A Comprehensive Analysis of Recommendation Algorithms Based on Deep Reinforcement Learning

Rui Wang

Software engineering, Xi'an Jiaotong university, Xi'an, China
rainy123@stu.xjtu.edu.cn

Abstract. Contemporary recommendation systems encounter the challenges posed by information overload and personalized user needs. Recently, there has been a widespread application of deep reinforcement learning algorithms (DRL) to tackle the aforementioned issues. The paper provides a detailed introduction to the basic principles and associated algorithms of DRL. It categorizes recommendation algorithms employing deep reinforcement learning into single-agent RL and multi-agent RL. Representative directions in each category are introduced, their design concepts analyzed, and the advantages and disadvantages of these methods summarized. Specifically, an in-depth analysis of single-agent algorithms is performed. These algorithms are categorized into model-free RL, model-based RL, and hierarchical RL, and the characteristics and current status of each method are discussed. Finally, the paper concludes by summarizing the entire content and analyzing the future research directions and corresponding development trends in this field.

Keywords: Deep reinforcement learning; recommender system; policy optimization; model prediction

1 Introduction

A recommendation system is a software tool that predicts and matches the user's preferences or ratings to provide interesting information, assisting users in discovering products that align with their interests. Such systems play a significant role in numerous prominent companies, including Google, Facebook, and Amazon, and find extensive application across various domains.

Over the past two decades, recommendation systems have received extensive attention and have introduced many different recommendation algorithms. Significant progress has been made in the field of recommendation systems (RS). The field of recommendation algorithms has progressed from traditional collaborative filtering, content-based recommendations, and matrix factorization to newer algorithms employing deep learning (DL). Nonetheless, traditional recommendation techniques encounter challenges including the cold start problem, uncertainty, provision of low-quality recommendations, and substantial computational costs. DL models often suffer from distribution shift issues, i.e., they are unable to effectively reflect rapidly

© The Author(s) 2023
P. Kar et al. (eds.), *Proceedings of the 2023 International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2023)*, Advances in Computer Science Research 108,
https://doi.org/10.2991/978-94-6463-300-9_36

changing real user preferences. These models also have issues of excessive emphasis on short-term rewards and neglect of long-term contributions.

In contrast, reinforcement learning (RL) trains an agent by interacting with the environment and learning from interaction trajectories. This agent can infer user preferences dynamically by utilizing real-time user feedback. Combining DL with traditional RL methods, known as deep reinforcement learning (DRL). DRL optimizes the environment using environmental rewards without requiring training data, making it especially well-suited for solving problems in recommendation systems. DRL can effectively address problems with a large state and action space.

Given the insufficient comprehensive analysis of DRL in the field of recommendation systems, this article provides a detailed introduction to the application status, aiming to provide a comprehensive and up-to-date review of DRL in recommendation algorithms.

Section 1 of this article introduces the relevant background and early research of DRL. Section 2 analyzes and summarizes single-agent reinforcement learning, further classifies them based on existing classification methods, and provides a detailed introduction to the basic ideas and advantages and disadvantages of each sub-method. Section 3 introduces and summarizes multi-agent reinforcement learning. The overall classification results are shown in Fig. 1 as an example. Section 4 summarizes the discussion on DRL-based recommendation systems and their future development trends.

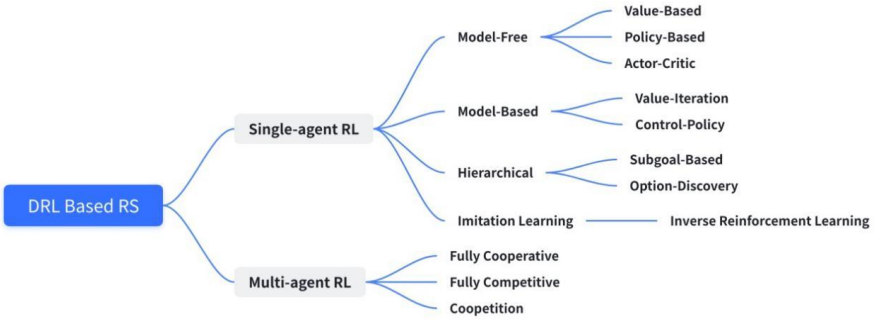

Fig. 1. Classification of recommendation algorithms based on DRL (Original)

2 Introduction to the Background and Early Research

The concept of DRL was introduced by Mnih et al. in 2013 [1]. It is an emerging technology that combines DL and RL. DRL trains an agent by learning the interaction process between the agent and the environment, enabling it to make optimal decisions in complex environments. The core idea of DRL is value-based learning, which guides the agent's decision-making by learning the value function of each state. In DRL, neural networks are utilized to approximate the value function, and the network parameters are updated using the backpropagation algorithm to improve the prediction of each state's value.

DRL-based recommendation systems consist of three fundamental components: environment construction, state representation, and recommendation policy learning. The goal of environment construction is to establish an environment based on users' historical behaviors. This environment provides state representations, which encompass user information such as historical behaviors and demographic data. Recommendation policy learning is a critical aspect of understanding and predicting users' future behaviors. DL-based RS captures users' interests and update the recommender through user feedback, such as ratings or click behavior. In contrast, DRL-based RS receives rewards from the environment. The rewards provided by the environment are predefined functions that incorporate multiple factors. The detailed process mapping of DRL-based recommendation systems can be found in Fig. 2.

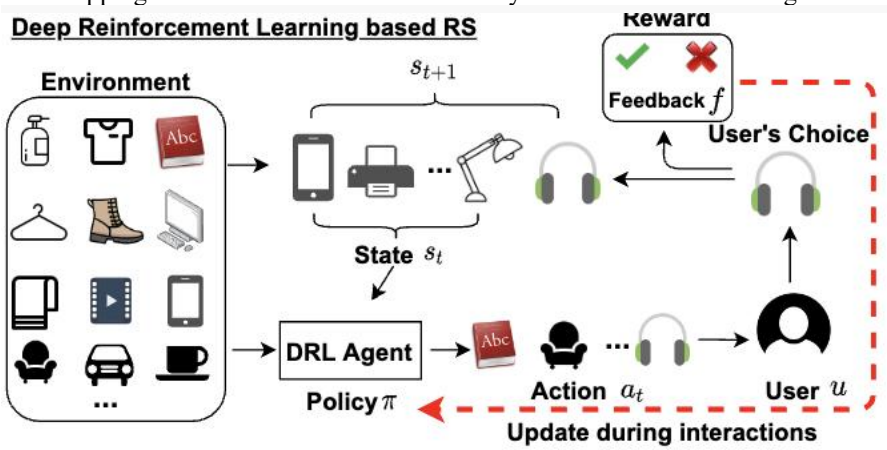

Fig. 2. Processing Flow of Recommendation Systems Based on Deep Reinforcement Learning [1].

3 Recommender Algorithm Based on SARL

The recommendation algorithm based on single-agent reinforcement learning (SARL) is a method that utilizes reinforcement learning techniques for personalized recommendations. This algorithm learns the preferences and behaviors of users through a single agent and generates personalized recommendations based on the learned strategies. The agent plays the role of the decision-maker in the recommendation system, choosing recommended items based on the current user's state and past behavior. It interacts with the environment to acquire user feedback and reward signals, which then guide the updating of its strategies. By continuously interacting and learning from users, the agent can gradually optimize the recommendation strategy, improve the accuracy and personalization of recommendations, and ultimately achieve the optimal recommendation effect.

The recommendation algorithm based on SARL will be divided into three categories based on the model type: model-free reinforcement learning (MFRL),

model-based deep reinforcement learning (MBRL), and deep hierarchical reinforcement learning (DHRL).

3.1 Recommender algorithm based on MFRL

MFRL does not require explicitly constructing an environment model but instead directly interacts with the environment to search and optimize strategies. By employing deep neural networks as a means to represent policies and utilizing optimization algorithms to update network parameters, the objective is to maximize the cumulative rewards. This approach is flexible, applicable to complex control problems, and can efficiently utilize a large amount of sample data for learning. However, in practical applications, challenges such as low sample efficiency and training instability still need to be addressed. MFRL can be further categorized into three main types: policy-based algorithms, value-based algorithms, and Actor-Critic.

Policy-based MFRL algorithms. These algorithms are based on strategies that do not rely on pre-modeling user behavior and preferences. Instead, they directly learn a recommendation strategy to generate optimal recommendation results. The algorithm learns user feature representation through a deep neural network in the recommendation system and predicts the level of interest users have in different items. Then, based on the prediction values, a policy optimization algorithm (REINFORCE or PPO) is used to optimize the neural network, maximizing user satisfaction in the generated recommendation list.

The advantages of this algorithm are its adaptability to dynamically changing user preferences, ability to handle sparse and cold-start problems, and flexibility in handling different recommendation scenarios.

REINFORCE estimates the gradient of the policy by sampling trajectories and uses this gradient to update policy parameters. Chen proposes a method to improve the REINFORCE recommendation system using a user response model [2]. First, an offline dataset is used to train a user response model that can predict user responses to recommendation results. Then, in the REINFORCE algorithm, the user response model is introduced as an additional reward signal, combined with environmental rewards. The advantage is that by combining the rewards from the environment and the user response model, the quality of recommendation results can be more accurately evaluated, guiding the training of the policy network more effectively.

PPO (Proximal Policy Optimization) collects sample data from multiple policy networks and uses importance sampling to estimate the relative advantages between different policies. In PPO, the magnitude of policy updates is constrained by comparing the ratio of probabilities before and after policy updates with a predetermined clipping threshold, thus avoiding instability in the policy optimization process.

Wu proposes a dynamic pricing method that combines the PPO algorithm with online reviews [3]. This method aims to optimize pricing decisions by leveraging customer feedback to maximize revenue. The method utilizes a policy network that

takes online reviews as additional inputs to formulate pricing decisions. The policy network takes online reviews along with other relevant information as inputs and outputs a pricing strategy to optimize pricing decisions. This approach improves training stability and sample efficiency, ultimately enhancing the revenue and customer satisfaction of businesses.

Value-Based MFRL algorithms. The fundamental concept underlying value-based MFRL algorithms involves evaluating various recommendation choices through the establishment of a value function, which is subsequently used to guide policy updates. This approach utilizes neural networks for estimating the value of states, effectively decoupling exploration and exploitation within recommendation systems. The accuracy of the value function can be improved by optimizing the neural network. The backpropagation algorithm allows the neural network to accurately estimate the value of each state and select actions based on their respective values. The recommendation system can generate personalized recommendations by utilizing estimated values obtained from a neural network. These recommendations are tailored to the user's behavior and preferences, allowing the model to achieve adaptive learning. Typical algorithms include Sarsa and DQN.

State-Action-Reward-State-Action (SARSA) is used to solve Markov Decision Process (MDP) problems and is suitable for online learning scenarios. The core idea behind Sarsa is to update the policy based on the current action and state, in addition to the value of the next action and state. By continuously iterating and updating, one can obtain the optimal policy.

The study proposes a personalized video recommendation engine called EmoWare, which utilizes context-aware collaborative filtering to capture users' emotional reactions to recommended videos [4]. The Sarsa algorithm is utilized to train a Deep Bidirectional Recurrent Neural Network (DBRNN) which acquires contextual patterns and produce fresh video sequences to enhance recommendation systems. EmoWare outperforms mainstream methods and effectively simulates users' emotional preferences. However, the system faces the cold-start problem when users interact without prior knowledge, resulting in high error rates initially.

The Deep Q-Network (DQN) leverages deep neural networks to approximate the Q-function and learn the optimal action-value function for guiding the decision-making of an intelligent agent. DQN accepts the current state as input and produces the Q-values for all possible actions. The optimal action is chosen by selecting the action with the highest Q-value. To enhance learning stability and effectiveness, DQN incorporates experience replay during the training process. Its advantages lie in its capability to handle high-dimensional state spaces and continuous action spaces.

Lei Y proposes a user-specific deep Q-learning method (UDQN) for interactive recommendation in explicit feedback recommendation systems [5]. The approach formulates the interactive problem as an MDP for each target user, taking into account multi-step feedback from users. The approach utilizes matrix factorization to construct a latent state space that is specific to each user.

Zheng introduces a framework, as shown in Fig. 3 based on the Double DQN (DDQN) algorithm for news recommendation [6]. The framework approximates the

optimal action-value function. DDQN employs two distinct neural networks for estimating Q-values. This approach diminishes the overestimation of Q-values and enhances the stability of the learning process. DDQN builds user profiles that reflect the frequency of user returns after a recommendation, treating user return patterns as supplementary labels for click/non-click, capturing more user feedback information. This framework effectively models dynamic news features and user preferences, and explicitly plans for the future to achieve higher long-term rewards.

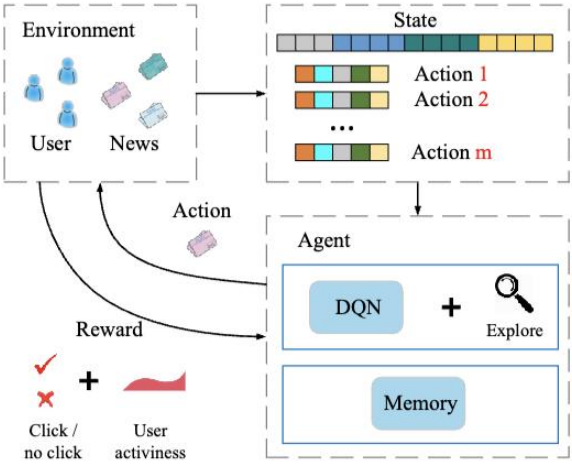

Fig. 3. News Recommendation System Framework based on DRL [6]

Actor-Critic. The Actor-Critic algorithm is composed of two neural network models: the Actor and the Critic. The Actor determines actions based on the current state and transmits them to the environment. The environment provides the next state and the reward signal. The Critic approximates the value of actions by considering the current state and action, and calculates an advantage function that directs the policy update of the Actor. The Actor-Critic algorithm can learn both the policy and value function simultaneously to optimize reinforcement learning tasks.

Deep Deterministic Policy Gradient (DDPG) is an extension of the Actor-Critic algorithm specifically designed for reinforcement learning tasks in continuous action spaces. The DDPG algorithm utilizes experience replay and target networks to train neural networks and stabilize the training process, reducing data correlation and oscillations during training.

Zhao proposes a method to solve the ranking problem in list recommendation [7]. The policy network takes the user's historical behavior sequence and candidate item features as input, and outputs a ranking score for the candidate items to be sorted. In each iteration's parameter update phase, DDPG is used to adjust the parameters. The advantage of this approach is that it can directly optimize the policy, avoiding the need for manual model tuning in traditional methods, while improving the stability and convergence speed of training.

3.2 Recommender algorithm based on MBRL

Model-based reinforcement learning (MBRL) combines the ideas of model prediction and policy optimization. The model predicts state transitions and the reward function, storing information about the dynamics of the environment. The policy selects actions based on the surrounding environment. The agent learns about the dynamic characteristics of the environment by building and utilizing a model for policy optimization.

The difference between MBRL and MFRL in recommendation systems is illustrated in Fig. 4. The agent constructs a model to capture user behavior and feedback. By inputting the current state and recommended action of the user, the model is able to generate the user's subsequent state and provide feedback on the recommended action. Through training this model, the agent can learn user behavior patterns and preferences, which enables predicting the user's preference for different recommendation results. The agent can use the model to predict future states and user feedback and select the optimal recommendation strategy based on these predictions to maximize user satisfaction. The core of MBRL consists of two elements: the model and the policy. Therefore, based on these two elements, we can classify it into model-based value iteration methods and model-based control policy methods.

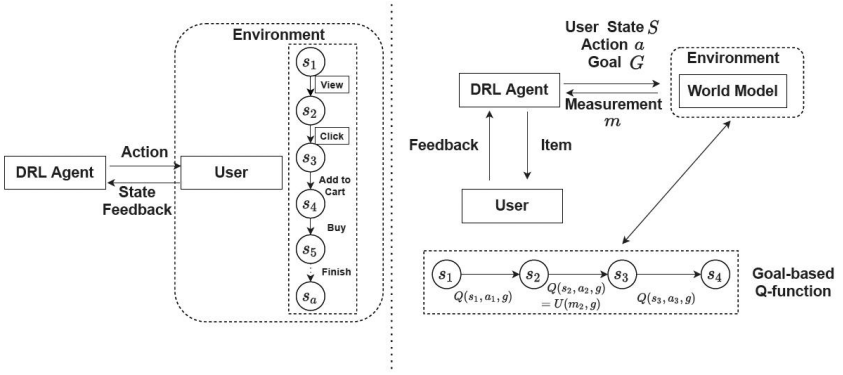

Fig. 4. shows sample trajectories to demonstrate the differences between MFRL and MBRL in recommendation methods [8].

Model-based Value Iteration Method. In this type of method, a neural network model or other function approximators are used to establish a model for predicting the state transitions and reward functions in the environment. These predicted results are utilized to calculate the value function, which represents the anticipated returns of different actions in the present state. Subsequently, the value iteration algorithm is applied to update the value function until convergence is achieved, aiming to rewards that accumulate over the long-term. At each iteration, the model predicts the transitions of the environmental state and the reward functions to optimize the system.

Creswell A presents a framework called GoalRec, which adopts a decoupled value function formalization called Value-Directed Fine-Grained Pruning (VDFP) and

introduces a novel decomposition of the general value function to handle user features and item features [9]. Firstly, an autoencoder is used to decouple the user features and item features, representing them as two independent low-dimensional vectors. Then, the aforementioned general value function is trained using these low-dimensional vectors as inputs to accurately predict the user's preference. Value iteration guides decision-making by calculating the value function for every state, effectively tackling three crucial challenges in recommendation systems: a large state and action space, a high variance environment, and non-specific reward settings in recommendations.

Huang proposes a novel interactive recommendation system called the Top-N system, which is based on DRL [10]. The paper introduces a model that considers the recommendation process as MDP. To simulate the interaction between RS and users, Recurrent Neural Networks (RNN) are used. Moreover, dynamic programming is employed to solve MDP problems. The model gradually improves its optimal policy by iteratively updating the Q-value function, which estimates the expected long-term return when specific actions are taken in certain states. The objective is to maximize the anticipated long-term return, computed as the sum of the immediate reward and discounted future rewards. The immediate reward measures the accuracy of recommendations at each step, while the discounted future rewards denote the anticipated long-term rewards in future steps. In terms of long-term recommendation hit rate and Normalized Discounted Cumulative Gain (NDGG), the model outperforms previous Top-N methods.

Model-based control strategy approach. This method involves establishing a model to predict the state transition and reward function of the environment, which is then utilized to generate a policy.

This achievement is obtained through the solution of an optimal control problem. The predicted results from the model serve as the dynamics of the environment, while control theory methods like the state-action value function guide decision-making. The objective is to design a policy that can maximize long-term cumulative rewards by selecting the optimal action at each state.

Chen presents the Cascading Deep Q Network (CDQN), a framework designed to tackle the challenges posed by recommendation systems that operate in environments with unknown rewards and environment dynamics [11]. A user model is created utilizing Generative Adversarial Networks (GAN), comprising a generator and a discriminator. The generator creates behavioral sequences for virtual users, whereas the discriminator differentiates between behavioral sequences of real users and virtual users. The user model is utilized to simulate user behavior dynamics and learn their reward function. It is then combined with CDQN to create a simulation environment. The predicted results are utilized for designing the optimal control strategy to recommend items and provide real-time suggestions. The CDQN algorithm surpasses other benchmark methods in terms of cumulative rewards and click-through rate (CTR).

3.3 Recommender algorithm based on DHRL

The core techniques of deep hierarchical reinforcement learning (DHRL) are semi-Markov decision process (SMDP) and temporal abstraction 12]. DHRL is commonly used to address three problems: sparse rewards, sequential decision-making, and weak transferability 13].

DHRL divides the recommendation process into multiple levels, each with different objectives and strategies. For instance, the bottom level consists of a model-based recommender that learns user behavior patterns; the middle level is a deep learning-based feature extractor that learns representations of user interests; and the top level is a reinforcement learning-based policy optimizer that selects the optimal recommendation strategy based on user feedback. DHRL can provide more personalized and accurate recommendation results. Depending on the difference in solution approaches, DHRL can be divided into G-DHRL and O-DHRL.

G-DHRL. G-DHRL, a subgoal-based Deep Hierarchical Reinforcement Learning framework (Fig. 5), is designed to tackle the challenges posed by high-dimensional state spaces and long time delays in complex tasks, particularly for homogeneous recommendation projects involving a single item type. t decomposes the task into multiple subtasks, allowing the agent to learn and optimize strategies for these subtasks in order to achieve higher-level objectives. The strategies for these subtasks can be acquired using a variety of reinforcement learning algorithms. These subtasks can be defined based on higher-level task objectives, prior experience, or can be automatically discovered and defined during the learning process. Therefore, G-DHRL encompasses foresight subgoals (FG) and hindsight subgoals (HG).

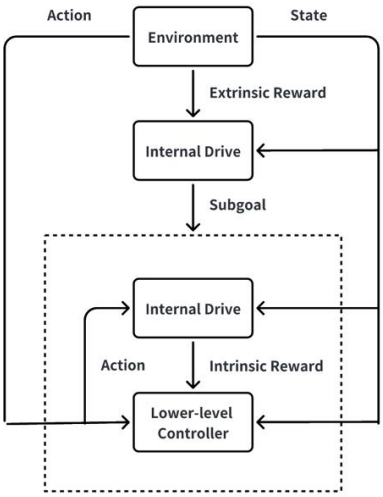

Fig. 5. Generalized Deep Hierarchical Reinforcement Learning (G-DHRL) framework 14]

Zhang proposed a personalized recommendation model that combines attention mechanism and hierarchical reinforcement learning for MOOC courses, taking into account students' interests and learning goals 15]. Through the attention-based recommendation model, the influence of different target courses on historical courses can be identified. The model divides the objectives into two levels. At a high level, the agent decides whether to modify the user's historical course configuration and chooses the most suitable low-level task based on the student's learning goals and preferences. At a low level, the task predicts students' interests in various courses by analyzing their historical behaviors and features. The agent makes recommendations based on completed tasks, which are considered as students' historical behaviors and are part of the HG model. This hierarchical approach can improve the accuracy and personalization of recommendations, help students better choose suitable courses, and improve learning effectiveness and satisfaction.

Xie proposed the Clustering-based Hierarchical Reinforcement Learning Network (CHRL) to overcome the challenges posed by noisy borrowing behaviors and sparse data in digital libraries 16]. The study used the NAIS model based on attention mechanism to perform sequence modeling, transforming users' historical reading records into state representations. Then, clustering methods were used to group books and assign a representative state to each group. This reference considers users' borrowing order data as MDP to solve the interaction problem that may mislead recommendation systems. The main objective is to assess if the complete sequence requires modification, whereas the subordinate task aims to determine if individual elements in the sequence should be deleted. The agent assigns delayed rewards based on both the environment and the modified sequence, which are part of the HG model.

O-DHRL. Option Discovery Deep Hierarchical Reinforcement Learning (O-DHRL) decomposes the decision-making process into multiple hierarchical subtasks, with distinct goals and constraints. High-level tasks set goals and allocate resources, while low-level tasks execute specific tasks and learn skills. Utilizing methods like shared information and collaborative training, the synergy and coordination between the hierarchical structure and subtasks are trained. ODHRL achieves improved constraint balance across multiple goals and delivers comprehensive and personalized decision outcomes.

According to reference 17], a novel framework called HRL-Rec is proposed for heterogeneous recommendation projects. This framework is employed to recommend diverse types of heterogeneous items from various information sources within a unified recommendation system. These items have different characteristics, so different channels are typically used for recommendation to facilitate model decoupling and customization. The framework consists of two tasks: channel recommendation and item recommendation. The low-level agent, known as the channel selector, is responsible for generating personalized channel lists. On the other hand, the high-level agent, known as the item recommender, recommends specific items from diverse channels while adhering to channel constraints. HRL-Rec has achieved the best experimental results in various offline and online experiments and

has been successfully deployed in the WeChat "Discover" online system, serving a large number of users.

4 Recommender algorithm based on MARL

Multi-Agent Reinforcement Learning (MARL) is a subfield of reinforcement learning specifically developed to address the challenges posed by multiple agents in a shared environment, aiming to resolve decision-making and learning tasks. MARL sets itself apart from SARL by taking into account the interactions and cooperation among multiple agents, as well as exploiting parallel computing techniques to enhance the learning process.

MARL can address resource allocation issues in recommendation systems, where limited resources need to be allocated to different users. By enabling collaborative learning among multiple agents, MARL dynamically allocates resources based on user needs and system resource availability, thereby enhancing overall system efficiency and user satisfaction.

MARL can also tackle multi-objective optimization problems in recommendation systems. In such systems, there are often multiple objectives that need to be simultaneously considered, such as balancing recommendation accuracy and diversity. Through collaborative learning among multiple agents, MARL adjusts recommendation strategies dynamically based on user feedback and system objectives, thus achieving multi-objective optimization.

Furthermore, MARL can address the game-theoretic challenges in recommendation systems. Due to the interdependencies and competition among different users and items in these systems, MARL employs multiple agents for game-based learning. By adapting recommendation strategies dynamically according to user feedback and competitive situations, MARL achieves competitive advantage for the system and enhances user satisfaction.

Therefore, MARL can be classified into three categories based on problem types: complete cooperation, complete competition, and cooperation with competition.

4.1 Fully Cooperative MARL Algorithms

Fully cooperative MARL algorithms facilitate the pursuit of shared global objectives by multiple agents, necessitating collaboration for their achievement. Effectively fostering collaboration among multiple agents and addressing potential instability and uncertainty are the main challenges in this context.

Each agent serves as a recommender, providing recommendations of a set of items to a user as its action, and receiving user's feedback as its reward. The state of each agent consists of not only its own recommender state but also the recommender states of other agents. The aim of this collaborative approach is to enhance the performance of the overall system.

Multi-Agent Recurrent Deterministic Policy Gradient (MA-RDPG) was proposed in literature 18] for joint optimization of multi-domain ranking. The model composes

of three main modules: Central Critic, Distributed Actor, and Communication Component. Each actor network employs a deterministic policy to map states to actions. In order to facilitate collaboration, the communication component encodes the local observations and actions into message vectors, which are then exchanged among the agents. The decision of each actor relies on both its own past observations and actions as well as those of other actors. In multi-domain ranking tasks, MA-RDPG outperforms traditional methods, exhibiting enhanced robustness and stability.

4.2 Fully Competitive MARL Algorithms

The agents in the system learn the optimal strategy using fully competition MARL algorithms. Each agent has its own observation and action space, allowing them to improve their recommendation effectiveness through competition with other agents. This approach has the advantage of stimulating innovation and progress among intelligent agents through competition.

Zhang proposes a dynamic academic collaboration recommendation, where the selection of collaborators is optimized based on several different measures of scholar similarity 19]. Firstly, each researcher in the academic community is considered as an independent agent. Then, these agents learn the best collaboration recommendation strategy through interaction with the environment. To achieve this, each agent competes with other agents and updates its strategy based on the competition results. Authors use the Q-learning algorithm to train the agents, iteratively optimizing their strategies to obtain the dynamic academic collaboration recommendation model. This method is able to promptly adjust the recommendation strategy in response to changes in collaborators' knowledge and skills, thereby greatly promoting collaboration in scientific research.

4.3 Cooperative And Competitive MARL Algorithms

The MARL algorithms involve both cooperation and competition. Agents need to consider the behaviors of other agents and adjust their strategies based on the relationship of cooperation and competition. Agents may need to cooperate to achieve common goals while also competing to gain more rewards or resources.

Hu proposes a framework called Master that treats charging station management as a MARL task 20]. The charging stations compete for limited charging resources while sharing the charging demands of electric vehicles and the state information of the power grid to optimize the overall charging system efficiency. A multi-Actor-Critic framework is developed to coordinate recommendations between agents. The framework also introduces a strategy known as delayed access, which aims to evaluate the influence of future charging competition while the training process is ongoing. Additionally, a dynamic gradient reweighting strategy is employed to effectively guide the optimization direction in an adaptive manner. Experiments demonstrate that, compared to nine baseline methods, Master achieves the best overall performance and can effectively recommend charging stations for electric vehicles.

5 Conclusion

This article introduces recommender algorithms based on deep reinforcement learning. By grouping them according to similarities and differences in algorithms and applications, they are divided into two categories: single-agent and multi-agent, which can be organized and summarized in a structured manner. The characteristics and advantages and disadvantages of each category are described. Single-agent algorithms include model-based, model-free, and hierarchical reinforcement learning, while multi-agent algorithms are suitable for multi-user or recommendation agent scenarios. Through deep reinforcement learning, recommender systems can provide more accurate and effective recommendations.

Deep reinforcement learning is a powerful technique with extensive prospects and significant application value within the domain of recommendation systems. However, further research and practice are still needed to address the issues and challenges, and to promote innovation and development of recommender algorithms. Further research can enhance algorithm performance and efficiency, optimize the training speed and generalization ability of models. It can also integrate with traditional recommendation algorithms to leverage their respective strengths and improve the performance and effectiveness of recommender systems. Additionally, improving the interpretability of algorithms should enable users to understand the reasons and basis behind recommendation results, promoting the development and innovation of recommender systems.

References

1. Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning. arXiv, 2013: 1312.5602.
2. Chen M, Chang B, Xu C, et al. User response models to improve a reinforce recommender system//Proceedings of the 14th ACM International Conference on Web Search and Data Mining. 2021: 121-129.
3. Wu C, Bi W, Liu H. Proximal policy optimization algorithm for dynamic pricing with online reviews. Expert Systems with Applications, 2023, 213: 119191.
4. Tripathi A, Ashwin T S, Guddeti R M R. EmoWare: A context-aware framework for personalized video recommendation using affective video sequences. IEEE Access, 2019, 7: 51185-51200.
5. Lei Y, Li W. Interactive recommendation with user-specific deep reinforcement learning. ACM international conference on knowledge discovery & data mining, 2019, 13(6): 1-15.
6. Zheng G, Zhang F, Zheng Z, et al. DRN: A deep reinforcement learning framework for news recommendation//Proceedings of the 2018 world wide web conference. 2018: 167-176.
7. Zhao X, Zhang L, Xia L, et al. Deep reinforcement learning for list-wise recommendations. arXiv preprint arXiv:1801.00209, 2017.
8. Wang K, Zou Z, Deng Q, et al. Reinforcement learning with a disentangled universal value function for item recommendation// Proceedings of the AAAI conference on artificial intelligence. 2021, 35(5): 4427-4435.

9. Creswell A, White T, Dumoulin V, et al. Generative adversarial networks: An overview. IEEE signal processing magazine, 2018, 35(1): 53-65.
10. Huang L, Fu M, Li F, et al. A deep reinforcement learning based long-term recommender system. Knowledge-Based Systems, 2021, 213: 106706.
11. Chen X, Li S, Li H, et al. Generative adversarial user model for reinforcement learning based recommendation system//International Conference on Machine Learning, 2019: 1052-1061.
12. Sutton R S, Precup D, Singh S. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial intelligence, 1999, 112(1-2): 181-211.
13. Huang Z, Zhang L, Liu Q, et al. Research and Development on Deep Hierarchical Reinforcement Learning . Journal of Software, 2022, 34(2): 733-760.
14. Kulkarni T D, Narasimhan K, Saeedi A, et al. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. Advances in neural information processing systems, 2016, 29.
15. Zhang J, Hao B, Chen B, et al. Hierarchical reinforcement learning for course recommendation in MOOCs// Proceedings of the AAAI conference on artificial intelligence. 2019, 33(01): 435-442.
16. Wang X, Wang Y, Guo L, et al. Exploring clustering-based reinforcement learning for personalized book recommendation in digital library. Information, 2021, 12(5): 198.
17. Xie R, Zhang S, Wang R, et al. Hierarchical reinforcement learning for integrated recommendation// Proceedings of the AAAI conference on artificial intelligence. 2021, 35(5): 4521-4528.
18. Feng J, Li H, Huang M, et al. Learning to collaborate: Multi-scenario ranking via multi-agent reinforcement learning//Proceedings of the 2018 World Wide Web Conference. 2018: 1939-1948.
19. Zhang Y, Zhang C, Liu X. Dynamic scholarly collaborator recommendation via competitive multi-agent reinforcement learning//Proceedings of the eleventh ACM conference on recommender systems. 2017: 331-335.
20. Hu Y, Da Q, Zeng A, et al. Reinforcement learning to rank in e-commerce search engine: Formalization, analysis, and application// Proceedings of the eleventh ACM conference on recommender systems. 2018: 368-377.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.