineering (Software) from Dr. Shariati Vocational and Technical Girls College in 2017 and her M.Sc. degree in Master of Arts in Computer Arts (Intelligent Simulators Design) from Faculty of Multimedia, Tabriz Islamic Art University in 2020. Her research interests include NLP, dialogue system, deep reinforcement learning.

- Email: y.saffari@grad.kashanu.ac.ir
- ORCID: 0000-0002-7178-0855
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: NA

Javad Salimi Sartakhti received his B.S. degree in Computer Engineering (Software) from University of Kashan in 2009, his M.Sc. degree in Computer Engineering (Software) from Tarbiat Modares University in 2012 and his Ph.D. degree in Computer Engineering (Software) from Isfahan University of Technology in 2016. His research interests include game theory & mechanism design, machine learning algorithms, deep learning and blockchain.

- Email: salimi@kashanu.ac.ir
- ORCID: 0000-0003-1183-1232
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: https://faculty.kashanu.ac.ir/salimi/en

How to cite this paper:
Y. Saffari, J. S. Sartakhti, “Actor double critic architecture for dialogue system,” J. Electr. Comput. Eng. Innovations, 11(2): 363-372, 2023.

DOI: 10.22061/jecei.2023.9346.614

URL: https://jecei.sru.ac.ir/article_1842.html