applied sciences

MDPI

Article

DDRCN: Deep Deterministic Policy Gradient Recommendation Framework Fused with Deep Cross Networks

Tianhan Gao [1,†], Shen Gao [1,*,†], Jun Xu [2] and Qihui Zhao [1]

[1] Software College, Northeastern University, Shenyang 110169, China
[2] Science and Technology on Special System Simulation Laboratory, Beijing Simulation Center, Beijing 100854, China
* Correspondence: 2071276@stu.neu.edu.cn
† These authors contributed equally to this work.

Abstract: As an essential branch of artificial intelligence, recommendation systems have gradually penetrated people's daily lives. It is the active recommendation of goods or services of potential interest to users based on their preferences. Many recommendation methods have been proposed in both industry and academia. However, there are some limitations of previous recommendation methods: (1) Most of them do not consider the cross-correlation between data. (2) Many treat the recommendation process as a one-time act and do not consider the continuity of the recommendation system. To overcome these limitations, we propose a recommendation framework based on deep reinforcement learning techniques, known as DDRCN: a deep deterministic policy gradient recommendation framework incorporating deep cross networks. We use a Deep network and a Cross network to fit the cross relationships between the data, to obtain a representation of the user interaction data. The Actor-Critic network is designed to simulate the continuous interaction behavior of users through a greedy strategy. A deep deterministic policy gradient network is also used to train the recommendation model. Finally, we conduct experiments with two publicly available datasets and find that our proposed recommendation framework outperforms the baseline approach in the recall and ranking phases of recommendations.

Keywords: recommendation system; deep deterministic policy gradient; deep cross network; reinforcement learning

check for updates

Citation: Gao, T.; Gao, S.; Xu, J.; Zhao, Q. DDRCN: Deep Deterministic Policy Gradient Recommendation Framework Fused with Deep Cross Networks. *Appl. Sci.* **2023**, *13*, 2555. https://doi.org/10.3390/app13042555

Academic Editor: Giancarlo Mauri

Received: 14 December 2022
Revised: 11 February 2023
Accepted: 15 February 2023
Published: 16 February 2023

1. Introduction

With the increasing convenience of information access through the Internet, a large amount of data are generated when obtaining videos, commodities, news, music, etc. For example, the transaction volume of Tmall's Double 11 in 2022 is CNY 557.1 billion, and the explosion of data causes the problem of information overload [1]. The recommendation system simulates the user's consumption preferences based on the user's behavior preference. It also predicts the items that users may be interested in, and provides personalized recommendation services. At the same time, it can also bring commercial value to enterprises. For example, 80% of Netflix movies come from the recommendation system [2], so more researchers and multimedia service providers pay attention to the recommendation system. The current recommendation systems can be divided into three categories: traditional model-based recommendation systems, deep learning-based recommendation systems, and deep reinforcement learning-based recommendation systems. However, the following limitations still exist.

Firstly, the quality of recommendation results mainly depends on the data between users and the system, while most recommendation systems do not fully exploit the cross relationship between data. The recommendation framework captures user preferences based on user data, item data, user–items interaction data, and statistical data. Each data point does not exist independently. Early machine learning (ML) practitioners still sought to improve the expressiveness of their frameworks by manually identifying feature crossings [3]. Recently, Wang et al. proposed a DCN [4] and improved the DCN-V2 [5] framework. The framework can effectively learn the explicit and implicit intersection of features and conduct ranking experiments in Google's system with high accuracy.

Second, most recommendations are assumed to be static without considering the impact of long-term returns. However, in fact, the recommendation system strongly depends on users' continuous decision-making behaviors. While short-term returns are essential, a failure to consider the long-term returns of a recommendation may lead to recommendation bias behavior. Xie et al. proposed a meta-learning framework (LSTTM) for online recommendations. The framework was deployed on the WeChat Top Stories, with remarkable results [6]. Shivaram et al. proposed an attention recommendation framework, which mainly increases the attention to specific topic words to avoid excessive attention to general hot terms and reduce the homogenization effect in the recommendation system [7]. Specifically, the recommendation system recommends a news article to the user, and the user has a series of actions, such as likes and favorites for this news. These actions indicate that the user may be interested in this topic. Still, the user may not like the long-term recommendation of related news since the user's long-term preferences are not considered. For example, the timeliness of news is very strong, and recommending early news to users may not be very attractive.

Reinforcement learning was first proposed to solve optimal control problems, an integral part of artificial intelligence. Reinforcement learning aims to maximize the goal, reward, and expectation through continuous trial and error between the agent and the environment. In the actual recommendation application, the user's behavior characteristics constantly change during interaction with the system. Only by dynamically adjusting the recommendation action according to the real-time behavior attributes of the user can the long-term maximum revenue be guaranteed, which is consistent with the features of the reinforcement learning algorithm. When reinforcement learning is adopted to perform a recommendation framework, it mainly includes value-based and policy-based methods. The dynamic recommendation process based on reinforcement learning is shown in Figure 1, which includes three parts: the user, the agent (recommendation system), and the item list. Users first view the recommended items in the list and then give feedback according to their preferences, including clicks, favorites, forwarding, etc. In particular, the items operated by users are included in the order. According to the user's feedback, the agent constantly learns to adjust the recommendation actions, and predicts the items that the user may be interested in, to make recommendations.

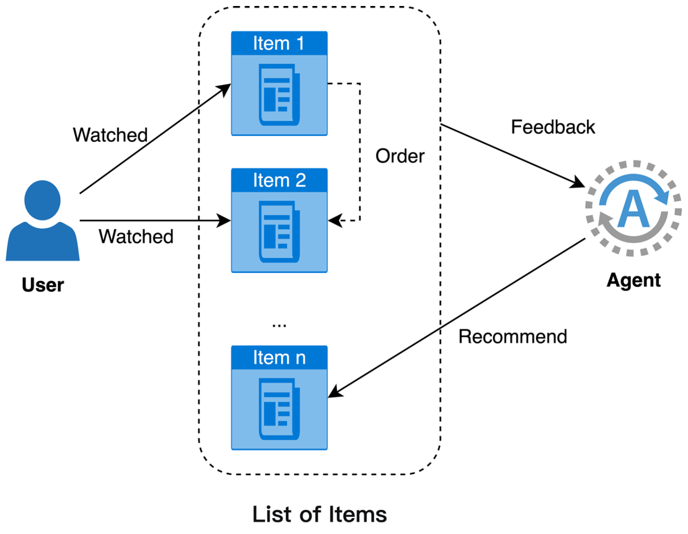

Figure 1. Dynamic recommendation process for reinforcement learning.

In our study, we provide a deep deterministic policy gradient-based reinforcement learning recommendation method called DDRCN, which includes Actor and Critic dual networks for recommendation action generation and action value evaluation, respectively. In particular, we use a deep cross network to process the basic features of the user data, and to fit a state representation for strategy selection and value estimation. During the action selection, the Actor policy neural network may fit the policy function through the deep cross network, which directly outputs the recommended actions saved to the project recommendation pool. The Critic network is a value estimation of the current action and state of the user, and the model is trained by two networks together. Through a large number of experiments, the proposed recommendation framework achieves good recommendation results. The contributions of this paper are as follows:

- We provide a deep deterministic policy gradient recommendation framework, DDRCN, that fuses deep cross networks. The framework uses the Actor-Critic approach, which maximizes the cumulative reward of recommendations through the continuous exploration and exploitation of the Actor and Critic networks, combined with greedy strategies.
- In this recommendation framework, we fit the data features between users and items, utilizing a deep cross network. The deep cross network consists of a Deep network and a Cross network, and the two networks work together to compute the cross relationship between the features.
- We conducted relevant recommendation experiments on the real movie and music datasets, and the experimental results show that our proposed model outperforms its competitors regarding recall and ranking effects.

The rest of the sections are organized as follows. Section 2 presents the related work. Section 3 introduces the preliminary knowledge. Section 4 elaborates on the proposed recommendation framework. Section 5 discusses the experimental results, and the paper is concluded in Section 6.

2. Related Work

Recommendation systems have been developed for more than 30 years. From filtering emails in the beginning to personalized recommendations today, recommendation systems have undoubtedly penetrated everyone's life. The recommendation technology develops from the traditional recommendation framework to deep learning and reinforcement learning recommendation frameworks.

2.1. Traditional Recommendation Methods

Collaborative filtering [8] is the earliest recommendation framework that has the most significant impact on the industry. Collaborative filtering approaches assume that users with similar preferences are interested in the same items. However, the data available for recommendation are sparse in the actual scenario, so the generalization ability of the collaborative filtering algorithm is poor. Matrix factorization [9] is also a collaborative filtering method. It is a decomposition and inner product of the data table to replace the missing values. However, matrix factorization only considers user behavior characteristics and it has poor interpretability. Logistic regression [10] regards recommendation as a binary classification problem and is often used to predict click-through rates. It is based on hypothesis testing, maximum likelihood estimation, and gradient descent to solve parameters, where the output result is between 0 and 1. Then, the user is judged to click or not to click, using the threshold, which has certain interpretability. The factorization machine [11] can be understood as a variant of matrix factorization. It reduces the complexity by learning the hidden weight feature vector and the feature vector cross-internal product. The factorization machine method reduces the time complexity using the pairwise cross inner product of feature vectors, and can also be generalized to all features. In addition, some combination frameworks, such as Facebook's automatic selection feature combination method, generate discrete random features and feed them into a logistic regression framework to estimate the CTR. However, the traditional recommendation method is too simple and can result in most of the recommended content being very similar. It can lead to user boredom.

2.2. Recommendation Methods with Deep Learning

With more powerful computing power and fitting ability, deep learning has developed rapidly in recent years, so more researchers apply deep learning to recommendation systems. AutoRec [12] is the first CF recommendation framework that introduces neural networks, autoencoders, to predict users' ratings for recommended items. Since the AutoRec framework leads to the pain of insufficient expression ability, the Deep Crossing [4] framework proposes a four-layer network structure, which perfectly solves the sparsity problem of feature vectors. The Wide & Deep [13] model guarantees the generalization ability and memory of the recommendations through a dual network. In the recommendation scenario, users' interests are changeable, so the DIN [14] framework introduces the attention mechanism and learns the attention weight through the framework results to consider the interest preferences of different users. Nowadays, the scale of the recommendation system is getting larger and larger, which brings low recommendation efficiency. HS-GCN [15] framework proposes a Hamming space graph convolutional network to model the ladder similarity in Hamming space, and recommends it through the user-item bipartite chart. In addition, the recommended data sources are becoming more and more abundant. The UMPR [16] framework is a multimodal recommendation method based on deep users, which makes recommendations by constructing a matching between the visual preference embeddings of users and the visual embeddings of items. Although deep learning-based recommendation systems have achieved good results, they also have fatal drawbacks. Deep learning relies entirely on datasets, and it predicts recommended items by learning the relationships between data. Deep learning recommendation methods assume that the recommendation is a past behavior that is incompatible with the user's behavioral changes.

2.3. Recommendation Methods with Reinforcement Learning

Traditional and deep learning-based recommendation frameworks rely on historical data to capture user preferences, but the actual recommendation is a continuous and dynamic decision-making process. Reinforcement learning simulates the ongoing "exploration-exploitation" process between the agent (recommendation system) and the environment (user), and finds a balance to obtain the recommendation process with the maximum cumulative benefit. Common reinforcement learning recommendation methods include policy-based methods and value function-based methods.

Value-based methods in reinforcement learning decide the recommended action by selecting the maximum Q value of the state. GoalRec [17] is a value-based deep reinforcement learning framework that addresses high-variability environments and uncertain reward Settings. FeedRec [18] is a recommendation framework that optimizes user engagement through reinforcement learning. The framework includes a Q-Network and an S-Network, and iteratively trains the two networks to optimize the indicators of user engagement. Policy-based recommendation methods find the optimal policy based on the current state, and outputs the probability distribution of each action. PDQ [19] is a recommendation method based on offline policy learning. The framework executes policy functions offline and introduces a simulation environment to help with policy improvement. OPS2 [20] is a two-stage off-policy gradient recommendation method, including candidate generation and ranking stages.

The recommendation method based on the value function is unsuitable for handling continuous actions. In contrast, the recommendation method based on policy function cannot evaluate the goodness of the policy. The Actor-Critic network combines the advantages of value function and policy function methods. The dual network's prediction and value estimation of recommendation actions can deal with continuous recommendations and user behaviors. Based on this, we propose a deep deterministic policy gradient recommendation method based on the Actor-Critic framework. The technique can handle large-scale continuous behaviors, is more suitable for recommendation scenarios, and has a faster model convergence rate.

3. Preliminaries

Reinforcement learning includes the environment and agent, and the goal of maximizing the cumulative reward is achieved through the continuous interaction between them, which can be modeled as a Markov Decision Process (MDP), abstracted as a five-tuple $< S, A, R, P, \gamma >$. S denotes the set of states, A denotes the set of actions, R is the reward function, P is the transition probability, and γ is the discount coefficient. Therefore, the recommendation system is abstracted as an agent, the user is abstracted as the environment, the recommendation system performing a recommendation is abstracted as an action, and the user's behavior characteristics are abstracted as states. In each recommendation process, the recommendation system dynamically updates the action based on the user's status and feedback to ensure the maximum cumulative profit of the recommendation. The specific MDP framework of the recommendation process is as follows:

- State space S: S is the set of environment states, and $s_t \in S$ represents the state of the agent at time t, which is the interaction between the user and the recommendation system at time t.
- Action space A: A is the set of actions that the agent can take, and $a_t \in A$ represents the action taken by the agent at time t. In particular, actions here refer to action vectors.
- Reward R: The recommendation system will recommend actions based on the user's state and behavior, and the user will provide feedback (click, rating, retweet, etc.). Recommendation systems receive instant rewards based on user feedback $r(s, a)$.
- State transition P: When the recommendation system recommends action at time step t, the state of the user at this time is transferred from s_t to s_{t+1}.
- Discount factor γ: $\gamma \in [0, 1]$is a discount factor used to indicate the importance of future rewards, with γ being close to 1 to consider long-term rewards, and γ being close to 0 to consider immediate rewards.

Figure 2 illustrates the interaction process between the agent and the environment. The agent gives an action a_t according to the current state s_t. After receiving the action of the agent, the environment converts the state from s_t to s_{t+1} and rewards the agent r_t for its behavior. The agent receives the reward r_t and the state s_{t+1}, and takes the next action a_{t+1}, and so on. In particular, the agent's action does not represent the recommended item or the recommendation sequence, but a continuous parameter vector. This vector is then the inner product with the item embedding to obtain the item's rating, and the specific item is recommended to the user according to the order of the rating. In this paper, we construct a recommendation framework through the Actor-Critic network. In the Actor-Critic network, the Actor-network is a policy network, and the Critic-network is a state value estimation network. Through the dual networks, the model acts toward high cumulative rewards. In contrast, Q-learning is performed by storing Q-values (values of state action pairs) in Q-tables and continuously going through the Q-tables to update them. This approach is not suitable for handling large-scale scenarios. SA2C [21] is a variant based on the Actor-Critic network for recommendation scenarios. It introduces supervised learning to simulate the Actor-network to generate correct actions. In this paper, the Actor-Critic network is trained by a deep deterministic policy gradient network. The model is converged by setting the Actor Target network and the Critic Target network.

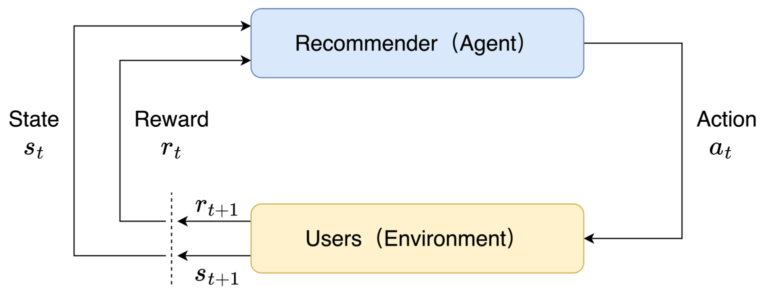

Figure 2. MDP decision process.

4. The Proposed Framework

4.1. The Architecture

As we mentioned in Section 1, traditional and deep learning recommendation methods neither treat user decision behavior as being static, nor do they consider immediate rewards. To address the shortcomings, we propose a deep deterministic policy gradient recommendation method incorporating a deep cross network, which mainly includes two parts: Actor policy network and Critic value network. In particular, there is a state representation part in the Actor-network, as shown in Figure 3.

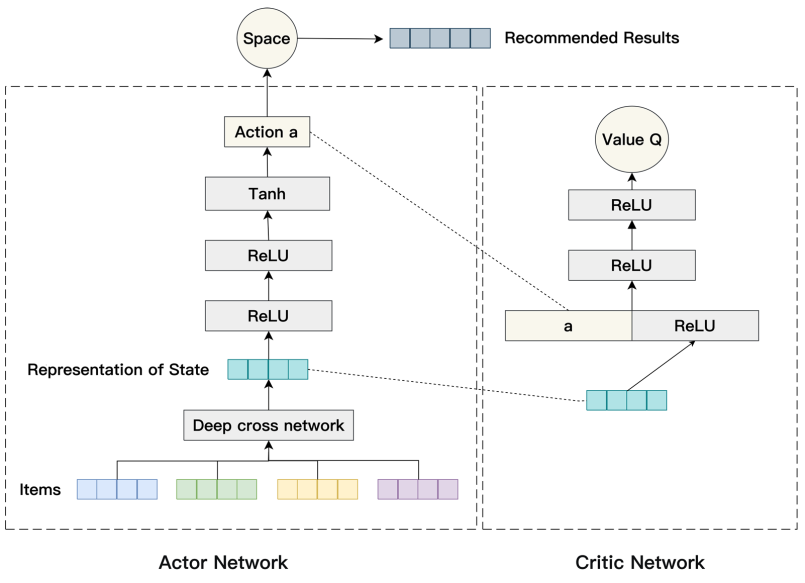

Figure 3. The architecture of the framework.

4.1.1. Actor Policy Neural Networks

The Actor network outputs actions based on user state features, as shown in the left part of the architecture. The user behavior vector includes user features, item features, statistical features, and scene features. These vectors are passed through the state representation module (deep cross network) to obtain the user state representation vector. The user is a generalized representation of the user's preferences, and the state at the moment t is defined as follows:

$$s_t = f(H_t), \tag{1}$$

where $f(x)$ is the state representation function, H_t represents the vector embedding of the history where the user has interacted with the recommendation system, $H_t = \{i_1, i_2, \ldots, i_n\}$. When the agent recommends an item according to the policy, if the user makes positive feedback on the recommended action, then s_{t+1} is updated to $f(H_{t+1})$. Otherwise, s_{t+1} is still equal to $f(H_t)$. The user feature representation vector is input, and the recommended action is output after a three-layer activation function. The action under state s is defined as follows:

$$a = \pi_\theta(s), \tag{2}$$

where a represents the continuous action vector, which is the output of the Actor network. π_θ represents the selection strategy; here we use ε–greedy. Specifically, the candidate items are calculated as follows [22]: //

$$score_t = i_t a^T, \tag{3}$$

Finally, the recommendation results are obtained by ranking each item according to its score.

4.1.2. Critic Value Neural Networks

The Critic network is a deep Q network, as shown on the right of the model architecture figure. It is used to estimate the quality of the state and action, namely, the estimate function of the Q value. The value function $Q_\pi(s,a)$ is estimated through neural network fitting out $Q_w(s,a)$. The Q-value is a scalar that allows the model to be updated and optimized to enhance the action. We update the Actor network based on a deterministic policy gradient approach [23], formulated as follows:

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_t \nabla_a Q_w(s,a)|_{s=s_t,a=\pi_\theta(s_t)} \nabla_\theta \pi_\theta(s)|_{s=s_t}, \tag{4}$$

where $J(\pi_\theta)$ denotes the expectation of the Q value. N is the size of the batch. The network of Critic is updated using the temporal difference learning method [24], as follows:

$$y_i = r_i + \gamma Q_{w'}(s_{i+1}, \pi_{\theta'}(s_{i+1})), \tag{5}$$

$$L = \frac{1}{N}\sum_i (y_i - Q_w(s_i, a_i))^2, \tag{6}$$

where y_i represents the time difference target value, γ represents the discount rate, L represents the mean square error, $w^{'}$ represents the parameters of the Critic Target network, and $\theta^{'}$ represents the parameters of the Actor Target network.

4.1.3. State Representation Module

The state representation module represents user characteristics and is the input for each of the two neural networks to make predictions. The BINN framework [25] provides an item embedding method for user interaction, and it is adopted for subsequent recommendation tasks. We take the deep cross network to mine the cross relationship between the data of the recommendation system to obtain the user state representation vector. The advantages include fully mining the cross relationship between the features, preventing the gradient from disappearing, and being memory and computation friendly, as shown in Figure 4.

The input of the state representation module is the concatenation of the user feature vector, the item feature vector, the statistical feature vector, and the scene feature vector in the recommendation data. DCN includes Deep Network and Cross Network, where Deep Network is a fully connected network defined as follows [5]:

$$H_i = W x_i + b, \tag{7}$$

where x_i represents the data feature embedding vector, and W and b are parameter matrices, which are then passed to the next layer of the network through the Relu activation function. The Cross Network excavates the features between data by disintegrating vectors into subspaces and performing feature cross. The feature vector is defined as follows [5]:

$$x_{i+1} = x_0 \odot (W x_i + b) + x_i, \tag{8}$$

where x_0 represents the splicing of initial data feature embedding vectors. The Deep Network and Cross Network output is stacked, and finally, the user state representation vector is output through a single linear mapping.

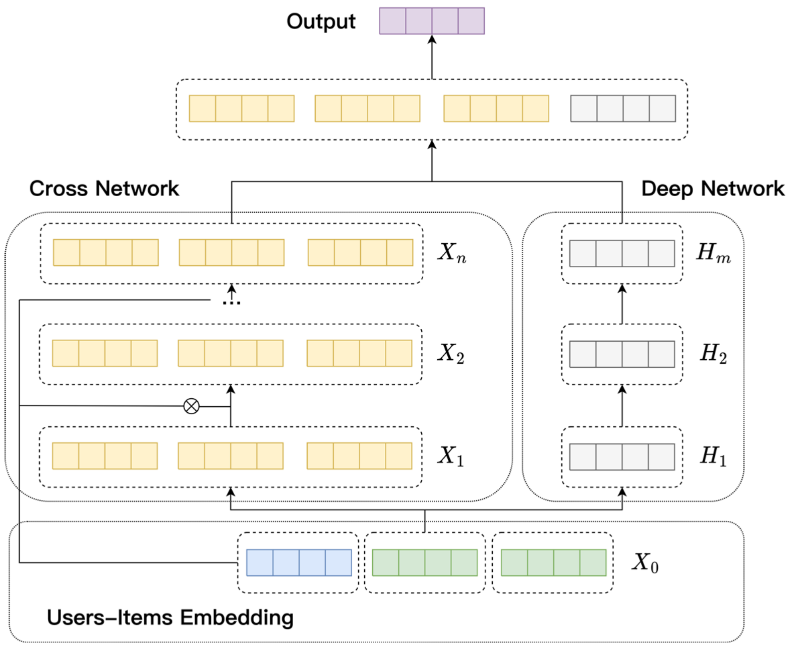

Figure 4. State representation module.

4.2. Training Process

We train the framework through the deep deterministic policy gradient algorithm. The training process mainly includes two stages: transfer (lines 4–12) and parameter update (lines 13–20), which are shown in Algorithm 1.

In the state transition generation phase, we initialize the Actor network and the Critic network parameters, and then we fit the user's state representation vector based on the deep cross network. Afterward, according to $\pi_\theta(s)$ and $\varepsilon - greedy$, the recommended action vector is calculated, and the inner product is completed with the embedding vector of the recommended item to obtain the score of the item. Afterward, the reward r_t is calculated based on the user's current state and action, and the quadruple (s_t, a_t, r_t, s_{t+1}) is saved in the experience pool.

During the parameters update stage, we first sample a small batch of four-tuples (s_i, a_i, r_i, s_{i+1}) from the experience replay pool, then calculate the objective function and loss function according to lines 14–15 in Algorithm 1. The parameters of the Actor network, Critic network, and target network are updated through the soft replace strategy.

4.3. Evaluation Process

As for reinforcement learning, the most direct way to evaluate the framework is to let users interact with the recommendation system, and to evaluate the quality of the framework through the real environment. However, there are many potential business risks and deployment costs in the online environment, so we use offline evaluation to complete this task.

The offline evaluation tests the policy's effect learned by the framework, as shown in Algorithm 2. The initial item and state representation are observed in the offline log. The recommendation agent calculates the recommended action vector a_t according to the current state and policy π_θ. Then, the item's score is calculated, and i_t is recommended according to the score. Finally, the current reward r_t is calculated according to the offline log, the user's state s_{t+1} is updated, and the recommended items are deleted from the candidate pool.

Algorithm 1: Training Process

```
  Input: Actor learning rate $\eta_a$, Critic learning rate $\eta_c$, Discount factor $\gamma$, Batch size $N$, state window size $n$, Reward function $R$, target network parameter $\tau$
  Output: Parameter $\theta$ for Actor network, parameter $\omega$ for Critic network
1 Randomly initialize parameters $\theta$ of Actor network $\pi_\theta$ and parameters $\omega$ of Critic network $Q_\omega$ Initializes parameters $\theta^{'}$ of Actor Target network $\pi^{'}$ and parameters $\omega^{'}$ of Critic Target network $Q^{'}$ Initialize the experience replay pool $D$ for $session = 1, M$ do
2     Observe that the initial state represents $s_0$ according to the state representation module for $t = 1, T$ do
3         Observe the current state $s_t = f(H_t)$, where $H_t = \{i_1, i_2, \dots, i_n\}$ Compute the action $a = \pi_\theta(s_t)$ according to $\varepsilon - greedy$ Recommended items $i_t$ are ranked according to item scores Calculate the reward $r_t = R(s_t, a_t)$ based on user feedback Observe the new state $s_{t+1} = f(H_{t+1})$ Save the state transition quadruple $(s_t, a_t, r_t, s_{t+1})$ to $D$ Sample a small batch of quad $(s_i, a_i, r_i, s_{i+1})$ from the experience replay pool Setting up $y_i = r_i + \gamma Q_{w'}(s_{i+1}, \pi_{\theta'}(s_{i+1}))$ Update the Critic network through the loss function $L = \frac{1}{N}\sum_i (y_i - Q_w(s_i, a_i))^2$ Update the Actor network by sampling the policy gradient function $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_t \nabla_a Q_w(s,a)|_{s=s_t, a=\pi_\theta(s_t)} \nabla_\theta \pi_\theta(s)|_{s=s_t}$ Update the target network parameters: $\theta^{'} \leftarrow \tau\theta + (1-\tau)\theta^{'}$ $\omega^{'} \leftarrow \tau\omega + (1-\tau)\omega^{'}$
4 return $\theta$ and $\omega$
```

Algorithm 2: Offline Evaluation Process

```
  Input: State window size $n$, and Reward function $R$
1 Observe the initial project and state set according to the offline log for $t = 1, T$ do
2     Observe that the initial state represents the $s_0$ according to the state representation module for $t = 1, T$ do
3         Observe the current state $s_t$ Calculate the action $a_t = \pi_\theta(s_t)$ according to the current policy Observe the recommended item $i_t$ according to the rating of the item Calculate the reward $r_t = R(s_t, a_t)$ according to the feedback in the log Update state $s_{t+1} = f(H_{t+1})$ Remove items from the candidate set
```

5. Experiment

5.1. Dataset and Evaluation Metrics

The MovieLens dataset contains multiple user ratings of various movies, movie metadata, and user attribute information. The Lastfm dataset is a public benchmark dataset released by HetRec in 2011, including users, artists, and their interactions with each other. The specific statistics of the dataset are in Table 1, and we use the real-world public MovieLens and Lastfm datasets for our experiments.

Table 1. Dataset Statistics.

Dataset	User	Item	Rating
MovieLens (100k)	943	1682	100,000
MovieLens (1M)	6040	3952	1,000,209
Lastfm	1892	17,632	92,834

- MovieLens (100k): A stable benchmark dataset of 100,000 ratings given to 1700 movies by 1000 users.

- MovieLens (1M) : A stable benchmark dataset consisting of 1 million ratings from 6000 users for 4000 movies.
- Lastfm dataset: a stable benchmark dataset of 100,000 listening histories of 17,000 songs by 2000 users.

We evaluated the MovieLens and Lastfm datasets offline, and used *Precision@k* and *NDCG@k* to assess the model's effectiveness. Precision focuses on prediction accuracy, while *NDCG* focuses on the relevance of recommendations. *Precision@k* indicates that the first *k* recommended items are relevant, and *Precision@k* is defined as follows:

$$Precision@k = \frac{Rel_u \cap Rec_u}{Rec_u}, \tag{9}$$

where Rel_u denotes the items relevant to user u (test set), and Rec_u represents the top k lists recommended to the user. NDCG is the metric in the recommendation that indicates predictive relevance, and *NDCG@k* is defined as follows:

$$DCG = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2 (i+1)}, \tag{10}$$

$$IDCG = \sum_{i=1}^{k} \frac{1}{\log_2 (i+1)}, \tag{11}$$

$$NDCG@k = \frac{DCG@k}{IDCG}, \tag{12}$$

where rel_i denotes the recommended relevance at position i. Relevance is denoted by 1, and irrelevance is denoted by 0.

5.2. Baseline Algorithms

As our baseline models, we select the BRP [26], NCF [27], and DRR [22] models, which are discussed below:

- BPR: BPR is a personalized ranking framework, a general learning algorithm to obtain the maximum a posteriori estimator via Bayesian analysis.
- NCF: NCF is a recommendation framework based on neural network collaborative filtering, which uses matrix factorization and neural networks to learn the user–item interaction function.
- DRR-p: DRR is a recommendation method based on deep reinforcement learning. This method models the dynamic interaction process of the recommendation system and uses three state representation structures. DRR-p is a method in the state representation module that exploits the product between items to capture local relationships between features.
- DRR-u: DRR-u is another feature representation method in the DRR framework, which uses the product between items and users to capture the relationship between features.
- DRR-ave: DRR-ave is also the third method of the state representation module in the DRR model. It is the average pooling and inner product of embedding vectors of users and items to obtain the user feature representation.

5.3. Experimental Setup

We randomly divide each user's session in each dataset with a ratio of 8 to 2 for the training and test sets, respectively. For the datasets in the paper, the ratings are between 0–5, so we assume that scores of 4 and 5 are positive, and 0–3 are negative. The number of recent positive scoring items is set to 5, and the discount factor γ is set to 0.9. We apply L_2 norm regularization to the framework method with Adam optimizer to prevent overfitting. In order not to recommend each item repeatedly, the item is removed from the candidate set after it has been recommended. For the reward function, we normalize it to [−1,1] and use the normalized result as the reward.

Specifically, in time step t, the recommendation system recommends item j to user i. If user u does overestimate item j, $rate_{i,j}$ is obtained from the interaction log. Otherwise, the $rate_{i,j}$ is calculated by the simulator as follows:

$$R(s,a) = \frac{1}{2}(rate_{i,j} - 3), \tag{13}$$

We view the recommendation process as modeling an interaction set of size T and adjusting it for different datasets.

5.4. Experimental Results and Analysis

The offline evaluation results are shown in Tables 2–4, where the best experimental results are shown in bold. Our recommendation framework is better than the baseline framework on the indicators Precision@5, Precision@10, NDCG@5, and NDCG@10. It also proves that our proposed framework is effective. Specifically, the accuracy of our proposed framework is much better than that of the control frameworks, which shows that our framework is robust in recommendation relevance. Furthermore, NDCG is also improved, which shows that our framework performs better in the ranking recommendation results.

Table 2. Experimental Results on MovieLens (100k) Dataset.

Model	Precision@5	Precision@10	NDCG@5	NDCG@10
BPR	$0.4814_{+51.9\%}$	$0.6532_{+9.0\%}$	$0.3284_{+188.9\%}$	$0.3838_{+143.4\%}$
NCF	$0.5217_{+40.2\%}$	$0.6066_{+17.4\%}$	$0.3501_{+171.0\%}$	$0.4164_{+124.3\%}$
DRR-p	$0.6866_{+6.5\%}$	$0.6679_{+6.6\%}$	$0.9318_{+1.8\%}$	$0.9314_{+0.3\%}$
DRR-u	$0.6991_{+4.6\%}$	$0.6670_{+6.8\%}$	$0.9429_{+0.6\%}$	$0.9247_{+1.0\%}$
DRR-ave	$0.6848_{+6.8\%}$	$0.6562_{+8.5\%}$	$0.9414_{+0.8\%}$	$0.9278_{+0.7\%}$
DDRCN	0.7312	0.7121	0.9489	0.9341

Table 3. Experimental Results on MovieLens (1M) Dataset.

Model	Precision@5	Precision@10	NDCG@5	NDCG@10
BPR	$0.5200_{+53.3\%}$	$0.6937_{+8.3\%}$	$0.3528_{+170.6\%}$	$0.4046_{+131.6\%}$
NCF	$0.4478_{+78\%}$	$0.6308_{+19.2\%}$	$0.2985_{+219.8\%}$	$0.2564_{+164.1\%}$
DRR-p	$0.7718_{+3.3\%}$	$0.7483_{+0.4\%}$	$0.9490_{+0.6\%}$	$0.9412_{+0.1\%}$
DRR-u	$0.7712_{+3.3\%}$	$0.7393_{+1.7\%}$	$0.9534_{+0.1\%}$	$0.9341_{+0.8\%}$
DRR-ave	$0.7839_{+1.7\%}$	$0.7415_{+1.4\%}$	$0.9508_{+0.4\%}$	$0.9389_{+0.3\%}$
DDRCN	0.7970	0.7516	0.9546	0.9413

Table 4. Experimental Results on Lastfm Dataset.

Model	Precision@5	Precision@10	NDCG@5	NDCG@10
BPR	$0.6173_{+5.6\%}$	$0.6332_{+1.1\%}$	$0.4929_{+91.1\%}$	$0.4359_{+112.3\%}$
NCF	$0.6081_{+7.2\%}$	$0.6390_{+0.2\%}$	$0.4928_{+91.1\%}$	$0.4214_{+119.6\%}$
DRR-p	$0.6266_{+4.0\%}$	$0.5902_{+8.5\%}$	$0.9355_{+0.7\%}$	$0.9152_{+1.1\%}$
DRR-u	$0.6312_{+3.2\%}$	$0.6000_{+6.7\%}$	$0.9343_{+0.8\%}$	$0.9207_{+0.5\%}$
DRR-ave	$0.6492_{+0.4\%}$	$0.6191_{+3.4\%}$	$0.9376_{+0.5\%}$	$0.9238_{+0.2\%}$
DDRCN	0.6516	0.6402	0.9419	0.9254

Therefore, we can draw two conclusions: (1) Our proposed recommendation framework fully mines the feature relationship between users and data, and the results are better than the DRR framework [22]. (2) Modeling the recommendation process as a continuous dynamic process is helpful for the long-term cumulative return of recommendations and the satisfaction improvement of recommendation results.

5.5. Parameter Study

We investigate the impact of the length of the episode on the recommendation performance, as shown in Figure 5. The dataset is MovieLens (1M), and we calculate Precision@5 and the average reward by setting different episodes. The results show that the accuracy and average reward of the recommendation framework increase and reach a peak with the increase in T, and then gradually plateau after decreasing. The reason for is that the recommendation agent continuously explores and exploits during the training process, reaching a peak at around $T = 11$, which is the equilibrium point. Moreover, the small accuracy and average reward in the early stage are due to insufficient interaction between the user and the recommendation agent.

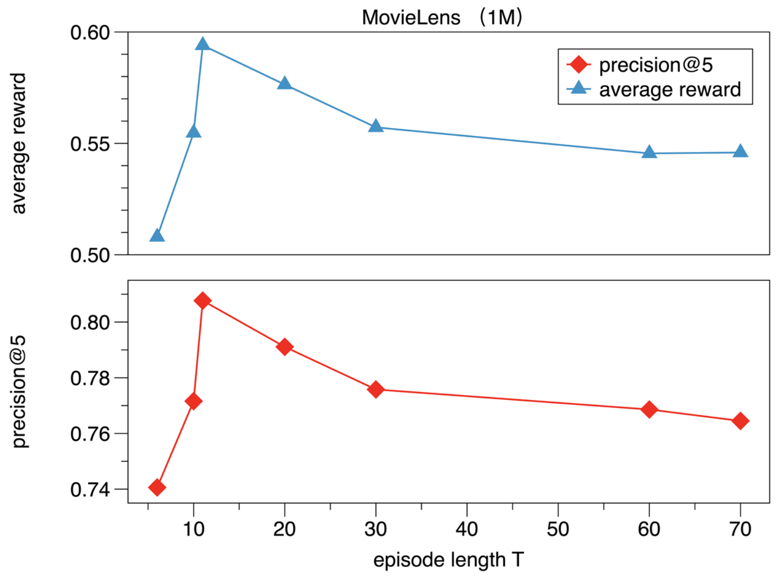

Figure 5. Study of episode length T on the MovieLens dataset.

6. Conclusions

In this paper, we propose a deep deterministic policy gradient recommendation framework incorporating deep cross networks to consider the data cross relationship, and we solve the continuity problem in recommendation systems. It is based on the Actor-Critic architecture, where the Actor network and the Critic network are responsible for recommendation action generation and value estimation, respectively. The dual networks interact continuously with feedback until the model converges. We use a deep cross network to mine the feature relationships between data, and a deep deterministic policy gradient approach to training the DDRCN, based on a greedy policy, to cause the model to converge to a recommendation reward with high cumulative returns. We use the Precision and NDCG metrics in the information retrieval domain to evaluate the DDRN. We conduct relevant experiments on the publicly available datasets MovieLens and Lastfm, and the experimental results outperform the existing baseline methods in terms of Precision@5, Precision@10, NDCG@5, and NDCG@10. In the following research, we will consider issues such as recommendation efficiency, and explore deeper recommendation data features such as semantics.

Author Contributions: Conceptualization, T.G. and S.G.; methodology, T.G., S.G., Q.Z. and J.X.; software, S.G.; validation, S.G.; formal analysis, S.G. and Q.Z.; investigation, T.G.; resources, S.G. and Q.Z.; data curation, T.G. and S.G.; writing—original draft preparation, T.G., S.G. and J.X.; writing—review and editing, T.G., S.G., Q.Z. and J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (grant no. 52130403) and the Fundamental Research Funds for the Central Universities (grant no. N2017003).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The principal datasets used in this research can be downloaded from https://grouplens.org/datasets/movielens/ (accessed on 1 December 2022) and https://grouplens.org/datasets/hetrec-2011/(accessed on 1 December 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ge, M.; Persia, F. A survey of multimedia recommendation systems: Challenges and opportunities. *Int. J. Semant. Comput.* **2017**, *11*, 411–428. [CrossRef]
2. Gomez-Uribe, C.; Hunt, N. The netflix recommendation system: Algorithms, business value, and innovation. *ACM Trans. Manag. Inf. Syst.* **2015**, *6*, 1–19. [CrossRef]
3. Seide, F.; Li, G.; Chen, X.; Yu, D. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In Proceedings of the IEEE Workshop on Automatic Speech Recognition & Understanding, Waikoloa, HI, USA, 11–15 December 2011; pp. 24–29.
4. Wang, R.; Fu, B.; Fu, G.; Wang, M. Deep & cross network for ad click predictions. In Proceedings of the ADKDD'17, Halifax, NS, Canada, 14 August 2017; pp. 1–7.
5. Wang, R.; Shivanna, R.; Cheng, D.; Jain, S.; Lin, D.; Hong, L.; Chi, E. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In Proceedings of the Web Conference, Ljubljana, Slovenia, 19–23 April 2021; pp. 1785–179.
6. Xie, R.; Wang, Y.; Wang, R.; Lu, Y.; Zou, Y.; Xia, F.; Lin, L. Long short-term temporal meta-learning in online recommendation. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event, AZ, USA, 21–25 February 2022; pp. 1168–1176.
7. Shivaram, K.; Liu, P.; Shapiro, M.; Bilgic, M.; Culotta, A. Reducing Cross-Topic Political Homogenization in Content-Based News Recommendation. In Proceedings of the 16th ACM Conference on recommendation Systems, Seattle, WA, USA, 18–23 September 2022; pp. 220–228.
8. Rendle, S. Factorization machines. In Proceedings of the 2010 IEEE International conference on data mining, Sydney, Australia, 13–17 December 2010; pp. 995–1000.
9. Purushotham, S.; Liu, Y.; Kuo, J. Collaborative topic regression with social matrix factorization for recommendation systems. In Proceedings of the International Conference on Machine Learning, Edinburgh, UK, 26 June–1 July 2012; pp. 691–698.
10. Montanés, E.; Quevedo, J.R.; Díaz, I.; Ranilla, J. Collaborative tag recommendation system based on logistic regression. In Proceedings of the European Conference on Machine Learning/ Principles and Practice of Knowledge Discovery in Databases, Bled, Slovenia, 7–11 September 2009; pp. 173–188.
11. Sun, Y.; Pan, J.; Zhang, A.; Flores, A. FM2: Field-matrixed factorization machines for recommendation systems. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 2828–2837.
12. Sedhain, S.; Menon, A.K.; Sanner, S.; Xie, L. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th international conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 111–112.
13. Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & deep learning for recommendation systems. In Proceedings of the 1st workshop on deep learning for recommendation systems, Boston, MA, USA, 15 September 2016; pp. 7–10.
14. Zhou, G.; Mou, N.; Fan, Y.; Pi, Q.; Bian, W.; Zhou, C.; Zhu, X.; Gai, K. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*; Association for the Advancement of Artificial Intelligence: Palo Alto, CA, USA, 2019; pp. 5941–5948.
15. Liu, H.; Wei, Y.; Yin, J.; Nie, L. HS-GCN: Hamming Spatial Graph Convolutional Networks for Recommendation. In *IEEE Transactions on Knowledge and Data Engineering*; IEEE: Piscataway, NJ, USA, 2022.
16. Xu, C.; Guan, Z.; Zhao, W.; Wu, Q.; Yan, M.; Chen, L.; Miao, Q. Recommendation by users' multimodal preferences for smart city applications. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4197–4205. [CrossRef]
17. Wang, K.; Zou, Z.; Deng, Q.; Tao, J.; Wu, R.; Fan, C.; Chen, L.; Cui, P. Reinforcement learning with a disentangled universal value function for item recommendation. In*Proceedings of the AAAI Conference on Artificial Intelligence*; Association for the Advancement of Artificial Intelligence: Palo Alto, CA, USA, 2021; Volume 35, pp.4427–4435.
18. Zou, L.; Xia, L.; Ding, Z.; Song, J.; Liu, W.; Yin, D. Reinforcement learning to optimize long-term user engagement in recommendation systems. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2810–2818.
19. Zou, L.; Xia, L.; Du, P.; Zhang, Z.; Bai, T.; Liu, W.; Nie, J.Y.; Yin, D. Pseudo Dyna-Q: A reinforcement learning framework for interactive recommendation. In Proceedings of the 13th International Conference on Web Search and Data Mining, Houston, TX, USA, 3–7 February 2020; pp. 816–824.

20. Ma, J.; Zhao, Z.; Yi, X.; Yang, J.; Chen, M.; Tang, J.; Hong, L.; Chi, E.H. Off-policy learning in two-stage recommendation systems. In Proceedings of The Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 463–473.
21. Xin, X.; Karatzoglou, A.; Arapakis, I.; Jose, J.M. Supervised Advantage Actor-Critic for Recommender Systems. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event, AZ, USA, 21–25 February 2022; pp. 1186–1196.
22. Liu, F.; Tang, R.; Li, X.; Zhang, W.; Ye, Y.; Chen, H.; Guo, H.; Zhang, Y. Deep reinforcement learning based recommendation with explicit user-item interactions modeling. *arXiv* **2018**, arXiv:1810.12027.
23. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. *Int. Conf. Mach. Learn.* **2014**, *32*, 387–395.
24. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2021**, arXiv:1509.02971.
25. Li, Z.; Zhao, H.; Liu, Q.; Huang, Z.; Mei, T.; Chen, E. Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1734–1743.
26. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, 18–21 June 2009; pp. 452–461.
27. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.S. Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.