International Journal for Housing Science and Its Applications
Publish August 5, 2025. Volume 46, Issue 3 Pages 3579-3592

https://doi.org/10.70517/ijhsa463302

Exploring the Enhancement Path of Generative Model-Based Optimization of Natural Language Generation in Multi-Round Dialogues

Na Zhang[1,*]
[1] College of Computer and Artificial Intelligence, Henan Finance University, Zhengzhou, Henan, 450046, China
Corresponding authors: (e-mail: znwang1010@126.com).

Abstract Dialogue generation is a key research direction in natural language processing, and the adversarial generative network GAN has been widely used in the field of dialogue generation. In this paper, based on the reinforcement learning method, combining the generative adversarial network with the proximal policy optimization algorithm, a PPO-GAN dialogue generation model is proposed, and experimental validation of the model is carried out. The experimental results show that, comparing with the Adver-REGS dialog generation model that uses policy gradient to train GAN, the PPO-GAN model achieves the optimal values of similarity metrics BLEU-1, BLEU-2, BLEU-3, and BLEU-4, which are 19.7, 14.6, 10.8, and 9.5, respectively, and outperforms Adver-Regs in terms of correctness, smoothness, and relevance in generating replies. It also outperforms the Adver-REGS model in terms of correctness, fluency, and relevance of generated responses. In addition, comparing the Seq2Seq-Attention, REGS, RCDG, and PAML models, the PPO-GAN model also achieves higher quality of dialog generation and outperforms in terms of consistency of generated dialog. This study opens up a feasible path for optimization of multi-round dialogue generation and provides strong support for human-machine dialogue learning.

Index Terms reinforcement learning, proximal policy optimization, generative adversarial networks, dialogue generation

I. Introduction

Currently, the development of dialog systems has gone through an evolutionary process from rule-based dialog systems, statistical machine translation-based dialog systems, to today's deep learning-based dialog systems [1]. In this process, dialog systems have continuously improved their generative capabilities, from simple single-round dialogs to open-domain multi-round dialog generation [2], [3]. Multi-round dialog generation refers to the ability of a dialog system to naturally and coherently conduct multiple rounds of dialog within the scope of a single topic and interact with the user in depth [4]. Compared to single-round dialogue generation, multi-round dialogue generation requires a more in-depth understanding and analysis of the context, as well as the need to maintain the coherence and consistency of the dialogue [5]. In real life, people's daily conversations are often in the form of multi-round conversations, so multi-round dialog generation is of great significance in the practical application of AI technology.

It is difficult to define the concept of Natural Language Understanding (NLU) accurately, in terms of human comprehension thinking, Natural Language Understanding should be the whole process of processing, analyzing and thinking about language, that is to say, mapping language to a reasonable logical representation, which is commonly known as Natural Language Understanding [6], [7]. User conversations are usually with certain intentions, and intention understanding is the process of analyzing the intention of a user's sentence, and deep learning is widely used as intention analysis in the field of natural language generation (NLP) [8]. Zheng, Y et al [9] pointed out that NLU as an important part of the dialog system, its ability to detect out-of-domain (OOD) inputs is crucial in practical applications, and the role of NLU focuses on linguistic analysis, which is mainly aimed at understanding the semantics of the text, and the intention. Abro, W. A et al. constructed a natural language understanding framework for use in the domains of information search and opinion construction, which consists of two sub-models, an intent classifier and parameter similarity, and fine-tuned the model with an attentional mechanism for recognizing user conversational intent [10]. Tian, J et al. proposed a new intent model, and Knowledge Graph of Requirements (KGR) to extend the scope of requirement knowledge for conversational AI bots, which can effectively reduce redundancy in conversations and improve the performance of user intent recognition for multi-round dialog strategies [11].

After identifying the user's intention, the next action is dialog state tracking, which is one of the core of a dialog system, dialog state tracking estimates the user's dialog goal in every conversation, usually also using slot-filling

patterns or semantic frames [12], [13]. Liao, L et al. with the help of MultiwoZ 2.1 dialog dataset proposed a method to track dialog states by stepwise inference of dialog transitions, which outperforms the state-of-the-art methods [14]. Li, Q et al. proposed a modifiable state prediction model for dialog state tracking that uses a two-stage prediction process that addresses the problem of error propagation in existing models [15]. Khan, M. A et al. combined BERT, stacked bi-directional LSTMs and multiple attention mechanisms to address the scalability challenge, and constructed a new end-to-end dialog state tracking framework by dealing with unseen pairs of slots and re-training the model in the context of changes in the domain ontology [16]. Heck, M et al. proposed a new framework for dialog state tracking (TripPy-R), which can be trained without fine-grained supervision and is robust to sample sparsity and new concepts for learning to track [17]. Since there are interactions and dependencies between questions and answers in a multi-round dialog, this requires some correlation between the responses generated by the model and the context [18]. The above research work focuses on the problem of modeling context, using hierarchical neural networks and attention mechanisms, etc. to encode multi-round context, and differentiating the importance of utterances in different rounds in order to enhance the model's understanding of the context and to help generate replies that are more relevant to the context of the multiple rounds.

In this paper, we combine generative adversarial network and proximal policy optimization algorithm to construct a multi-round conversation generation model PPO-GAN. The algorithm generates conversations through the GAN model, differentiates between generated and real conversations through the discriminative model, adopts proximal policy optimization to train the GAN, and applies Monte Carlo sampling (MC-Search) to generate the conversation through Monte Carlo Sampling (MC-Search) while ensuring monotonous and non-decreasing training of the generation model. Calculate the reward corresponding to each word in the generated responses. And the generative model is trained based on forced guidance, and the rewards obtained from the discriminative model can be reused by limiting the gradient of the generative model iterations. In order to verify the effectiveness of the model, it is compared with other models in experiments, and the consistency of the multiple rounds of dialog generated by the model is evaluated.

II. Dialogue generation model based on reinforcement learning and proximal policy optimization

In order to achieve the optimization of natural language generation in multi-round conversations, this paper proposes a PPO-GAN conversation generation model that combines a reinforcement learning approach with a proximal policy optimization algorithm.

II. A.Enhanced learning

Reinforcement learning, in which the agent learns behavioral actions from the environment and maximizes the numerical reward payoff by interacting with the state of the environment, is often used to solve sequential decision problems. This section focuses on the Markov decision process and the policy gradient algorithm in reinforcement learning.

II. A. 1) Markov decision-making process

Markov Decision Processes (MDPs) [19] are a general framework for solving sequential problems and can be used to model reinforcement learning problems. A Markov decision process follows Markovianity, which means that in a sequential task, the next state s_{t+1} at the current moment t is related only to the current state s_t and the current action a_t, and is independent of the history state. A finite MDP can be represented by a quaternion (S, A, R, f) as follows:

(1) S is the set of all states in the environment, and $s_t \in S$ denotes the state of the agent at moment t.

(2) A is the set of possible actions to be performed by the agent, and $a_t \in A$ denotes the action taken by the agent at moment t.

(3) $R: S \times A \to \square$ is the reward function, which is the immediate payoff value r_t obtained by the agent for performing the action a_t in the state s_t at the moment t, and can be expressed as $r_t \sim R(s_t, a_t)$.

(4) $f: S \times A \times S \to [0,1]$ is the state migration function, which is the probability that the agent performs the action a_t to transfer to the next state s_{t+1} at the moment t located in the state s_t, which can be expressed as $s_{t+1} \sim f(s_t, a_t)$.

The ultimate goal of agent learning in a reinforcement learning problem is to maximize the cumulative expected reward payoff to obtain the optimal policy. The cumulative reward from the initial moment t to the termination moment T is defined as:

$$G_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'} \tag{1}$$

where G_t is called the payoff or cumulative reward and $\gamma \in (0,1]$ is called the discount factor.

The state-value function, $V^{\pi}(s)$, represents the expected reward that the agent receives for following the strategy π from state s_t to the end of the episode:

$$V^{\pi}(s) = \mathrm{E}_{\pi}[G_t | \ s_t = s] \tag{2}$$

The state action value function, $Q^{\pi}(s,a)$, represents the expected reward that the agent receives for following the policy π from executing the action a_t from state s_t to the end of the episode:

$$Q^{\pi}(s,a) = \mathrm{E}_{\pi}[G_t | \ s_t = s, a_t = \pi(s_t)] \tag{3}$$

A policy π is a strategy for an agent to take action a_t in state s_t, which is a mapping from state space S to action space A. A policy π is said to be optimal, denoted as π^*, when the expected payoff obtained by the agent by following the policy $\pi(s)$ is greater than or equal to the expected payoff value of any other policy.

The optimal policy is solved using a generalized policy iteration consisting of policy evaluation and policy improvement. Policy evaluation refers to the process of computing the value function with the policy known, and policy improvement refers to the process of improving the policy by the value function. The solution of the value function Q in policy evaluation follows the Bellman equation in recursive form:

$$Q^{\pi}(s_t, a_t) = \mathrm{E}_{\pi, s_{t+1} \sim E}[R(s_t, a_t) + \gamma E_{a_{t+1} \sim \pi}[Q^{\pi}(s_{t+1}, a_{t+1})]] \tag{4}$$

The optimal state action value function under the optimal policy also follows the Bellman equation:

$$Q^*(s,a) = \max_{\pi} \mathrm{E}\left[R_t \middle| s_t = s, a_t = a, \pi \right] \tag{5}$$

II. A. 2) Strategy Gradient

The policy gradient approach is an important method for solving reinforcement learning problems. Unlike value function-based reinforcement learning methods that first directly compute the value function by iteration and then improve the policy based on the value function, policy-based reinforcement learning methods optimize the policy by directly parameterizing the policy to compute the direction in which the policy may be updated. Since the magnitude of each update is small, parameterized policy methods are more likely to converge.

The policy gradient method parameterizes the policy by means of a stochastic policy network $\pi_{\theta}(s_t, a_t)$ with parameter θ. During the training process, the agent updates the policy by optimizing the gradient, with the ultimate goal of making the objective function optimal. The objective function can be defined as the expected return about the strategy:

$$J(\theta) = \mathrm{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \right] = \sum_{\tau} P(\tau;\theta) R(\tau) \tag{6}$$

where $P(\tau;\theta)$ denotes the probability of occurrence of the sequence τ.

The objective function is iteratively updated by the gradient descent method, i.e., the derivatives of the objective function with respect to θ are found and its parameters are updated with the following update formula:

$$\theta_{t+1} = \theta_t + \alpha \nabla_{\theta} J(\theta) \tag{7}$$

where α is the step size of the network parameter update, which is continuously updated with iterations to eventually converge.

II. B.Optimization Algorithm for Proximal Policies

The PPO algorithm [20] is an improved algorithm based on the TRPO algorithm [21]. Each iteration of the TRPO algorithm tries to select an appropriate step size from the current strategy such that the cumulative return obtained from the new strategy is monotonically increasing. Its objective function is as follows:

$$
\begin{aligned}
J^{TRPO}(\tilde{\theta}) &= \mathrm{E}_{s_t\sim\rho_{\pi_\theta},a_t\sim\pi_{\tilde{\theta}}}[A_{\pi_\theta}(s_t,a_t)] \\
&= \mathrm{E}_{s_t\sim\rho_{\pi_\theta},a_t\sim\pi_{\tilde{\theta}}}[k_t(\theta,\tilde{\theta})A_{\pi_\theta}(s_t,a_t)] \\
&s.t.\bar{D}_{KL}^{\rho_{\pi_\theta}}(\theta,\tilde{\theta}) \le \delta
\end{aligned} \tag{8}
$$

where $A_{\pi_\theta}(s_t,a_t) = Q_{\pi_\theta}(s_t,a_t) - V_{\pi_\theta}(s_t)$ is the dominance function. $k_t(\theta,\tilde{\theta}) = \frac{\pi_{\tilde{\theta}}(a_t|\ s_t)}{\pi_\theta(a_t|\ s_t)}$ is the importance sampling weight, also known as the ratio of old to new strategies. $\pi_{\tilde{\theta}}(a_t|\ s_t)$ denotes the probability distribution of the target strategy, and $\pi_\theta(a_t|\ s_t)$ denotes the probability distribution of the current behavioral strategy. The $\bar{D}_{KL}^{\rho_{\pi_\theta}}(\theta,\tilde{\theta})$ denotes the average KL dispersion of the new strategy $\tilde{\theta}$ over the old strategy θ.

In order to control the update magnitude of the strategies, the PPO algorithm employs two different optimization schemes, both of which can better limit the discrepancy between the old and new strategies. The PPO algorithms described in this paper all use a truncated objective function approach.

II. B. 1) Truncated objective function method

In the PPO algorithm, $k_t(\theta,\tilde{\theta})$, i.e., the importance sampling weights, are restricted to an interval, and the step size of the update is limited by controlling the size of the interval. Compared to the TRPO algorithm, which uses KL scatter for limiting, the PPO algorithm uses $k_t(\theta,\tilde{\theta})$ for limiting, which is simpler and easier to implement. The objective function of the PPO algorithm is as follows:

$$
\begin{aligned}
&J^{CLIP}(\tilde{\theta}) = \mathrm{E}_{s_t\sim\rho_{\pi_\theta},a_t\sim\pi_{\tilde{\theta}}} \\
&\left[\min\left(k_t(\theta,\tilde{\theta})A_{\pi_\theta}(s_t,a_t), clip\left(k_t(\theta,\tilde{\theta}),1-\varepsilon,1+\varepsilon\right)A_{\pi_\theta}(s_t,a_t)\right)\right]
\end{aligned} \tag{9}
$$

where $C_t(\varepsilon) = clip\left(k_t(\theta,\tilde{\theta}),1-\varepsilon,1+\varepsilon\right)$ denotes the truncation term, which constrains the importance sampling weights $k_t(\theta,\tilde{\theta})$ constrained to be within $[1-\varepsilon,1+\varepsilon]$, with ò as the limiting parameter. The $\min$ function serves to minimize the original and truncated terms, so that the truncated term acts as a restriction when the policy update is offset outside the predetermined interval.

II. B. 2) Adaptive KL Penalty Coefficient

The KL scatter is constrained using an adaptive KL penalty coefficient κ. The method eliminates the constraints by constructing a Lagrangian function with the following objective function using the adaptive KL penalty coefficients:

$$
\begin{aligned}
&J^{KL}(\tilde{\theta}) = \mathrm{E}_{s_t\sim\rho_{\pi_\theta},a_t\sim\pi_{\tilde{\theta}}} \\
&\left[k_t(\theta,\tilde{\theta})A_{\pi_\theta}(s_t,a_t) - \kappa KL\left[\pi_\theta\left(\cdot|s_t\right),\pi_{\tilde{\theta}}\left(\cdot|s_t\right)\right]\right]
\end{aligned} \tag{10}
$$

Because it is difficult to determine the value of the penalty coefficient κ in different problems at different stages, it is not possible to optimize the policy network parameters directly using Eq. (10). The size of the penalty coefficient κ is adjusted according to the relationship between the size of the sample-estimated average KL scatter $\bar{D}_{KL}$ and the defined target value D_{target}. The formula for $\bar{D}_{KL}$ is as follows:

$$
\bar{D}_{KL}(\theta,\tilde{\theta}) = \mathrm{E}\left[KL\left[\pi_\theta(\cdot|\ s_t),\pi_{\tilde{\theta}}(\cdot|\ s_t)\right]\right] \tag{11}
$$

If $\bar{D}_{KL} < D_{target}$, it implies that the strategy update is restricted, so make $\kappa \leftarrow \kappa/2$, which is equivalent to relaxing the restriction on the KL scatter constraint. If $\bar{D}_{KL} > D_{target}$, it means that the strategy update is too large, so make $\kappa \leftarrow \kappa \times 2$, which is equivalent to strengthening the restriction on the KL scatter constraint. where the penalty coefficient κ is equivalent to the adaptive learning rate. The updated penalty coefficient κ will be used in the next round of policy update. Even if there is a large difference between the KL scatter and D_{target}, the penalty coefficient κ can be adaptively adjusted relatively quickly.

The dominance function estimation method and the optimization method of adding extra entropy reward are also employed in the PPO algorithm to further improve its performance. Constructing the dominance function using

Generalized Advantage Estimation (GAE) reduces the variance so that the algorithm does not produce large fluctuations.GAE is calculated as follows:

$$\begin{aligned}\hat{A}_t &= \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \\ \delta_t &= r_t + \gamma V(s_{t+1}) - V(s_t)\end{aligned} \tag{12}$$

When applying the PPO algorithm to a network structure with shared parameters of the Actor-Critic method, in addition to the truncated payoffs, the objective function adds an error term on the estimation of the state-value function and an entropy regularity term on the policy model to encourage exploration. Thus, the optimized objective function is as follows:

$$L^{CLIP}(\tilde{\theta}) = \mathrm{E}_{s_t\sim\rho_{\pi_\theta},a_t\sim\pi_{\tilde{\theta}}}\left[J^{CLIP}(\tilde{\theta}) - c_1(V_\theta(s_t) - V_{target})^2 + c_2 H(s_t,\pi_\theta)\right] \tag{13}$$

where c_1 and c_2 are two constant hyperparameters. $(V_\theta(s_t) - V_{target})^2$ is the mean-square error of the state-value function, the smaller the error the better. $H(s_t,\pi_\theta)$ denotes the entropy value of the strategy π_θ, the larger the entropy the better.

II. C.PPO-GAN Multi-Round Dialogue Generation Model

II. C. 1) Pre-training to generate models

The structure of the generative model is shown in Fig. 1, which is an encoder-decoder structure with an attention mechanism. Both the encoding part and the decoding part of the generative model are composed of RNNs. First, the encoding part uses the RNN to encode the input word h_k into a vector representation m_k. Then, an attention mechanism is used to obtain the effect of each word in the input dialog on the words that will be generated during the decoding process. Finally, the output c_t is conditionally generated. In this case, the attention mechanism is computed by performing a dot product operation to obtain the similarity weight α_k^t from the hidden output m_k at each moment of the encoder and the hidden state z_{t-1} at the previous moment of the decoder, and utilizing the softmax function to transform α_k^t into probability to get $\hat{\alpha}_k^t$. In this case, $\hat{\alpha}_k^t$ represents the weights of the input word encoding vector m_k when generating the word at the t th moment, and the weighted summation yields the effect of each word in the input dialog on the words that will be generated in the decoding process m^t.

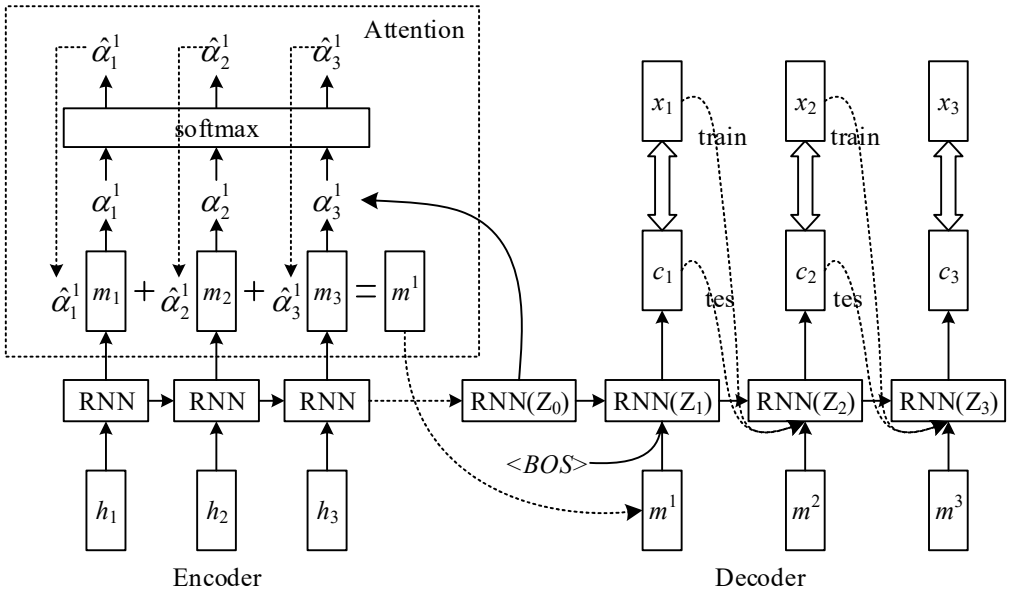

Figure 1: Generative model structure

The goal of the generative model is to maximize the probability that each output is a true reply, and the generative model is pre-trained using MLE as the loss function:

$$L(\theta) = \frac{1}{|S_1|}\sum_{(x,h)\in S_1}\sum_{t=1}^{n}\log p_\theta(x_t|\ x_{1:t-1},h) \tag{14}$$

where θ denotes the parameters of the generative model and $p_\theta(x_t|\ x_{1:t-1},h)$ denotes that the next word generated is the true reply word given the input dialog h and the true reply word $(x_1,\cdots,x_{t-1})$ the probability that the next word produced is a true reply word.

II. C. 2) Pre-trained discriminant models

The structure of the discriminative model is shown in Figure 2 as a hierarchical neural network. First, one RNN is used to encode the input h of the conversation, and another RNN encodes the real replies x or generated replies c, and the hidden state of the last moment of the RNN is used as the encoding vector of the sentence to get the sentence level information. Then, the encoding of the sentence is used as the input vector of the next layer of the RNN, and the hidden state of the 2nd layer of the RNN contains the whole dialog level information. Finally, a binary softmax layer is added for classification.

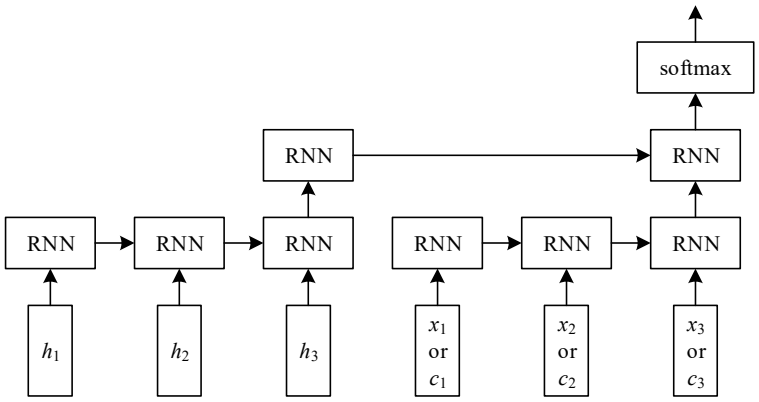

Figure 2: Discriminant model structure

Pre-training discriminative models using cross entropy as a loss function:

$$L(\phi)=-\left(\frac{1}{|S_1|}\sum_{(x,h)\in S_1}\log D_\phi(x,h)+\frac{1}{|S_2|}\sum_{(c,h)\in S_2}\log(1-D_\phi(c,h))\right) \tag{15}$$

where ϕ denotes the parameters of the discriminative model. $D_\phi(x,h)$ denotes the probability that the discriminative model judges a true response as a true response, and $D_\phi(c,h)$ denotes the probability that the discriminative model judges a generated response as a true response.

II. C. 3) Confrontation training

(1) Rewards for Generating Conversations

In this paper, we use Monte Carlo Sampling (MC-Search) method [22] to calculate the rewards corresponding to each word in the generated responses. The process of generating a dialog using MC-Search is shown in Figure 3. With the first t words $c_{1:t}$ known, the generation of the whole sentence is continued to be completed from the model distribution, generating a total of N sentences, $c^1 \sim c^N$. In calculating the reward, the average of these N sentence rewards is the reward for the t th word. This process is repeated until rewards are obtained for all words.

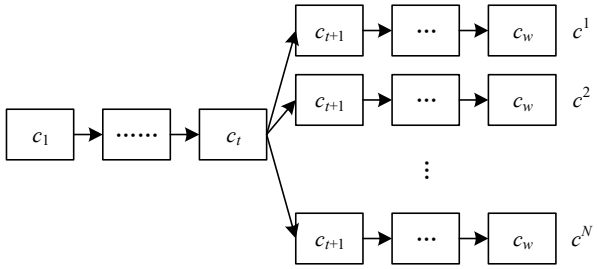

Figure 3: Using MC-Search to generate dialogue

The word reward is calculated as shown in equation (16):

$$R(c_t,h)=\begin{cases}\frac{1}{N}\sum_{i=1}^{N}D_\phi(c_t^i,h), & t<w\\ D_\phi(c,h), & t=w\end{cases} \tag{16}$$

where $R(c_t,h)$ denotes the reward for generating the word c_t, and c_t^i denotes the i th sentence of the response generated when computing the reward for the t th word using MC-Search. $D_\phi(c_t^i,h)$ denotes the probability of being judged as a true conversation when using the inputs h of the conversation and the generated replies c_t^i as inputs to the discriminative model. $D_\phi(c,h)$ denotes the probability of being judged as a true dialog when using the generated entire dialog as input to the discriminative model, using it as a reward for the last word.

(2) Training the generative model using the PPO algorithm

The process of training the generative model using the PPO algorithm is shown in Figure 4. First, the dialog input is fed into the generative model to generate responses to the dialog. Second, the input of the dialog and the generated replies are fed into the discriminative model to get the rewards. Finally, the rewards obtained from the discriminative model are used to guide the generative model to update the parameters and improve the quality of the dialog generated by the generative model. In this, MC-Search method is used to get the reward for each word.

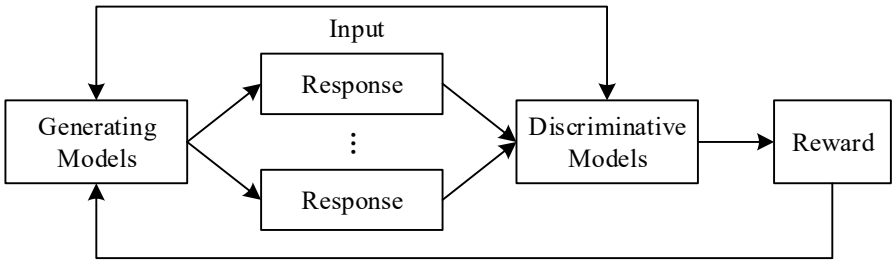

Figure 4: Adversarial training generation model

The loss function for training the generative model using the PPO algorithm is as follows:

$$L_{ppo}(\theta)=\frac{1}{|S_2|}\sum_{(c,h)\in S_2}\sum_{t=1}^{n}\min\left(m^t(\theta)R^t,clip(m^t(\theta),1-\varepsilon,1+\varepsilon)R^t\right) \tag{17}$$

Among them:

$$m^t(\theta)=\frac{p_\theta(c_t|\ c_{1:t-1},h)}{p_{\theta_{old}}(c_t|\ c_{1:t-1},h)} \tag{18}$$

$$R^t=R(c_t,h) \tag{19}$$

In Eq. (17), θ denotes the parameters of the generative model, i.e., the parameters of the strategy π. $m^t(\theta)$ is an agent objective function of the generative model, which represents the rate of change of the model parameters at the t th word of training. R^t denotes the reward for the t th word. The $clip\left(m^t(\theta),1-\varepsilon,1+\varepsilon\right)$ ensures that the rate of change of the model parameters is between $(1-\varepsilon,1+\varepsilon)$, and is $1-\varepsilon$ when the value of $m^t(\theta)$ is less than $1-\varepsilon$ is $1-\varepsilon$ when the value of $m^t(\theta)$ is less than $1+\varepsilon$ and $1+\varepsilon$ when the value of $m^t(\theta)$ is greater than $1+\varepsilon$. The function takes the smaller value between $m^t(\theta)R^t$ and $clip(m^t(\theta,1-\varepsilon,1+\varepsilon))$, which ensures that when getting a high reward, the value of $m^t(\theta)$ increases but does not exceed $1+\varepsilon$, and that the value of $m^t(\theta)$ decreases but is not less than $1-\varepsilon$ when a low reward is received. This allows the generative model to be adaptively trained in multiple iterations. In Eq. (18), $p_\theta(c_t|\ c_{1:t-1},h)$ means that given the input dialog h and the generated word $(c_1,\cdots,c_{t-1})$, the probability that the next generated word is c_t. θ is the parameter of the strategy π, i.e., the parameter of the generative model being iterated. θ_{old} is the parameter of the strategy π_{old}, i.e., the parameter of

the generative model of the last adversarial training. $(c_1,\cdots,c_t)$ is generated by the strategy π_{old}. In equation (19), $R(c_t,h)$ is the reward for generating the word c_t using the strategy π_{old}.

When using the PPO algorithm to train a dialog generation model for GAN, $m^t(\theta)$ is a proxy objective function of the generation model, and the rate of change of the model parameters is guaranteed to be within a certain range by adding a regular term, so that $m^t(\theta)$ approximates the objective function of the generation model $p_\theta\left(c_t\left|c_{1:t-1},h\right.\right)$, and the direction and step size of the parameter update of the generative model are obtained by optimizing the agent objective function $m^t(\theta)$ with regular terms. Meanwhile, since $m^t(\theta)$ is a lower bound of the generative model objective function, this ensures that the updates of the generative model parameters are monotonically non-decreasing, and the model can be better trained.

(3) Using Forced Guidance to Train Generative Models

During adversarial training, it may happen that the discriminative model is trained well enough while the generative model is not trained enough. Forced guidance can be used to avoid this situation. The training method of forced guidance is the same as that of pre-training the generative model, using the real dialogues in the dataset and using MLE as the loss function to train the generative model. This ensures that the generative model will have a real dataset to guide the training during adversarial training.

II. C. 4) Algorithmic steps

The specific steps of the PPO-GAN algorithm are as follows:

Step1: Pre-train the generative model G_θ by maximizing the loss $L(\theta)$.

Step2: Generate the dialog (c,h) using the generative model G_θ.

Step3: Pre-train the discriminative model D_ϕ by minimizing the loss $L(\phi)$.

Step4: Save the generative model parameters θ_{old} for generating responses, after which iterative training of the discriminative model with the generative model begins.

Step5: Discriminative model training, first use the generative model $G_{\theta_{old}}$ to generate dialogues, then train the discriminative model D_ϕ by minimizing the loss $L(\phi)$.

Step6: Generate dialogs using the generative model $G_{\theta_{old}}$, then use MC-Search to calculate the reward for each word.

Step7: Use the PPO algorithm to train the generative model G_θ by maximizing the loss $L_{PPO}(\theta)$.

Step8: Train the generative model G_θ by maximizing the loss $L(\theta)$ using forced guidance.

Step9: Update the generative model parameters θ_{old} used to generate responses.

III. Experimental validation of the model and analysis of the results

In order to verify the effectiveness of the proposed PPO-GAN model, this paper experimentally compares it with the Adver-REGS model that uses policy gradient to train GAN, verifies the diversity and accuracy of the model's reply quality through automatic evaluation metrics and manual evaluation, and examines the model's practical effectiveness in coherent dialog generation.

III. A. Model comparison experiments

III. A. 1) Experimental setup

(1) Experimental environment

This experiment uses Python programming language and Pytorch framework to realize the PPO-GAN model, Python has rich library resources, such as matplotlib, jieba, flask, numpy, pandas, etc., and the jieba lexicon is used in the data processing stage to lexicalize the text data, and the PyCharm platform supports CUDA acceleration, using GPU and Pytorch to accelerate the computation of the model, Anaconda3 can manage the environment resources, built-in many deep learning library resources, it is convenient to switch the virtual environment when experimenting with different models.

(2) Experimental dataset

The dataset used for the experiments in this section is a corpus of a human-computer dialogue system project and specialized service phrases, which, in this system project, saves about 500,000 real human-computer dialogue data. The specialized service phrases include Q&A data such as city and school introduction. The training data and model test data allocation are shown in Table 1, and the data processing of the unsegmented data is carried out for this experiment.

Table 1: Dataset statistics

Statistical items	Corpus
Total data	499815
Generator pre-data	485472
Adversarial training data	11033
Test data	3310

(3) Model parameter settings

The experiments set two markers e and m for reading the beginning and end of a segment in a conversation, set the number of training times to 600, the learning rate to 0.001, and set the maximum length of a single conversation to 25 using the Adam optimizer.

(4) Evaluation metrics

1) Automatic Evaluation Indicators

In the current natural language evaluation index, the BLEU algorithm is used to calculate the accuracy of the utterance.BLEU can be used to determine the degree of similarity between professional human translators and machine translators.BLEU evaluates the quality of the current sentence by outputting a value between 0 and 1.The closer the value is to 1, the better the response is. In this section of the experiment, BLEU-1, BLEU-2, BLEU-3, BLEU-4 will be used to evaluate the effect of dialog-generated replies.

2) Manual evaluation metrics

In manual evaluation, the evaluator rates grammatical correctness, utterance fluency, and relevance according to the evaluation criteria details. The output reply content is evaluated by contextual dialog information, user input and other information. Finally, the average of the evaluation scores of multiple evaluators is taken as the final score.

III. A. 2) Evaluation of experimental results

(1) Loss function

In order to verify the effectiveness of the PPO-GAN model, some of the data in the corpus is taken to train the model, and the loss function for training is shown in Figure 5. At the beginning of training, the loss function function value is large and the network is fitted faster. The loss function value reaches a relatively stable state by 600 iterations. After 600 iterations of model training, due to the presence of low-quality data in the training dataset, there will be a certain impact on the training, which will lead to small fluctuations in the loss function, but does not affect the overall quality of the training model.

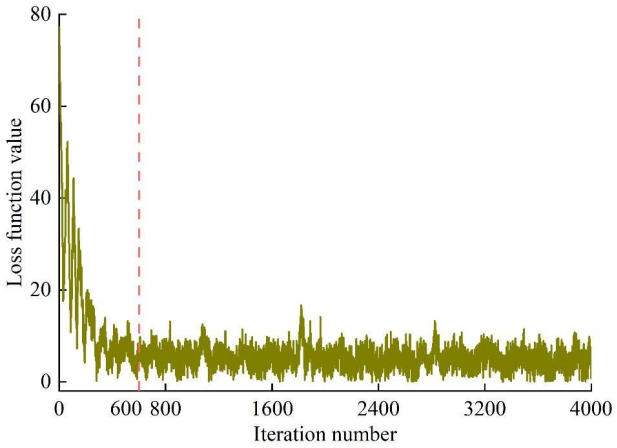

Figure 5: PPO-GAN loss function

Increasing the number of neurons in the PPO-GAN model is experimented and the loss function under different neurons is shown in Fig. 6. It can be observed that the loss function decreases less as the number of training times increases, and the loss function does not reach a steady state at 3000 times. The value of the loss function leveled off at 4000 iterations. The experiment proves the effectiveness of the PPO-GAN model in dialog training. However, increasing the number of neurons does not reduce the loss function value due to the limitations of the model structure and dataset.

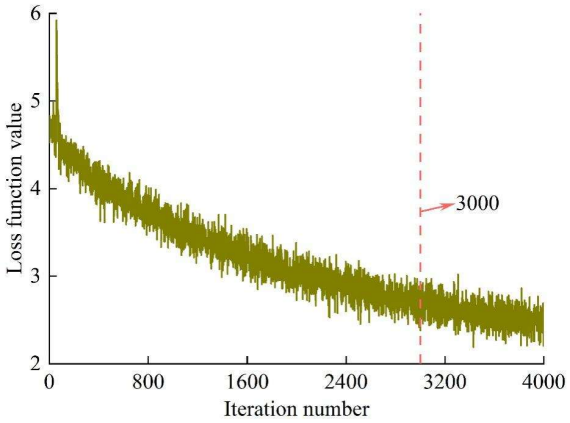

Figure 6: Loss functions under different neurons

Some data from the corpus were selected for the experiment on the PPO-GAN model. The PPO-GAN loss function in this experiment is shown in Figure 7, and the loss function reaches a relatively stable state when the iteration is up to 100 times. Between 100 and 600 iterations, the loss function has a small fluctuation, but is basically in a stable state, proving the feasibility of PPO-GAN model training.

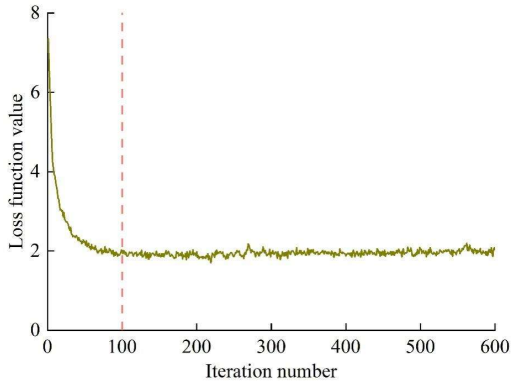

Figure 7: The PPO-GAN loss function in this experiment

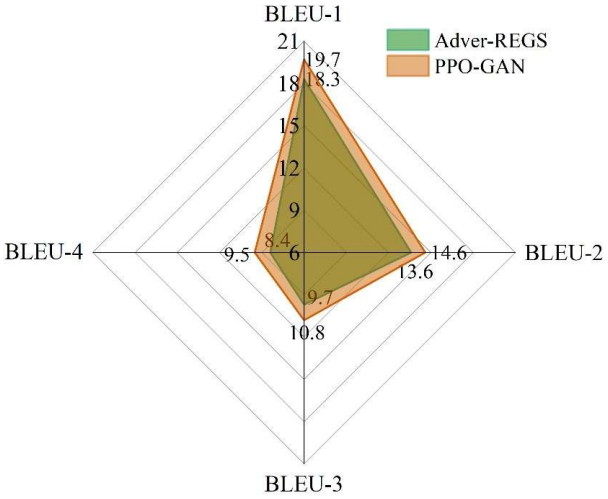

Figure 8: Automatic Evaluation Results

(2) Evaluation results

In this experiment, Adver-REGS and PPO-GAN models are experimentally compared on the experimental dataset, and the BLEU metrics are used to demonstrate the feasibility and superiority of PPO-GAN model in natural language

processing, and the automatic evaluation results are shown in Figure 8. It can be seen that the similarity metrics BLEU-1, BLEU-2, BLEU-3, and BLEU-4 of PPO-GAN are 19.7, 14.6, 10.8, and 9.5, respectively, which are higher than those of the Adver-REGS model, which indicates that the generative adversarial network model combined with the proximal policy optimization algorithm is able to continuously adjust the parameters in the generator, and can ultimately generate responses that are similar to real text similar responses, which verifies the feasibility of the PPO algorithm in generative adversarial network optimization.

In addition to the automated evaluation metrics, the experiment also used manual evaluation metrics. The content of the generated responses was rated by volunteers and the final average score was taken. The correctness, fluency, and relevance of the generated replies are scored for multiple dimensions of the Adver-REGS and PPO-GAN models.

The manual evaluation results are shown in Fig. 9, and the experimental results prove that the responses generated by the PPO-GAN model optimized based on proximal policies in this experiment are more in line with the way people communicate, with better sentence fluency and better correlation between questions and answers.

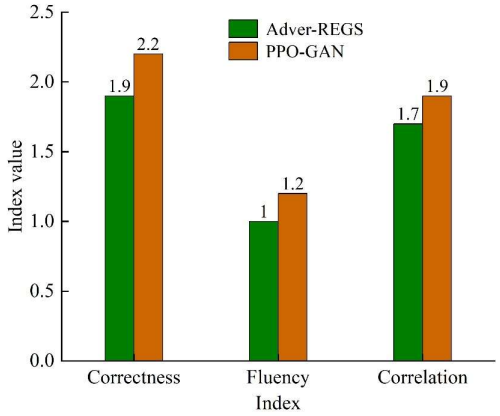

Figure 9: Manual evaluation results

In addition, it is found through experiments that Adver-REGS has the problem of secure replies and poorer responses under casual conversations, while PPO-GAN can avoid the problems of secure replies and repetitive replies, with more natural replies and higher relevance of replies, and when questions are asked using specialized service terms, PPO-GAN generates more complete replies compared to Adver-REGS.

III. B. Experiments on consistency of multi-round dialog generation

In this section, the PPO-GAN model is applied to multi-round dialogue generation consistency experiments to further evaluate the applicability of the model in multi-round dialogue generation quality optimization.

Table 2: The accuracy rates of different models on the DNLI dataset

Model	Verification set	Test set
InferSent	86.74%	86.59%
ESIM	87.25%	89.13%
Bert	88.52%	90.07%
MC-Search	89.98%	91.14%

III. B. 1) Experimental results of the MC-Search method

The experiments in this section use the dataset DNLI for dialog-based natural language reasoning, which contains 321,342 training pairs and 17,600 validation pairs and 17,600 test pairs. The performance of different models on the DNLI dataset is shown in Table 2. It can be seen that the PPO-GAN model using the MC-Search method has a better performance and achieves the optimal results on both the test and validation sets, outperforming the Bert-based model by 1.46% on the validation set and 2.73% on the ESIM model, as well as outperforming the two models on the test set by 1.07% and 2.01% in terms of accuracy, respectively. The excellent performance of the PPO-GAN model using the MC-Search method on the dataset illustrates the ability of the MC-Search method to compute the rewards corresponding to each word and guide the generative model to update its parameters, thus providing the generator with more effective coherent features and improving the quality of the dialog generated by the generative model.

III. B. 2) Results of the automatic evaluation of dialogues

To further evaluate the proposed models, this paper compares the Seq2Seq model (S2SA) [23] based on the attention mechanism, the REGS model and the RCDG model using reinforcement learning ideas, and the meta-learning based model PAML in multiple rounds of dialog generation experiments. The results of the automated evaluation of the quality of the dialogs generated by the different models and the probability of generating a contradiction in the responses are shown in Table 3. Where PPL denotes the perplexity degree, which is used to evaluate and measure the predictive ability of each natural language model. The smaller the perplexity degree, the smoother the generated responses. AVE and GRD denote the degree of similarity between the generated responses and the target responses using the vector averaging method and the greedy matching method as an evaluation matrix, respectively. DST denotes the degree of diversity obtained by calculating the different labeling ratios.

A comparison of the data shows that the conversation generation model PPO-GAN using MC-Search method achieves four optimal results. Among them, the PPO-GAN model has the smallest probability of generating contradictory responses, which is only 3.29%. On the other hand, for the other evaluation indexes, the PPO-GAN model is also more effective, compared to the traditional Seq2Seq model, the PPO-GAN model is effective because the use of MC-Search method, PPO algorithm, and the forced guidance method for the model training, which makes the quality of the model-generated replies achieved a good effect on the automatic evaluation.

Table 3: The results of automatic evaluation

Model	PPL	AVE	GRD	DST	Contradiction rate /%
S2SA	35.71	60.71	46.22	726	13.57
REGS	34.83	65.24	46.85	1015	11.64
PAML	42.52	70.38	54.31	1304	8.16
RCDG	30.81	67.85	48.35	1289	6.48
PPO-GAN	22.73	75.41	49.65	1502	3.29

III. B. 3) Manual evaluation results

In the manual evaluation of the multi-round dialog system, five fair and unbiased persons were invited to evaluate the experiments in this section. For each model 100 responses were extracted, which also included personal information and historical responses.The five persons were asked to score each response, evaluating the responses from the following three perspectives:

(1) Fluency: 1 to 3 indicates disfluency, relative fluency, and fluency, respectively.

(2) Contextual relevance: 1 to 3 indicates irrelevant, relatively relevant, and relevant, respectively.

(3) Contradiction or not: 0 and 1 indicate contradiction and no contradiction, respectively.

For fluency and contextual relevance, a weighted sum approach is used to calculate the final result.

The manual evaluation results of different models for generating multi-round dialogs are shown in Table 4. It can be seen that the dialog generation model PPO-GAN using the MC-Search method achieves the lowest inconsistency rate of 4.95%. However, it is noted through the experimental data that the responses generated by the PPO-GAN model perform poorly in terms of fluency and relevance because the introduction of the consistency feature causes the generator to take into account the consistency feature in the generation process, which affects the original encoder's features, causing the encoding-decoding message transfer to receive influence, and thus causing some impact on the fluency and relevance of the dialog. However, on the whole, the model in this paper outperforms other comparative models in terms of manual evaluation, and achieves a higher quality of dialog generation.

Table 4: Manual evaluation of dialogue quality

Model	Fluency	Correlation	Contradiction rate of manual evaluation /%
S2SA	191	137	-
REGS	204	164	10.52
PAML	225	205	7.64
RCDG	226	167	7.53
PPO-GAN	221	201	4.95

IV. Conclusion

In this paper, based on generative adversarial network and proximal policy optimization algorithm, we constructed a dialogue generation model PPO-GAN and experimentally evaluated the effectiveness of the model.

The similarity metrics BLEU-1, BLEU-2, BLEU-3, and BLEU-4 of PPO-GAN are 19.7, 14.6, 10.8, and 9.5, respectively, which are higher than those of the Adver-REGS model, demonstrating the feasibility of the PPO algorithm in generative adversarial network training. In terms of manual evaluation, the correctness, fluency, and relevance scores of the PPO-GAN model-generated replies are 2.2, 1.2, and 1.9, respectively, which are also all better than the Adver-REGS model, indicating that the replies generated by the PPO-GAN model are more in line with the way people communicate, with better sentence fluency, and better relevance between questions and answers.

In the multi-round dialog generation experiments, the accuracy of the PPO-GAN model using the MC-Search method is 1.46% and 2.73% higher than that of the Bert-based model and the ESIM-based model on the validation set, and 1.07% and 2.01% higher than that on the test set, respectively. It shows that the MC-Search method can provide more effective consistency features for the generator and improve the quality of the generated dialog produced by the generative model. Comparing the S2SA, REGS, RCDG and PAML models, the probability of generating contradictory replies for the PPO-GAN model is the smallest in both automatic and manual evaluation, 3.29% and 4.95%, respectively, and also for the other evaluation metrics, the PPO-GAN model achieves better results, which indicates that the model is more effective in improving consistency of the multilaterals' dialog generation, and can be be used as an enhancement path for the optimization of multi-round dialogue generation.

Funding

This work was supported by Henan Province Science and Technology Research Project which name is ‘Research on large model knowledge graph construction and personalized recommendation’ and ‘Research on fault diagnosis method based on transfer learning(NO. 232102220022)’, Doctoral Research Startup Project (No. 2021BS005).

References

[1] Wang, H., Wang, L., Du, Y., Chen, L., Zhou, J., Wang, Y., & Wong, K. F. (2023). A survey of the evolution of language model-based dialogue systems. arXiv preprint arXiv:2311.16789.

[2] Yin, L., Liu, X., Li, P., & Gu, H. (2024). TEMDG: A Multi-Round Dialogue Generation Model Incorporating Topic and Historical Information. IEEE Access.

[3] Li, K., Wang, J., Zhang, M., & Wang, X. (2025). OMR-Diffusion: Optimizing Multi-Round Enhanced Training in Diffusion Models for Improved Intent Understanding. arXiv preprint arXiv:2503.17660.

[4] Yin, H., Lu, P., Li, Z., Sun, B., & Li, K. (2024). MODE: a multimodal open-domain dialogue dataset with explanation. Applied Intelligence, 54(7), 5891-5906.

[5] Li, J. (2022, April). Multi-round Dialogue Intention Recognition Method for a Chatbot Baed on Deep Learning. In International Conference on Multimedia Technology and Enhanced Learning (pp. 561-572). Cham: Springer Nature Switzerland.

[6] Liu, X., He, P., Chen, W., & Gao, J. (2019). Multi-task deep neural networks for natural language understanding. arXiv preprint arXiv:1901.11504.

[7] McShane, M. (2017). Natural language understanding (NLU, not NLP) in cognitive systems. AI Magazine, 38(4), 43-56.

[8] Li, Y., Shen, X., Yao, X., Ding, X., Miao, Y., Krishnan, R., & Padman, R. (2025). Beyond Single-Turn: A Survey on Multi-Turn Interactions with Large Language Models. arXiv preprint arXiv:2504.04717.

[9] Zheng, Y., Chen, G., & Huang, M. (2020). Out-of-domain detection for natural language understanding in dialog systems. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 28, 1198-1209.

[10] Abro, W. A., Aicher, A., Rach, N., Ultes, S., Minker, W., & Qi, G. (2022). Natural language understanding for argumentative dialogue systems in the opinion building domain. Knowledge-Based Systems, 242, 108318.

[11] Tian, J., Tu, Z., Li, N., Su, T., Xu, X., & Wang, Z. (2022). Intention model based multi-round dialogue strategies for conversational AI bots. Applied Intelligence, 52(12), 13916-13940.

[12] Tsinganos, N., Fouliras, P., & Mavridis, I. (2023). Leveraging dialogue state tracking for zero-shot chat-based social engineering attack recognition. Applied Sciences, 13(8), 5110.

[13] Wang, Y., He, T., Mei, J., Fan, R., & Tu, X. (2022). A stack-propagation framework with slot filling for multi-domain dialogue state tracking. IEEE transactions on neural networks and learning systems, 35(1), 1240-1254.

[14] Liao, L., Long, L. H., Ma, Y., Lei, W., & Chua, T. S. (2021). Dialogue state tracking with incremental reasoning. Transactions of the Association for Computational Linguistics, 9, 557-569.

[15] Li, Q., Zhang, W., Huang, M., Feng, S., & Wu, Y. (2023). RSP-DST: Revisable state prediction for dialogue state tracking. Electronics, 12(6), 1494.

[16] Khan, M. A., Huang, Y., Feng, J., Prasad, B. K., Ali, Z., Ullah, I., & Kefalas, P. (2023). A multi-attention approach using bert and stacked bidirectional lstm for improved dialogue state tracking. Applied Sciences, 13(3), 1775.

[17] Heck, M., Lubis, N., Niekerk, C. V., Feng, S., Geishauser, C., Lin, H. C., & Gašić, M. (2022). Robust dialogue state tracking with weak supervision and sparse data. Transactions of the Association for Computational Linguistics, 10, 1175-1192.

[18] Hu, Q., Yang, Y., Zhang, Y., & Zheng, J. (2023). The Advance of Multi-Round Dialogue System with Deep Learning. Applied and Computational Engineering, 8, 693-700.

[19] Xiaoming Duan, Yagiz Savas, Rui Yan, Zhe Xu & Ufuk Topcu. (2025). On the detection of Markov decision processes. Automatica,175,112196-112196.
[20] Yaohuan Wu & Nan Xie. (2025). Design of digital low-carbon system for smart buildings based on PPO algorithm. Sustainable Energy Research,12(1),9-9.
[21] Yasaman Tavakol Moghaddam, Mehrdad Boroushaki & Majid Astaneh. (2024). Reinforcement learning for battery energy management: A new balancing approach for Li-ion battery packs. Results in Engineering,23,102532-102532.
[22] Iman Goroohi Sardou & Ali Goroohi. (2017). Hybrid model for stochastic clearing of joint energy and reserves market. IET Generation, Transmission & Distribution,11(10),2608-2617.
[23] Zhou Gaoyu, Hu Guofeng, Zhang Daxing & Zhang Yun. (2023). A novel algorithm system for wind power prediction based on RANSAC data screening and Seq2Seq-Attention-BiGRU model. Energy,283.