DCFGAN: An Adversarial Deep Reinforcement Learning Framework with Improved Negative Sampling for Session-based Recommender Systems

Jianli Zhao[1*], Hao Li[1], Lijun Qu[1], Qinzhi Zhang[1], Qiuxia Sun[2], Huan Huo[3], Maoguo Gong[1,4*]

[1] College of Computer Science & Engineering, Shandong University of Science and Technology Qingdao, China

[2] College of Mathematics and Systems Science, Shandong University of Science and Technology Qingdao, China

[3] Faculty of Engineering and Information Technology University of Technology Sydney, Australian

[4] Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, Xidian University, Xi'an 710071, China

*Corresponding author : Jianli Zhao, Maoguo Gong

Abstract

In recent years, with the development of Internet technology, recommender systems have been widely used by virtue of their ability to meet the personalized needs of users. In order to make full use of users' interactive behaviors, session-based recommender systems have attracted growing research interest. In previous session-based recommender systems, users' historical interactive behavior is utilized to train and update users' preferences, but users' responses to the current recommendation results (immediate feedback) are not effectively exploited to optimize the recommendation strategy. This leads to the decrease of subsequent recommendation accuracy. Aiming at this problem, based on the Recurrent Neural Network (RNN), this paper combines Reinforcement Learning (RL) and Generative Adversarial Networks (GANs). We fully exploit the users' immediate feedback with RL, and simultaneously take advantage of GAN to satisfy the requirement of training data brought by RL. Furthermore, we optimize the negative sampling method and propose Deep Generative Adversarial Networks-based Collaborative Filtering (DCFGAN). The experimental results show that this algorithm can effectively improve the recommendation accuracy in session-based recommender systems.

Keywords: Session-based Recommender Systems, Reinforcement Learning, Generative Adversarial Networks, Recurrent Neural Network, Negative Sampling

1. Introduction

With the rapid information explosion on the World Wide Web, the advantages of recommender systems in alleviating the problem of information overload have become increasingly prominent. Recommender systems, which have been applied in all walks of life, are gradually being associated with deep learning and social relationship analysis [1]. Traditional recommender systems usually exploit the accumulated user interactive behavior to generate recommendation results and update the recommendation strategy in the stages when more interactive behavior is obtained. However, this recommendation mode ignores the user's immediate feedback. On the other hand, with the development of Internet technology, there is a rising demand for better user experience, which forces recommender systems to pay more attention to the accuracy of the recommendation results. Thus, the current task of recommender systems is to utilize the user's various feedback, especially immediate feedback, to accurately grasp the user's interests from the user's fragmented interactive behavior. Session-based recommender systems (SBRS) are commonly used in this aspect because they can take full advantage of users' immediate feedback. These regard a series of user behavior (continuous interaction within a short period of time) as one session and complete personalized recommendations by sequentially handling this session. Users generate real-time behavior data through the current session; if recommender systems cannot handle the data as soon as it is generated, it will cause a lack of timeliness and lead to a decrease in the accuracy of recommendation results and decline in the user experience [2]. Therefore, in session-based recommender systems, when a user interacts with the system, it is highly important for the recommendation algorithm to model and apply the immediate feedback.

The current models commonly used for session-based recommender systems are Recurrent Neural Network (RNN) [5] and reinforcement learning models [8]. The RNN-based models [4, 10] can be used to process sequential data in session-based recommender systems. As the training progresses, the front hidden layer in the RNN model will affect the next hidden layer, taking the user's historical behavior as input and spreading through multi-layer networks to achieve the purpose of predicting the user's interests. For example, as the first deep learning approach to SBRS, GRU4REC [38] and models [39] that combine different types of information can both model the entire session. However, these models do not easily make effective use of immediate feedback [35]. On the other hand, reinforcement learning methods [6, 7] are Markov chain-related, i.e., based on the assumption of Markov independence and utilize the user's previous behavior to predict the user's next behavior [3], such as DRN [37], and the Pseudo Dyna-Q model [34]. Nonetheless, a critical bottleneck of reinforcement learning is the unstable convergence of the models in the training process. In practice, the negative impact of reinforcement learning and RNN is the increased demand for training data by the system, which is a particularly evident challenge in the originally sparse session-based recommender systems.

One solution is to generate part of the data through Generative Adversarial Networks (GANs) [13] for the reinforcement learning method, thereby breaking the limit of insufficient data. For example, CFGAN [14] combines generative adversarial networks with collaborative filtering, and it optimizes the discrete value processing in the training process to improve the fitting degree of user preferences. However, these methods apply the policy gradient (PG) in the gradient propagation step, which cannot effectively use immediate feedback and solve the problem of data dispersion,

resulting in instability and low convergence efficiency in the training process. In addition, the negative sampling method used to indicate that the user dislikes the items is random sampling, which cannot fully express the user's preferences. Random negative sampling is to draw negative samples from items that the user has not interacted with, and the number of samples is small. However, such items may include both user likes and dislikes. In this process, it is very easy to extract the items that users like as negative samples, which cannot effectively express user preferences in session-based recommender systems.

To solve the aforementioned problems, we propose a Deep Generative Adversarial Networks-based Collaborative Filtering (DCFGAN) model, which combines the Q-learning and the Actor-Critic models commonly used in reinforcement learning. The main contributions of this paper are summarized as follows:

(1) To address the instability of the training process caused by the uncertain probability problem in the PG algorithm used in the previous SBRS models, we employ the Deep Deterministic Policy Gradient (DDPG) algorithm to return the gradient, thus enhancing the stability of the training process. At the same time, the use of DDPG to optimize the value function formula reduces the number of iterations required for convergence.

(2) Since the original random negative sampling method cannot effectively improve the accuracy of the recommendation results in session-based recommender systems, a new negative sampling method is proposed. It uses collaborative filtering pre-training to negatively sample items with low user interest in advance, which effectively improves the accuracy of negative sampling and is thus more suitable for recommendation scenarios.

(3) For the original GAN-related model, the state element is used as the training input, which cannot effectively fit the user's upcoming feedback. Therefore, we improve the traditional GAN in training the generator network G by experience replay. The improved GAN does not only input the state element, but expands it to a tuple (s, a, r, s'). Then, the experience replay method is used to store the behavior data that the system has conducted with the user through the tuple (s, a, r, s'), and these tuples are randomly taken for training, so that the generator network G can better fit the user's interest.

The rest of this paper is organized as follows: The technology and research related to this work are introduced in Section 2. Details of the principle of DCFGAN and the main content of its implementation are elaborated in Section 3. Then, we conduct a comparative experimental analysis in Section 4. In the final section, our work is recapitulated.

2. Related Works

In the currently applied recommender systems, user data are mostly presented through sessions. Different from traditional recommender systems, interactive behaviors present sequentiality and are concentrated as sessions. In addition, the session-based system often has interactive behavior only for a single user but less information about the user's attribute and the related user. Modeling user behavior as a session can help the system sequentially extract the features of this user from their recent historical interactive behavior and generate recommendation results when their related information is poor. This advantage has made session-based recommender systems a research hotspot in recent years [2]. In these systems, user behavior data are distributed in a sequence. Due to the temporal logic relationship of the data, the problem that common methods like collaborative

filtering and matrix factorization cannot retain the disadvantages of sequential logic is exposed. Session-based recommender systems often use algorithms that can effectively process sequential data and perform training by retaining the sequential logic of user data. Thus, these systems can effectively apply user behavior directly in the context of the current session and iterate the recommendation strategy for the user to update the recommendation algorithm.

The current mainstream algorithms in session-based recommender systems [2] are recurrent neural networks and related algorithms (LSTM, GRU, GNN etc.). Recurrent neural networks effectively process sequential data by using the output of the previous network cell as the input of the next cell. Finally, RNN trains data for the entire session and outputs recommendation results. LSTM and GRU go one step further and combine the long-term interest of users with short-term behavior through the gates to optimize the recommendation results, thereby improving the recommendation accuracy [10]. With the development of graph networks in session-based recommendation, many state-of-the-art algorithms have been combined with the graph neural network. SR-GNN [25] provides more accurate recommendations by analyzing and training directed graphs generated from the users' behavior sessions. DGTN [26] analyzes the preference migration of a single user between different sessions and merges neighbor sessions to achieve higher recommendation accuracy.

Reinforcement learning, which has the additional ability to process sequential data, has been widely used in various fields of artificial intelligence[40][41]. In recommender systems, reinforcement learning algorithms can utilize user feedback to model the user's interests. In recent years, there have been attempts to use reinforcement learning methods in session-based recommender systems. The basic reinforcement learning algorithms, Q-learning, SARSA [15], Policy Gradient [16], and other methods [34] based on the Markov decision process, can be used to carry out the algorithm improvement of session-based recommender systems. At the same time, with the popularity of deep learning and its adaptability to reinforcement learning, deep reinforcement learning algorithms such as DQN [17] and DDPG have also become increasingly concerned [18][19]. Munemasa et al [9] combined reinforcement learning with the session-based recommender system and used the Markov process to optimize the recommender system. For news recommendation, Zheng et al [37] proposed a new Reinforcement Learning Framework to model the dynamic nature of news features and user preferences. Considering the problem of high computational expense of offline learning algorithms, Zou et al [34] proposed the Pseudo Dyna-Q model to simulate the environment and handle the selection bias of logged data. However, reinforcement learning often requires high data volumes to train for practical applications, and the data of the recommender systems are often sparse. That is to say, recommender systems cannot meet the sufficiency requirements of the training process of reinforcement learning, resulting in insufficient recommendation accuracy.

Subsequently, more and more researchers have begun to pay attention to the data augmentation problem of recommender systems, and have gradually started to use deep learning models (such as VAE [20] and GAN [13], etc.) that can solve this problem. Since GAN has proved its potential to learn from large datasets, it is naturally applied to optimize recommender systems[42-44]. The GAN model establishes two network structures: a generator network G and a discriminator network D. The former generates fake data through random generation or pre-training generation strategies and mixes the generated data with the real data as the input of D. Meanwhile, D judges the validity of each piece of data based on the loss function. The two networks continue to fight against each other,

which ultimately makes the fake data generated by G have the best fit to the real data. Then, the trained generator network is applied to continue to generate data, so as to compensate for experiments with insufficient data. For example, DA-GAN [21] obtains data from the source domain and learns to obtain and aggregate any data items to generate more data. Regarding practical work, SimRec [22] combines the GAN model with the commonly used RNN model in session-based recommender systems to improve the applicability of the recommendation. Notably, most data augment methods focus on continuous data (such as images and video) and rarely use models to generate discrete data, however, the interactive data in recommender systems is often discrete. The SeqGAN [23] combines the PG algorithm and the GAN model to solve the problem that it is difficult to return the user's discrete feedback when using the traditional GAN, which greatly improves the stability and accuracy of the GAN model in recommender systems. Subsequently, IRGAN [24] was employed, which realizes the game relationship in GAN by optimizing the loss function. In addition, in order to better obtain the user's preference in a certain recommendation session, this method models the user's behavior generated by GAN training, whose value in the vector is no longer just simple purchase behaviors of 0 and 1. The optimized value is filled with similar rating information that can more clearly reflect the user's preference. This move further transforms the discrete feedback of the recommender system into a non-discrete value to improve the training stability. Based on the algorithm, CFGAN [14] employs several collaborative filtering methods commonly used in recommender systems to improve the accuracy of recommendation results.

In the above method, SimRec and seqGAN mainly improve the GAN on the network structure model [45], and make full use of the sequential data in session-based recommender systems; meanwhile, IRGAN and CFGAN improve the loss function based on seqGAN and IRGAN. However, these methods still have problems. GAN has poor performance when processing discrete data and it has not been effectively improved in SimRec. SeqGAN was a milestone in introducing the reinforcement learning method Policy Gradient to solve this problem, but the instability of training and the uncertainty of the probability caused by the PG algorithm make seqGAN underperform in the recommender system. IRGAN describes the priority order or prediction score of recommended items so that the training data of GAN is no longer discrete at 0, 1, and it better adapts to the recommender system, but this algorithm does not use sampling methods and fails to grasp the user preference in the recommender system in advance. CFGAN directly uses continuous real values between 0 and 1 to obtain a more optimal solution to the discrete input problem, but this algorithm cannot fully utilize the information that the user has interacted with, and the CF method is randomly sampled such that user preference cannot be accurately represented. SR-GNN needs to build a graph structure that is more verbose than a single session, which may result in that SR-GNN cannot train effectively when the data for a single user are insufficient.

3. Deep Generative Adversarial Networks-based Collaborative Filtering Model

In this section, we first introduce the background of recommender systems, including the algorithm environment and the GAN models. Then, we analyze the remaining problems and propose the method of this paper. Afterwards, we describe the negative sampling method, the DDPG method, and the experience replay method in the order of the problems, solving them one by one. Lastly, we present the proposed algorithm's pseudocode in a list style.

3.1 Reinforcement learning and GAN method in recommender systems

3.1.1 Reinforcement learning

Reinforcement learning emphasizes how to act based on the environment and maximize the expected rewards. The basic elements (environment) of reinforcement learning include state, action, reward, discount rate, etc. In this paper, we take reinforcement learning as a modeling idea. Through continuous trial and exploration, the recommender system is trained from completely random prediction to accurate prediction. In the session-based recommender system, we model user behavior data and the recommender system as a reinforcement learning environment. This environment is set as follows [27]:

State space S: $state\ s_t \in S$, which represents the user's interacted data at time t;

Action space A: $action\ a_t \in A$, that directly corresponds to s_t, which means an item sequence recommended by the recommender system to the user in the current state;

Reward R: it represents the feedback $r(s_t, a_t)$ provided by the user when the recommender system recommends items to the user at time t;

Discount γ: the discount rate hyperparameter, which is used to adjust the proportion of immediate feedback.

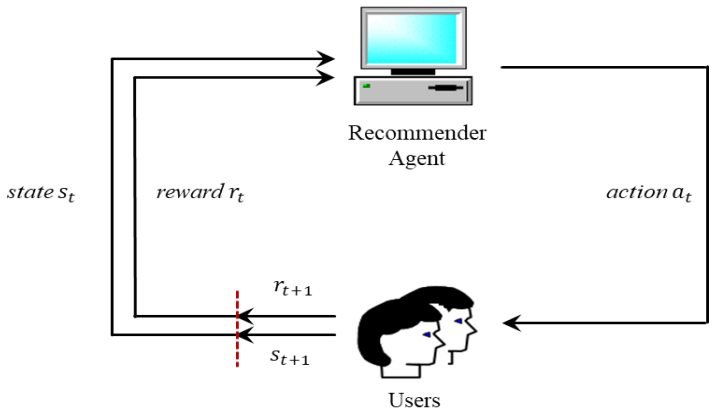

Figure 1. Reinforcement learning in the recommender system

Figure 1 depicts the reinforcement learning process in the recommender system. The recommendation strategy predicts the user behavior at time *t*+1 based on the user's interactive behavior at time *t*. Finally, reinforcement learning adjusts the recommendation strategy according to the user's feedback formed by real behavior and predicted behavior. In this paper, we completed the training through such modeling idea, and prepared for the subsequent increase of data volume and improvement of training data dimension.

3.1.2 GAN model

In order to improve the training accuracy and address the data volume requirements of reinforcement learning, we regard GAN as the agent in reinforcement learning. It is an adversarial model, including generator network G and discriminator network D. Network G generates data that can obfuscate network D, and network D judges the probability that the input data are true or false. The prediction is achieved through the adversarial between G and D. The main training process for the GAN model is to continuously minimize two loss functions:

$$L_D = -\left\{\mathbb{E}_{s\sim p_{data}}\left[\log D_\phi(s,a)\right] + \mathbb{E}_{s\sim p_{data}}\left[\log D_\phi\left(s, G_\theta(s)\right)\right]\right\} \tag{1}$$

$$L_G = \mathbb{E}_{s\sim p_{data}}\left[\log D_\phi\left(s, G_\theta(s)\right)\right] \tag{2}$$

Among them, $G_\theta(s)$ is the simulated behavior data generated by network G to the user according to the current state; s is the state in reinforcement learning, which represents the user's interacted data at the current time; a is the action, which means an item sequence recommended by the recommender system to the user in the current state.

The IRGAN algorithm [24] further improves the model formula into a maximum and minimum optimization game problem:

$$J = \max_\theta \min_\phi \sum_{n=1}^{N}(\mathbb{E}_{x\sim p_{data}}\left[\log D(x)\right] + \mathbb{E}_{\hat{x}\sim p_\phi}[\log(1-D(\hat{x}))]) \tag{3}$$

In order to prevent the local over-fitting of generator G and obvious errors in generated data (such as feature vectors with a value of 1 in each dimension), CFGAN [14] adds the masking blocks to discriminator D. Only the items actually interacted by the user are taken to reduce the influence of irrelevant items on discriminator D:

$$\begin{aligned} J^D &= -\mathbb{E}_{x\sim P_{data}}[\log D(x|c)] - \mathbb{E}_{\hat{x}\sim P_\phi}\left[\log\left(1-D\left(\hat{x}|c\right)\right)\right] \\ &= -\sum_u \log D(r_u|c_u) + \log\left(1-D\left(\left(\hat{r}_u \odot e_u\right)|c_u\right)\right) \end{aligned} \tag{4}$$

Correspondingly, the loss function of G also changes:

$$J^G = \sum_u \log\left(1-D\left(\left(\hat{r}_u \odot e_u\right)|c_u\right)\right) \tag{5}$$

Furthermore, the collaborative filtering method is used to take out the product items of the user's interacted behavior for training. At the same time, CFGAN adds regularization to obtain the final loss function of the method:

$$J^G = \sum_u\left(\log\left(1-D\left(\hat{r}_u \odot (e_u + k_u)|c_u\right)\right) + \alpha_2 \cdot \sum_j\left(x_{uj} - \hat{x}_{uj}\right)^2\right) \tag{6}$$

$$J^D = -\sum_u\left(\log D(r_u|c_u) + \log\left(1-D\left(\hat{r}_u \odot (e_u + k_u)|c_u\right)\right)\right) \tag{7}$$

where r_u, $\hat{r}_u$ represents the user's interactive data in the real dataset and the fake data generated by G; e_u denotes the mask vector, which takes the value of 1 when it is the user's actual interacted item, otherwise it is 0. The parameter x in this research is built as s in this paper.

The above method exposes the problem that the system cannot effectively train items which the user has not interacted with, and the original negative sampling has little effect in such a recommender system.

3.2 The proposed method

In session-based recommender systems, compared with the traditional recommendation system, the sparsity of the training data still remains. Many researchers attempted to solve this problem by using the GAN algorithm. The CFGAN method indeed has greatly alleviated this issue, however, with the widespread use of PG and the method of training for the user's interacted information, the

idea above still presents the following problems: 1) The negative sampling method in CFGAN directly uses random sampling commonly used in natural language processing, which easily leads to the decline of sampling accuracy in the recommender system; 2)The PG algorithm is an uncertain probability selection problem, and its use for gradient back-propagation will lead to the lack of stability in training; 3) The generator network G does not effectively utilize the user's immediate feedback to optimize the training process.

In order to solve these problems, we propose DCFGAN, with an architecture shown in Figure 2. In this framework, the model training procedure is divided into three stages: pre-training, training, and training up to K times (i.e., a training threshold). In the first stage, a collaborative filtering algorithm is used to pre-train user behavior, and negative training samples are collected from the items with low prediction ratings. These negative training samples and real data are then used to pre-train generator G and discriminator D. In the training stage, generator G generates fake data based on negative samples and positive samples that have had interactions; meanwhile, discriminator D predicts the probability of items being true or false based on the output of the generator G. Discriminator D obtains feedback by comparing the predictions with the real data and steadily returns it to the generator through the DDPG method, which feedback will be stored in the experience pool. In the third stage, when the training number reaches K, the experience pool extracts the data and feeds it to generator G with the strategy obtained by the previous training. Experience replay is added during the training process to further improve the recommendation accuracy.

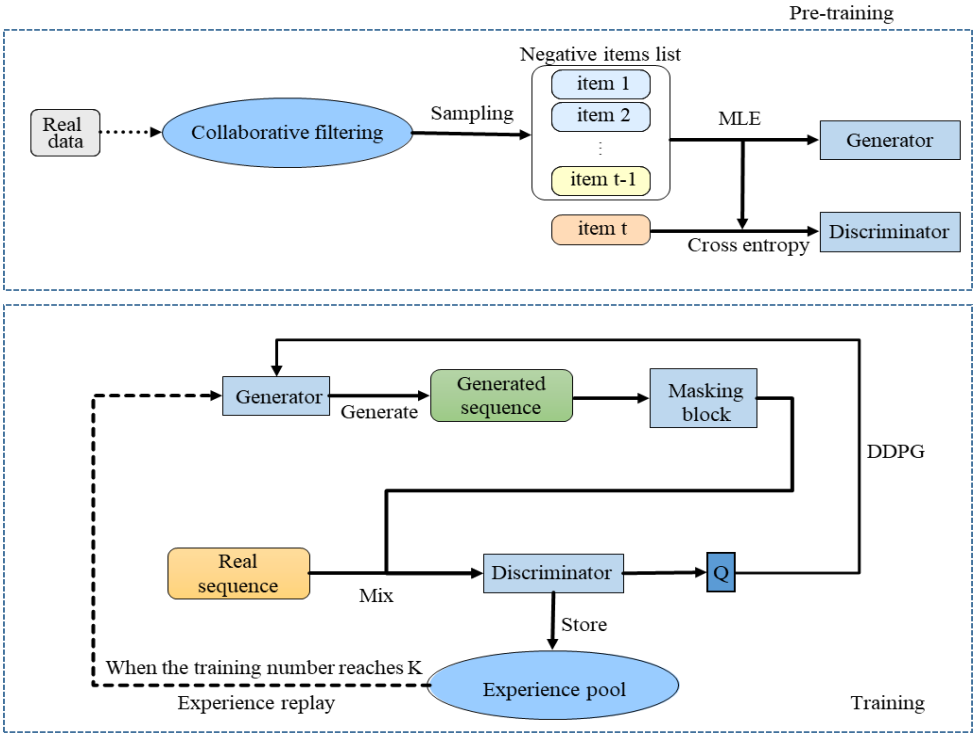

Figure 2. The architecture of our DCFGAN model framework

Correspondingly, to solve the first problem, we propose a new negative sampling method explained in Section 3.2.1. To solve the second problem, we use the DDPG method described in Section 3.2.2. To solve the third problem, we improve the generator network G of traditional GAN and introduce the experience replay method in Section 3.2.3.

3.2. 1 Negative sampling method

The negative sampling method is commonly used in CFGAN [32]. In the pre-training process,

each item that has not been interacted by the current user may be marked as a negative sample, so that the generator network G tries best not to recommend these items to the user according to the recommendation strategy. However, an item that the user has not interacted with is not equivalent to a negative sample. Such items include two categories: the user's likes and their dislikes. Taking items that the user has not interacted with directly as a negative sample will make the learned user preferences inaccurate.

In this paper, we propose a new negative sampling method to solve the above problem. This method first pre-trains the data and collects user interests through collaborative filtering based on real data. Then, a few samples with low user interest are marked as negative in advance. At the same time, the number of negative samples η is expanded instead of the 5 or 10 commonly used in CFGAN, such that η can be more in line with the scene of the session-based recommender system and maximize the influence of the negative sampling method.

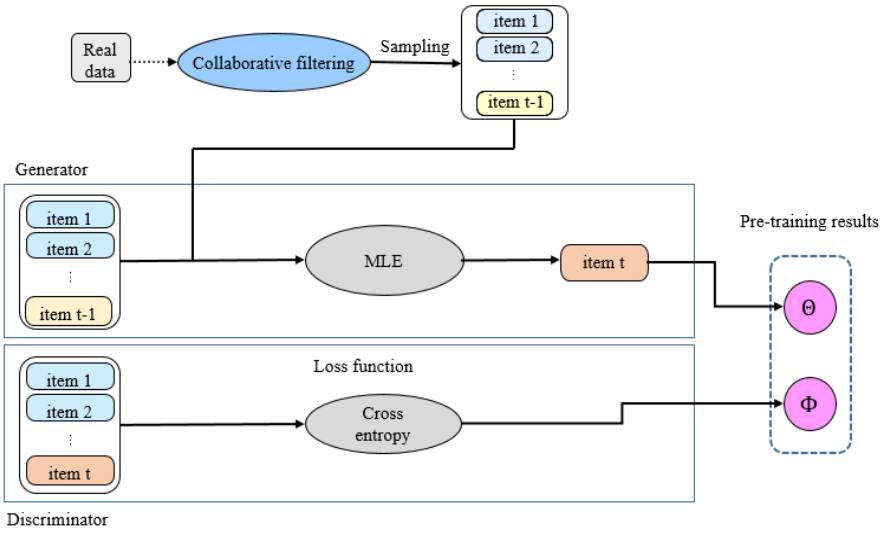

Figure 3. The improved negative sampling method and pre-training process

3.2. 2 DDPG method

The DDPG is a gradient back propagation algorithm. According to the update history of the reinforcement learning algorithm given in Section 2, compared with the PG algorithm, the DDPG adds deterministic probability selection [28], which guarantees the training stability of deep reinforcement learning. In addition, DDPG adopts the classic idea of Actor-critic [29], which not only relies on the action selection method based on probability but also combines the idea of Q-learning [30]. Through the long-term feedback changes of the value function, recommender systems evaluate and guide the selection of historical actions, in order to optimize the actor's strategy to choose action [31]. The update strategy is shown below.

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N} \sum_{i} \nabla_{a} Q(s, a \mid \theta^{Q}) \mid_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu\left(s \mid \theta^{\mu}\right) \mid_{s_i}$$

$$\theta \leftarrow \theta + \nabla_{\theta} J\left(\theta\right) \tag{8}$$

Regarding the Actor, its parameter updating will also involve the Critic. It can be known from the above formula that the first half of it comes from the Critic, that is, how to choose the action of the Actor this time to obtain a greater Q. The latter part comes from the Actor, that is, how the Actor should update its own parameters, so that the Actor has a greater chance to choose this action.

Therefore, the entire DDPG becomes a combination of the two, that is, the Actor should update the action parameters in a direction which is more likely to obtain a larger Q.

The above combination improves the update iteration efficiency of the reinforcement learning algorithm and reduces the number of iterations needed to reach convergence. Therefore, this work first replaces the PG algorithm, commonly used in GAN models for returning the gradient with the DDPG algorithm, using a dynamic game between two network models to complete the learning process with the goal of improving the stability and efficiency of the training process.

3.2.3 Experience replay method

Experience replay is a commonly used technique in reinforcement learning. It is used to adjust the timing of training data input, such that the user's long-term interest model can be better grasped. In this method, the results will not be over-fitted to recent user interests. In CFGAN, the training model already uses the GRU model, which has a certain grasp of the user's long and short-term memory, and the traditional advantages of experience replay will be no longer obvious.

The previous methods often only used existing user behavior data to train user preferences, which resulted in subsequent sessions not having a major impact on the current session. Herein, we use the tuple (s, a, r, s') stored in the experience pool for training data during the GAN training process, which originally used only the current state as the input. In this way, other than the traditional GAN training process, a threshold training number K of experience replay is set. When the training number reaches K and its integer is multiple, the generator model G is re-trained with random data in the experience pool. The purpose of this method is to fit some of the historical records stored in the experience pool, so that G's training will be more adequate. The experiment shows that, when the experience replay of G is added, the accuracy of the recommendation results will be improved compared to the original training process.

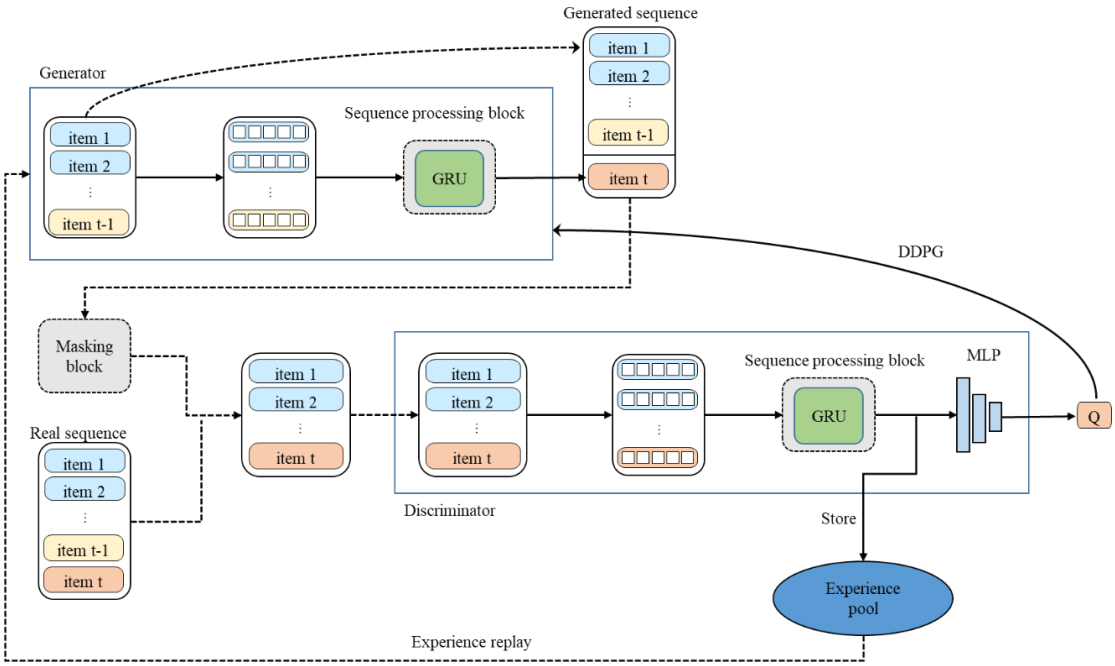

Figure 4. The algorithm training process and experience replay method

As shown in Figure 3, DCFGAN learns user behavior in the pre-training process and completes the collection of negative training samples. Through the network model, the negative sample information is transmitted to the generator and the discriminator for confirmation, hence the subsequent training process can recommend items other than the negative samples to the user as

much as possible.

Next, we make a prediction of the user's next choice based on the existing session data in the generator and propagate the result as input to the discriminator. Then, the discriminator obtains feedback by comparing it with the real data, and stably returns it to the generator by the DDPG method. This process is repeated to train the model parameters, and experience replay is added during the training process to further improve the recommendation accuracy, which is rendered in Figure 4.

3.2.4 Algorithm pseudocode

Algorithm 1 DCFGAN

Input: dataset P_{data} ,generator policy G_θ , roll-out policy G_β , discriminator D_ϕ ,learning rate and batch-size M

Initialize G_θ, D_ϕ with random weights θ, ϕ
for each $u \in U$ **do**
    Sample N_u
end for
Pre-train G_θ using N_u and MLE on P_{data}
$\beta \leftarrow \theta$
Generate fake sequences using G_θ for training D_ϕ
Pre-train D_ϕ via minimizing the cross entropy
Initialize experience replay pool E
repeat
    for g-steps **do**
        Generate a_t
        Compute $Q(s_t, a_t)$
        Update generator parameters via DDPG Eq. (8)
        Train generator G_θ by Eq. (6)
    end for
    for d-steps **do**
        Use G_θ to generate fake sequences and combine with P_{data}
        Train discriminator D_ϕ by Eq. (7)
        Add current tuple (s_t, a_t, r_t, s_{t+1}) into E
    end for
    if training-epochs arrives K
        Take M tuples from E for retraining G_θ
    $\beta \leftarrow \theta$
until converges

Due to the optimizations made to traditional session-based methods in this work, the effective improvements are the accuracy and stability of the proposed method. On the other hand, at the initial stage of training, there is less user behavior information, and the deep reinforcement learning method DDPG can explore and grasp user preferences more quickly and effectively. With the

training stability brought by DDPG, our method can often quickly fit the user's interest and constantly update and iterate the preferences in the training process. These ensure that our method has a significant improvement effect on sparse samples. It is worth noting that our method focuses on the improvement of recommendation accuracy rather than the cold start problem. Moreover, at the cost of higher accuracy, the time complexity of our method has been improved.

4. Experiment

In this section, we conduct experiments to answer the following research questions:

RQ1 Does our proposed DCFGAN model outperform the state-of-the-art recommendation algorithms?

RQ2 How is the performance of DCFGAN impacted by different choices of hyperparameter?

RQ3 How does DCFGAN preform against the popular SR-GNN model with different numbers of sessions?

RQ4 Which part plays a key role in the DCFGAN algorithm modification in this paper?

Next, we present the datasets and experimental settings in Section 4.1. Then, we make a performance comparison with and analysis of the state-of-the-art recommendation algorithms.

4.1 Experimental design

The experiment uses the YOOCHOOSE dataset, which is the public dataset of RecSys Challenge 2015. It contains users' session behavior data, which mainly includes users' purchase data; it uses 0, 1 to indicate purchase or not, and 1-12 to indicate star rating. The dataset is divided into 8:2 and is cross-validated five times.

In order to ensure the validity of the dataset, we randomly selected a certain number of session data from YOOCHOOSE. We eliminated the sessions with user interaction behavior less than 5 in a single session and re-selected the sessions that meet the requirements. In the comparative experiment between DCFGAN and each algorithm, we randomly selected 100,000 sessions and randomly divided them into five equal parts to ensure that the training set and testing set have a ratio of 8:2. This method is uniform for all methods involved in the experiment.

Table 1. Contents of the YOOCHOOSE dataset

Dataset	Sessions	Users	Items	Purchases
YOOCHOOSE	100000	6694	10843	341956

The experiment first compares various basic methods, existing improved methods and the DCFGAN method proposed in this article. Then, it compares various improvement strategies existing in DCFGAN one by one, confirms the effectiveness of each improvement strategy, and calculates the degree of improvement of the results.

Table 2. Hyperparameter settings

Hyperparameter	Setting
Number of batches M	100
Learning rate α	0.1
Empirical replay threshold K	1000
Number of negative samples η	600

In order to unify the network structure of each model derived from GAN, the RNN model of SimRec is changed to a GRU network, so that each GAN model is trained using the GRU network

model to increase the credibility of the comparison results. For the experience replay threshold K in the improved method, through the value test with a step size of 500, it is found that when K>1,000, the method has less influence on the recommendation result. Meanwhile, when the value is between 500-1,000, there is almost no impact. At this time, taking K=1,000 can reduce the computational complexity of experience playback as much as possible while ensuring the recommendation effect. For the 10,843 items in total, the negative sample is determined by the value test with a step size of 50, and the optimal result is finally obtained when the number of negative samples is η=600.

The metrics used in the experiment are as follows:

(1) Root Mean Square Error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{r_{uic} \in R} (r_{uic} - \hat{r}_{uic})^2}{|R|}} \tag{9}$$

(2) Precision:

$$P = \frac{1}{n} \sum_{u \in U} \frac{L_u \bigcap B_u}{L_u} \tag{10}$$

where L_u denotes the list of items recommended to the user *u* (the recommended number in the experiment is 1); B_u is the set of items that interact with the user *u* in the testing set.

4.2 Comparisons with the state-of-the-art recommendation algorithms

For **RQ1**, we compare the accuracy of the proposed method with those of the state-of-the-art recommendation methods, which are listed as follows:

- DDPG [11]: It is a simple attempt of the deep reinforcement learning algorithm to deal with the sequential data in recommender systems.
- GAN [13]: It is a standard algorithm for fitting user interests through adversarial training.
- GRU [10]: It adjusts the fitting trend of the system to user interests through the gating mechanism. It is a widely used method for processing sequential data and a derivative algorithm of RNN.
- SimRec [22]: It is the first algorithm that combines GAN ideas with GRU methods, and it has achieved a certain breakthrough in recommendation results.
- SeqGAN [23]: It combines the idea of reinforcement learning with GAN for the first time, stably returning the gradient during the training process.
- IRGAN [24] and CFGAN [14]: These are explained in detail in Section 2. They mainly obtain high-quality recommendation results by transforming discrete data into easy-to-train data.
- SR-GNN [25]: It provides more accurate recommendations by analyzing and training directed graphs generated from users' behavior sessions.

We choose the classic baseline algorithm as a comparison to better represent the lifting point and amplitude of the method of this paper. At the same time, the state-of-the-art methods of GAN correlation algorithm and session-based recommendation are selected to reveal the advantages and disadvantages between DCFGAN and each method, which can prove the effectiveness of the proposed method.

4.3 Experimental results and analysis

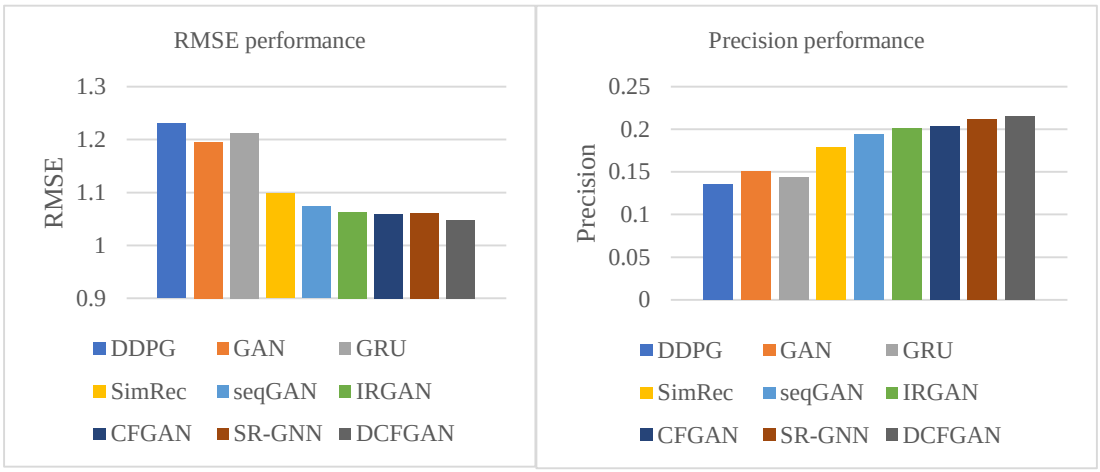

Figure 5. Three basic algorithms and four fusion methods compared with the DCFGAN method in this paper

Table 3. Comparison of public algorithms and methods in this paper

	DDPG	GAN	GRU	SimRec	seqGAN	IRGAN	CFGAN	SR-GNN	DCFGAN
RMSE	1.2305	1.1955	1.2124	1.0976	1.0732	1.0631	1.0589	1.0611	**1.0477**
Precision	0.1359	0.1505	0.1439	0.1783	0.1944	0.2010	0.2033	0.2114	**0.2149**

(1) We conducted unified experiments to further understand the performance of the proposed method compared with the baseline algorithm and the state-of-the-art recommendation algorithms. On the premise of ensuring the same dataset and testing method, it could be seen that our method has achieved generally good performance.

In terms of baseline algorithms, GAN, GRU and DDPG all performed poorly. DDPG, which represents reinforcement learning, does not acquire the data of generator network and thus performed relatively poorly in this scenario. The other two baseline approaches were slightly more effective, but the single structures and their respective shortcomings limited their performance.

In the derived RNN, by virtue of its combination with the adversarial network, SimRec improved the accuracy performance by 23% compared with the single GRU. The ability of generative adversarial network to supplement data and learn user preferences in this experimental environment was well reflected. The graph neural network of SR-GNN is much more complex than GRU and requires a large amount of data. In the current experimental environment, it still achieved good performance, and its accuracy rate was improved by 18% compared with SimRec.

The GAN correlation algorithm, regardless of the original GAN that is not closely related to reinforcement learning, seqGAN, IRGAN, CFGAN, and the proposed DCFGAN all achieved good results, which are generally better than those of the previous baseline and RNN correlation methods. These algorithms can find the problems of previous algorithms and improve them. After using the collaborative filtering idea commonly applied in recommender systems, CFGAN achieved a high accuracy rate. The proposed method combines several methods and incorporates a new negative sampling method, thus achieved the best results in this experiment. It could be seen from the above experimental comparison that the DCFGAN proposed in this paper improved the RMSE by at most 0.17 and the accuracy by at most 0.07, which constitutes a huge improvement to the basic algorithm. Compared with the four derivative algorithms of GAN, DCFGAN still showed an increase of more

than 6%.

(2) To answer **RQ2**, some of the parameter tuning experiments of GAN-related methods are shown below.

Table 4. Performance comparison between several algorithms under different discount rates γ

		seqGAN	IRGAN	CFGAN	DCFGAN
$\gamma = 0.9$	RMSE	1.0739	1.0636	1.0595	**1.0481**
	Precision	0.1941	0.2008	0.2027	**0.2147**
$\gamma = 0.8$	RMSE	1.0741	1.0635	1.0592	**1.0479**
	Precision	0.1941	0.2005	0.2029	**0.2148**
$\gamma = 0.7$	RMSE	1.0732	1.0631	1.0589	**1.0477**
	Precision	0.1944	0.2010	0.2033	**0.2149**

Taking several experimental algorithms using reinforcement learning related methods, the experimental results were observed and compared by setting different reinforcement learning discount rates. As can be seen from the above table, under the setting of several discount rates, the DCFGAN method in this paper achieved better results compared to the previous algorithms, and the best experimental results were obtained when γ=0.7.

(3) Based on the popularity of graph neural network in recent years, SR-GNN builds the user behavior data sequence into a graph neural network, and thus it has achieved great success in the session-based recommender system. In order to answer **RQ3** and study the advantages and disadvantages of the proposed method and SR-GNN in detail, we conducted the below additional experiments.

Table 5. Performance comparison of algorithms with different numbers of sessions

Sessions		50,000	100,000	150,000	200,000	250,000	300,000	350,000	400,000
RMSE	DCFGAN	1.0515	1.0477	1.0452	1.0439	1.0433	1.0429	1.0430	1.0425
	SR-GNN	1.0933	1.0611	1.0529	1.0448	1.0390	1.0321	1.0219	1.0175
Precision	DCFGAN	0.1981	0.2149	0.2206	0.2241	0.2253	0.2259	0.2261	0.2265
	SR-GNN	0.1736	0.2114	0.2213	0.2359	0.2436	0.2581	0.2632	0.2697

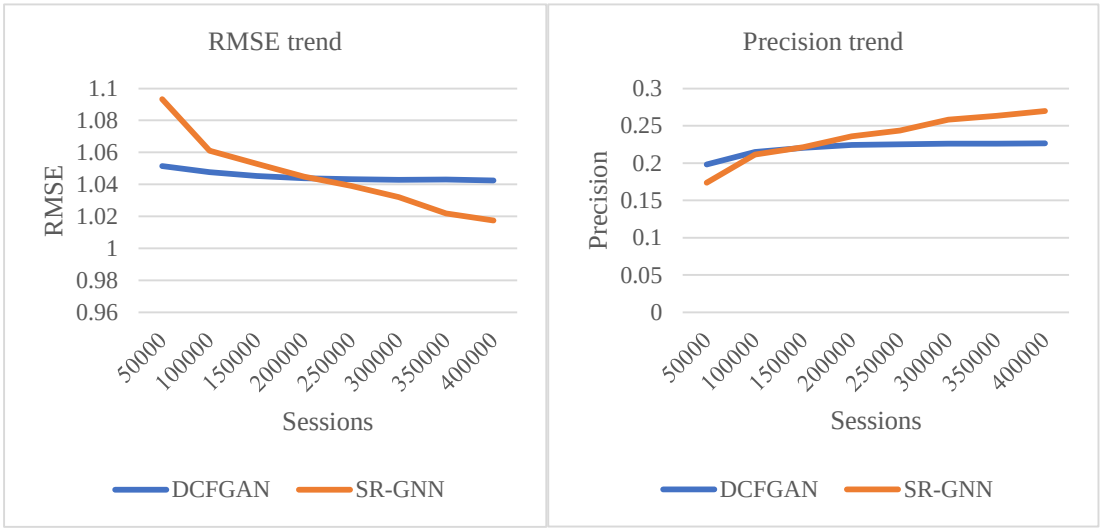

Figure 6. Comparison of SR-GNN and DCFGAN with different numbers of sessions

It is clearly reflected in the figure that the results obtained at 100,000 sessions gradually reverse as the number of sessions increases. The recommendation accuracy of DCFGAN and SR-GNN is very close when the number of sessions is 150,000-200,000, and the SR-GNN achieves better recommendation results when the number of sessions increases. When the number of sessions is low, the DCFGAN speeds up the convergence speed by relying on the generated data. Meanwhile, the SR-GNN is unable to construct the graph network effectively due to the small amount of data, thus obtaining a lower accuracy. With the growth of datasets, the construction of graph network becomes more and more mature and accurate. At this time, compared with DCFGAN, which processes serialized data, the graph network can learn the relationship between neighbor sessions, and SR-GNN shows higher accuracy.

(4) To answer **RQ4**, we decomposed and combined each method for the experiment.

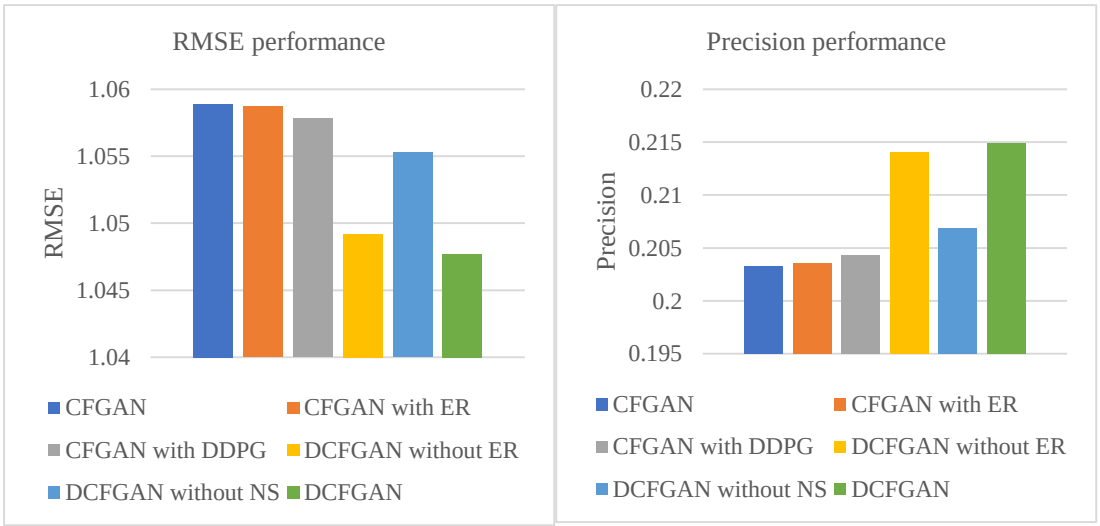

Figure 7. Comparison of CFGAN and DCFGAN after adding several improvement ideas proposed in this paper

Table 6. Comparison of the experimental results after the addition of several improved ideas of this paper

	CFGAN	CFGAN+ER	CFGAN+DDPG	DCFGAN-ER	DCFGAN-NS	DCFGAN
RMSE	1.0589	1.0587	1.0578	1.0492	1.0553	**1.0477**
Precision	0.2033	0.2036	0.2043	0.2141	0.2069	**0.2149**

This part is a summary of the application of various methods in this paper. Here, 'CFGAN+ER' is the CFGAN algorithm using experience replay, 'CFGAN+DDPG' only replaces the PG part of CFGAN with the model composed of DDPG, 'DCFGAN-ER' only removes the DCFGAN algorithm for experience replay, and 'DCFGAN-NS' is the DCFGAN algorithm with only negative sampling removed. The results show that the effect of using experience replay in the experiment with improved CFGAN is not obvious, which reduces the RMSE value by about 0.004. Meanwhile, the experiment using only the improved negative sampling method reduces the RMSE value by 0.008. Therefore, it can be concluded that the three main innovations of this work all have a certain improvement effect on the results, with the negative sampling method exhibiting the most obvious improvement effect.

5. Conclusions and future work

This paper proposes a Generative Adversarial Networks-based Collaborative Filtering method combined with Deep Reinforcement Learning (DCFGAN) to improve the original CFGAN method. The DCFGAN solves the problem of gradient transfer instability in the training process of the model by using DDPG to replace the original PG algorithm. At the same time, DCFGAN includes a new negative sampling method, which uses pre-training and dynamic sample updates during training to make the sampling results be more in line with the real preferences of users. Moreover, experience replay is used to improve the traditional generative network G, thereby improving the training dimension and accuracy, and avoiding overfitting. Experiments are conducted to prove that our method can grasp user preferences more quickly and more accurately compared with previous algorithms in session-based recommender systems when the data are small, so as to alleviate the cold start problem of the recommender system to a certain extent. In these methods, the improved negative sampling method plays a key role in the process of improving prediction accuracy.

Our method is effective in solving the cold start problem, but it is still weaker than some methods that can use user attribute information to predict user preferences when the training begins. At the same time, the accuracy of our method for user preference learning in scenarios with large data volume is poorer than that of the graph neural network model, which has become popular in recent years. These are situations where our method is weak and worthy of enhancement.

In future work, we aim to use matrix decomposition and other methods that are more effective in the recommender system to calculate user preferences in advance, so as to obtain an increase in terms of accuracy. We also plan to combine the graph neural network to achieve a more accurate result for user preferences including a large amount of data.

Acknowledgments

This paper is supported by Natural Science Foundation of Shandong Province (No.ZR2021MF104, ZR2021MF113), National Natural Science Foundation (No. 62072288), Key R&D Projects of Qingdao Science and Technology Plan (No.21-1-2-19-xx), Qingdao West Coast New District Science and Technology Plan (No.2020-1-6).

References

[1] K. Patel, H. B. Patel, A state-of-the-art survey on recommendation system and prospective extensions, Computers and Electronics in Agriculture 178 (2020) 105779.

[2] S. Wang, L. Cao, Y. Wang, et al., A survey on session-based recommender systems, ACM Computing Surveys (CSUR) 54 (7) (2021) 1–38.

[3] P. Brémaud, Markov chains: Gibbs fields, Monte Carlo simulation, and queues, Vol. 31, Springer Science & Business Media, 2013.

[4] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939, 2015.

[5] Z. Wang, W. Wei, G. Cong, et al., Global Context Enhanced Graph Neural Networks for Session-Based Recommendation, Association for Computing Machinery, New York, NY, USA, 2020, pp.

169–178.
[6] L. P. Kaelbling, M. L. Littman, A. W. Moore, Reinforcement learning: A survey, Journal of artificial intelligence research 4 (1996) 237–285.
[7] M. R. Samsami, H. Alimadad, Distributed Deep Reinforcement Learning: An Overview, arXiv preprint arXiv:2011.11012, 2020.
[8] M. Sabatelli, G. Louppe, P. Geurts, et al., The deep quality-value family of deep reinforcement learning algorithms, in: 2020 International Joint Conference on Neural Networks (IJCNN), 2020, pp. 1-8.
[9] I. Munemasa, Y. Tomomatsu, K. Hayashi, et al., Deep reinforcement learning for recommender systems, in: 2018 International Conference on Information and Communications Technology (ICOIACT), 2018, pp. 226–233.
[10] K. Cho, B. van Merrienboer, Ç. Gülçehre,, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, Computer ence, 2014.
[11] D. Silver, G. Lever, N. Heess, et al., Deterministic policy gradient algorithms, in: Proceedings of the 31th International Conference on Machine Learning, JMLR, 2014, pp. 387–395.
[12] F. Liu, R. Tang, X. Li, et al., State representation modeling for deep reinforcement learning based recommendation, Knowl. Based Syst. 205 (2020) 106170.
[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, et al., Generative adversarial nets, in: Advances in neural information processing systems. 2014: 2672-2680.
[14] D.-K. Chae, J.-S. Kang, S.-W. Kim, J.-T. Lee, Cfgan: A generic collaborative filtering framework based on generative adversarial networks, in: Proceedings of the 27th ACM international conference on information and knowledge management, 2018, pp. 137–146.
[15] M. Naeem, S. T. H. Rizvi, A. Coronato, A gentle introduction to reinforcement learning and its application in different fields, IEEE Access 8 (2020) 209320–209344.
[16] P. S. Thomas, E. Brunskill, Policy gradient methods for reinforcement learning with function approximation and action-dependent baselines, arXiv preprint arXiv:1706.06643, 2017.
[17] Mnih, K. Kavukcuoglu, D. Silver, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv:1312.5602, 2013.
[18] Y Li, Deep reinforcement learning: An overview, arXiv preprint arXiv:1701.07274, 2017.
[19] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning, Nat. 518 (7540) (2015) 529–533.
[20] R. Lopez, P. Boyeau, N. Yosef, M. I. Jordan, J. Regier, Decision-making with auto-encoding variational bayes, arXiv preprint arXiv:2002.07217, 2020.
[21] A. Antoniou, A. Storkey, H. Edwards, Data augmentation generative adversarial networks. arXiv, 2017.
[22] X. Zhao, L. Xia, L. Zou,, et al. Toward simulating environments in reinforcement learning based recommendations, arXiv preprint arXiv:1906.11462, 2019
[23] Yu L, Zhang W, Wang J, et al. Seqgan: Sequence generative adversarial nets with policy gradient[C]//Thirty-First AAAI Conference on Artificial Intelligence. 2017.
[24] L. Yu, W. Zhang, J. Wang, Y. Yu, Seqgan: Sequence generative adversarial nets with policy gradient, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 31, 2017.
[25] S. Wu, Y. Tang, Y. Zhu, et al., Session-based recommendation with graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 346–353.
[26] . Zheng, S. Liu, Z. Li, S. Wu, Dgtn: Dual-channel graph transition network for session-based

recommendation, in: 2020 International Conference on Data Mining Workshops (ICDMW), IEEE, 2020, pp. 236–242.
[27] X. Zhao, L. Zhang, L. Xia, et al., Deep Reinforcement Learning for List-wise Recommendations, arXiv preprint arXiv:1801.00209, 2017.
[28] D Silver, G Lever, N Heess, et al. Deterministic policy gradient algorithms, In: International conference on machine learning, PMLR, 2014, pp. 387-395.
[29] V. Konda, Actor-critic algorithms, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA (2002).
[30] C. J. C. H. Watkins, P. Dayan, Technical note q-learning, Mach. Learn. 8 (1992) 279–292.
[31] T. P. Lillicrap, J. J. Hunt, A Pritzel, et al. Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971, 2015.
[32] Q. Wang, H. Yin, H. Wang, et al., Enhancing collaborative filtering with generative augmentation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 548–556
[33] T. K. Dang, Q. P. Nguyen, V. S. Nguyen, A study of deep learning-based approaches for session-based recommendation systems, SN Comput. Sci. 1 (4) (2020) 216.
[34] L. Zou, L. Xia, P. Du, et al., Pseudo dyna-q: A reinforcement learning framework for interactive recommendation, in: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, ACM, 2020, pp. 816–824.
[35] Y. Lei, H. Pei, H. Yan, et al., Reinforcement learning based recommendation with graph convolutional q-network, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1757–1760.
[36] S. Latifi, N. Mauro, D. Jannach, Session-aware recommendation: A surprising quest for the state-of-the-art, Inf. Sci. 573 (2021) 291–315.
[37] G. Zheng, F. Zhang, Z. Zheng, et al., Drn: A deep reinforcement learning framework for news recommendation, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 167–176.
[38] B. Hidasi, A. Karatzoglou, Recurrent neural networks with top-k gains for session-based recommendations, in: Proceedings of the 27th ACM international conference on information and knowledge management, 2018, pp. 843–852.
[39] T. M. Phuong, T. C. Thanh, N. X. Bach, Combining user-based and session-based recommendations with recurrent neural networks, in: International Conference on Neural Information Processing, Springer, 2018, pp. 487–498.
[40] G. Zaks, G. Katz, Recom: A deep reinforcement learning approach for semi-supervised tabular data labeling, Information Sciences 589 (2022) 321–340.
[41] Y. Zhao, B. Chen, X. Wang, Z. Zhu, Y. Wang, G. Cheng, R. Wang, R. Wang, M. He, Y. Liu, A deep reinforcement learning based searching method for source localization, Information Sciences 588 (2022) 67–81.
[42] M. Gao, J. Zhang, J. Yu, J. Li, J. Wen, Q. Xiong, Recommender systems based on generative adversarial networks: A problem-driven perspective, Information Sciences 546 (2021) 1166–1185.
[43] J. Liao, W. Zhou, F. Luo, J. Wen, M. Gao, X. Li, J. Zeng, Sociallgn: Light graph convolution network for social recommendation, Information Sciences 589 (2022) 595–607.
[44] W. Li, Y. M. Tang, K. M. Yu, S. To, Slc-gan: An automated myocardial infarction detection model based on generative adversarial networks and convolutional neural networks with single-lead electrocardiogram synthesis, Information Sciences 589 (2022) 738–750.

[45] H. Liu, L. Guo, P. Li, P. Zhao, X. Wu, Collaborative filtering with a deep adversarial and attention network for cross-domain recommendation, Information Sciences 565 (2021) 370–389.