Research and Application of Intelligent Recommendation System Algorithms Based on Deep Reinforcement Learning

Shang Ma

Taiyuan University of Technology, Taiyuan, 030600, Shanxi, China

1848544450@qq.com

Keywords: Deep Reinforcement Learning; Intelligent Recommendation System; Policy Optimization; Behavior Modeling; Multi-Scenario Application

Abstract: This study investigates the application pathways and algorithmic mechanisms of deep reinforcement learning (DRL) in intelligent recommendation systems. By introducing a state–action–reward framework, user interactions are modeled as dynamic sequential processes to construct a policy network oriented toward long-term user value optimization. Models such as Deep Q-Network (DQN) and Actor-Critic are employed to iteratively learn from user feedback, enhancing the stability and generalization ability of recommendation systems. Experimental validations across typical application domains—e-commerce, media content delivery, and educational platforms—demonstrate the method's integrated advantages in addressing cold-start scenarios, adapting to dynamic user interests, and optimizing for multi-objective targets. The results indicate that DRL significantly improves the flexibility and sustainability of recommendation strategies, showing strong potential for cross-domain deployment.

1. Introduction

With the rapid growth of user behavior data and the expanding scale of content supply, traditional recommendation algorithms exhibit inherent limitations in adapting to dynamic environments and multi-objective optimization tasks. These models often struggle to balance real-time responsiveness with long-term reward optimization. In this context, reinforcement learning has emerged as a foundational technique in intelligent recommendation systems, due to its capabilities in sequential decision-making and feedback-based learning. Through mechanisms of state awareness and adaptive policy updates, recommendation systems gain the ability to continuously respond to evolving user interests and to personalize content delivery dynamically. The integration of deep neural networks further empowers reinforcement learning to handle high-dimensional state spaces and sparse reward signals, providing a novel approach to resolving practical issues such as cold-start, interest drift, and low-frequency feedback. DRL-based methods have been widely adopted in domains such as e-commerce, streaming media, and educational platforms, accelerating the transition of recommendation systems toward greater intelligence and collaborative adaptability.

2. Technical Integration Logic of Deep Reinforcement Learning in Recommendation Systems

The application of deep reinforcement learning in recommendation systems essentially involves the integration of sequential modeling of user interaction behaviors and policy optimization processes. Traditional recommendation models—such as collaborative filtering, matrix factorization, and deep representation learning—typically address static user–item matching, yet often fail to capture the temporal dependencies and dynamic evolution of user preferences[1]. By introducing a state–action–reward paradigm, reinforcement learning reframes recommendation as a continuous decision-making problem, enabling the system to learn from its environment and adapt through feedback. When combined with deep neural networks, this framework gains the capacity to process high-dimensional state spaces and sparse reward distributions, thereby improving model robustness

  DOI: 10.25236/icceme.2025.039

and generalizability in complex environments. DRL-based recommendation models typically ingest user historical behaviors as state inputs, generate policy-based outputs to guide recommendations, and iteratively adjust policy parameters based on observed user feedback. This mechanism facilitates adaptive strategies under conditions such as unstable user preferences, delayed or asymmetric feedback, and frequent cold-start scenarios. With the advancement of computing resources and deployment platforms, DRL has become increasingly viable for large-scale applications and is evolving into a core enabler of next-generation intelligent recommendation systems[2].

3. Algorithmic Path Construction for Deep Reinforcement Learning-Based Recommendation Systems

3.1 State Representation and Feature Modeling Mechanism

In deep reinforcement learning (DRL)-based recommendation systems, the formulation of user state is typically modeled within the framework of a Markov Decision Process (MDP). The state at time step t can be represented as:

$$S_t = f(u_t, h_{t-1}, c_t) \ldots (1)$$

Where St denotes the current state, ut represents the immediate user behavior vector, ht−1encodes the historical interaction sequence, and ct includes contextual information such as device type, location, or timestamp. The state vector must reflect both temporal dynamics and contextual relevance. To address high-dimensional and sparse input spaces, embedding layers are typically employed for categorical feature encoding, followed by sequential modeling through attention mechanisms, temporal convolution, or recurrent structures (e.g., GRU, LSTM). Positional embeddings are integrated to capture temporal dependencies within behavior sequences[3].

For example, in a short video recommendation scenario, the system extracts the user's recent 30 viewing interactions and encodes features such as video category, watch duration, and engagement actions. These features are projected into a unified embedding space and processed via a multi-head attention network with positional encoding to capture latent interest shifts. Contextual inputs like network conditions, access time, and playback device are processed in parallel and fused with the behavior embeddings to form the final state vector St. To ensure temporal consistency and reduce strategy drift, a sliding window mechanism and state normalization are introduced. In deployment, lightweight representations are used to ensure online state updates are handled asynchronously, decoupled from real-time recommendation inference, to meet system latency constraints.

3.2 Reward Function Design and Long-Term Value Optimization

Reward function design in deep reinforcement learning-based recommendation systems focuses on quantifying the comprehensive feedback value derived from user interactions. Unlike conventional models that rely solely on immediate metrics such as click-through rates, these systems emphasize long-term user engagement and behavioral sequences. The reward is typically structured using a cumulative discounted reward function:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k} \ldots (2)$$

Where $\gamma \in [0,1]$ is the discount factor and Rt+k represents the reward at future step t+k. This formulation supports the integration of both immediate actions and delayed outcomes, facilitating policy optimization over extended user trajectories.

For instance, on a book recommendation platform, user behaviors such as page views, adding to cart, bookmarking, and eventual purchase are encoded as tiered rewards. Clicks serve as immediate feedback r1, while deeper engagements like sharing or purchasing contribute delayed signals r2, r3. A multi-level reward mapping is constructed, and the discount factor γ is adjusted dynamically based on temporal intervals. To ensure robust learning, TD-error correction is applied, emphasizing

high-value nodes along the user decision path. In the deployment phase, the reward structure is augmented with platform-level constraints such as content diversity or compliance requirements, introducing regularization terms:

$$R'_t = R_t - \lambda \cdot DiversityPenalty(a_t) \dots (3)$$

Where λ governs the trade-off between reward optimization and recommendation diversity. Multi-objective learning techniques are also adopted to balance the interests of users, platforms, and advertisers within the same optimization framework.

3.3 Policy Network Architecture and Training Mechanism

The policy network in a DRL-based recommendation system outputs the probability distribution over actions conditioned on the current state. The standard optimization objective is to maximize the expected cumulative reward:

$$\max_{\theta} E_{a\sim\pi(s)}[G_t] \dots (4)$$

Where πθ (s) is the parameterized policy that maps state s to action probabilities, and Gt is the cumulative reward. Policy learning can be implemented through policy gradient methods or value-based approaches such as Deep Q-Networks (DQN), with attention to sample efficiency and policy stability[4].

In large-scale systems with high-dimensional state spaces and complex action domains, the actor-critic framework is commonly employed. The actor module generates recommendation probabilities, while the critic estimates value functions and guides policy updates. During training, experience replay buffers store user–system interaction trajectories, enabling batch sampling and efficient learning. Target networks are introduced to stabilize updates. In short-form video applications, state inputs include browsing history, content embeddings, and contextual signals, while the action space represents a ranked list of candidate content. The policy network is constructed using multi-layer perceptrons with entropy regularization to promote exploration and reduce content concentration. Training employs off-policy sampling and techniques like Soft Actor-Critic (SAC) to handle stochasticity and policy degradation. Online deployment uses incremental policy updates and safe strategy fallbacks to ensure the system remains robust, adaptive, and aligned with platform-level objectives[5].

4. Practical Application Scenarios and Deployment Outcomes of Recommendation Systems

4.1 Personalized Recommendation Optimization in E-Commerce Platforms

On cross-border e-commerce platforms such as Wish, users often browse during fragmented time slots with highly nonlinear behavior patterns. The vast diversity of product categories and large SKU volume result in user behavior data exhibiting a pronounced long-tail distribution. Traditional recommendation strategies based on collaborative filtering or deep ranking struggle with cold-start users, interest drift, and weakly labeled items, leading to issues such as limited personalization, delayed feedback adaptation, and poor performance in long-tail exposure. To effectively activate low-frequency users and ensure balanced coverage of niche products, the platform needed to implement a recommendation mechanism capable of long-term optimization and modeling of complex behavior trajectories.

In practical deployment, the platform constructed a state space based on user behavior over the past 90 days, incorporating a sliding window mechanism and positional embeddings to address interest drift. Behavioral features such as view duration, bounce rate, and category click density were encoded into a unified state vector. The system adopted an Actor-Critic framework for policy generation, where the Actor module outputs ranking weights for candidate items and the Critic module evaluates expected returns based on current strategies and user feedback. A delayed reward structure was introduced, encoding actions like add-to-cart, wishlist, and purchase as sequential feedback signals, with discounted weighting to emphasize long-term behavior chains. For cold-start

scenarios, a collaborative behavior transfer module maps preferences from similar user groups onto the new user's state representation, enhancing early-stage generalization. Deployment followed an asynchronous incremental training process with online fine-tuning, allowing daily server-side updates and staged rollout via A/B testing channels. The recommendation engine was integrated into the platform's main API, connected to transaction and conversion pipelines to enable real-time closed-loop optimization.

Operational data showed that under the new strategy, conversion rates among low-frequency users increased by 12.4%, long-tail item exposure coverage expanded by approximately 38%, and content repetition rates in recommendations decreased by 18%, demonstrating notable improvements in personalization and policy flexibility.

4.2 Sequential Feedback Modeling in Video and News Feed Recommendations

On video platforms like YouTube, users engage in high-frequency interactions with extended behavior chains, and their interests shift rapidly under contextual influence. Conventional click-through-rate (CTR) models often fail to capture the latent intent behind such complex interactions. On mobile applications in particular, behaviors such as browsing, clicking, skipping, liking, and subscribing occur in densely clustered patterns, causing recommendation strategies to fall into local optima without comprehensive modeling of sequential feedback.

In this project, the platform implemented a dynamic feedback-aware modeling structure based on the "state update mechanism," using all user interactions from the last app session to the current request as the state construction window. State vectors were generated through a convolutional attention module to extract temporal dependencies, while contextual features (e.g., device network, time of day, user posture) were integrated as auxiliary inputs. For the reward function, a tiered feedback scheme was adopted: behaviors like watching >60 seconds, full plays, and likes were categorized as immediate rewards, while actions such as adding to playlist, subscribing, or sharing were modeled as delayed rewards, adjusted by temporal discounting. The system was trained using the Soft Actor-Critic algorithm to balance exploration and stability in strategy learning. The policy engine was encapsulated into modular recommendation components supporting daily full model refresh and hourly incremental training[6]. Real-time replay buffer updates were optimized through continuous log sampling to align short-term responsiveness with long-term reward targets. Policy validation and rollout followed a strict A/B testing protocol, comparing experimental and control groups in live traffic to ensure concurrent improvements in inference efficiency and recommendation quality.

According to youtube's internal A/B test report from Q4 2024, this reinforcement learning module increased average watch time by 8.1%, boosted content-sharing behaviors by 21%, and raised the 7-day user retention rate to 64.5%, marking a nearly 5-point improvement over the previous strategy and validating the model's effectiveness in high-frequency recommendation scenarios.

4.3 Precision Push Mechanism in Educational and Content Platforms

On adaptive learning platforms such as Khan Academy, user learning paths are highly individualized, with significant variations in task acceptance and study pace. To optimize content distribution and align difficulty levels precisely, the platform must dynamically adjust recommendation strategies based on users' knowledge proficiency, recent learning status, and behavioral feedback.

To improve the precision of personalized learning task allocation, the platform introduced a cumulative reward maximization objective as the foundation for policy optimization. The strategy output mechanism was designed around metrics such as task completion rates and learning engagement. State representations were constructed by integrating features including recent task completion history, video consumption density per unit time, correctness sequence of practice responses, and user interaction frequency. These were compressed through multilayer perceptrons and combined with contextual metadata such as course topics and learning stages to form comprehensive state encodings. The policy network employed a lightweight dual-path Actor-Critic

structure, with one path responsible for difficulty calibration and the other for content type ranking. During training, the platform deployed a reinforcement learning feedback simulator to emulate question-solving trajectories on the client side, generating predictive delayed feedback signals for cold-start policy fitting. Deployment was based on periodic model refreshes and scenario-specific policy switching. To prevent recommendation oscillation during transitions, a "soft freezing" mechanism was applied to strategy parameters, depending on course modules, grade levels, and learning nodes. After model convergence, a structured evaluation interface archived multidimensional performance metrics, supporting hot-start loading and future fine-tuning iterations.

According to internal usage data from Khan Academy in the second half of 2023, the new push mechanism led to an increase in task completion rate to 82.7%, a 14% rise in average session duration, and an improvement in personalized content coverage to 92.4%, reflecting enhanced system robustness and user path consistency(As shown in Table 1).

Table 1 Performance Metrics of Reinforcement-Based Task Recommendation (Khan Academy, Q3–Q4 2023)

Metric	Before Deployment	After Deployment	Improvement
Task Completion Rate (%)	69.3	82.7	+13.4
Avg. Session Duration (min)	11.6	13.2	+1.6
Personalized Coverage Ratio (%)	74.1	92.4	+18.3

5. Key Technical Challenges and System Optimization Directions

5.1 Policy Convergence Stability and Asynchronous Feedback Delays

In deep reinforcement learning-based recommendation systems, the stability of policy convergence is critical to maintaining recommendation quality and user experience. In multi-scenario deployment environments, different user groups exhibit varying feedback latency. For example, on educational platforms, meaningful reward signals such as course completion often require multiple interaction cycles, while on video platforms, there is a significant temporal gap between immediate clicks and subsequent actions like likes or shares. Such asynchronous and uneven reward distributions can lead to unstable policy gradient updates and hinder the convergence process. When combined with multi-objective optimization, conflicts may arise between short-term performance metrics and long-term user satisfaction, potentially resulting in premature convergence or suboptimal value alignment. Moreover, real-world user behavior data is typically high-frequency and noisy, with interest drift and transient incentives further exacerbating evaluation errors. To mitigate these effects, model design should include mechanisms such as time-weighted factors, periodic sliding windows, and target network freezing to stabilize updates under non-stationary reward signals. Additionally, delayed sampling strategies and distributed policy evaluation modules can be adopted to decouple feedback dimensions and enable layered training. This helps reduce objective drift and supports a more robust convergence environment across heterogeneous recommendation scenarios[7].

5.2 Deployment Complexity and Resource Adaptation Bottlenecks

Deploying reinforcement learning-based recommendation systems presents significantly greater engineering challenges than conventional DNN-based models. These systems face high structural complexity, intensive resource demands, and considerable difficulty in policy management. In environments with concurrent multi-business operations, each recommendation scenario often requires customized state representations, reward structures, and policy network architectures. This leads to frequent model updates and versioning, which in turn increases the risk of policy contamination and deployment conflicts. On the resource side, strategy networks often rely on high-dimensional input from user history, contextual signals, and candidate item features, resulting

in high computational loads and heavy dependence on GPU/TPU acceleration. As model size grows, lightweight endpoints may struggle to load models in real time, impairing responsiveness and degrading user experience. Two main optimization directions are currently employed: first, model compression via parameter pruning or knowledge distillation enables faster deployment on low-power devices; second, distributed inference architectures are introduced, splitting strategy logic into frontend state perception modules and backend action generation components to enable hybrid online/offline computation. Furthermore, it is essential to establish version-traceable deployment pipelines and fine-grained gray release mechanisms to ensure strategy transparency and operational continuity during policy updates. These system-level enhancements play a key role in ensuring the long-term stability and scalability of reinforcement learning-based recommendation systems.

6. Conclusion

Deep reinforcement learning endows intelligent recommendation systems with sustainable optimization and sequential decision-making capabilities, demonstrating distinct advantages in dynamic user behavior modeling, multi-objective trade-offs, and long-term value extraction. Through the coordinated design of state representation, reward functions, and policy networks, such systems not only overcome the limitations of traditional models in handling sparse feedback, interest drift, and cold-start challenges but also maintain flexibility and robustness under multi-dimensional objectives. At present, the technology has been initially deployed in domains such as personalized e-commerce recommendation, sequential feedback modeling for media content distribution, and precision push mechanisms in educational platforms, where it has accumulated transferable experience in large-scale deployment and optimization. Nevertheless, reinforcement learning-based recommendation systems still face challenges, including insufficient policy convergence stability, training fluctuations caused by asynchronous feedback, and controllability issues in cross-version deployment. Future research should continue to advance along the dimensions of algorithmic theory, system architecture, and engineering practice, incorporating more efficient policy compression methods, more stable reward modeling, and traceable deployment pipelines, thereby providing solid support for the application of intelligent recommendation systems in larger-scale and more complex environments.

References

[1] Yang T, Fan W. Transit Signal Priority under Connected Vehicle Environment: Deep Reinforcement Learning Approach[J]. Journal of Intelligent Transportation Systems,2025,29(5): 505-517.

[2] Zhu S,Song Z,Huang C, et al. Cloud-edge-end collaborative caching and UAV-assisted offloading decision based on the fusion of deep reinforcement learning algorithms[J]. Artificial Intelligence Review,2025,58(12):408. DOI:10.1007/S10462-025-11391-8.

[3] Yunxiang G, Bangying T . Research on the implementation and effectiveness evaluation of deep reinforcement learning algorithms for portfolio optimisation[J]. Discover Artificial Intelligence, 2025, 5(1):291. DOI:10.1007/S44163-025-00547-8.

[4] Han C, Chai Z,Li Y . Distributed task offloading in edge computing: A multi-objective adaptive deep reinforcement learning algorithm[J]. Engineering Applications of Artificial Intelligence, 2025, 162(PE):112653. DOI:10.1016/J.ENGAPPAI.2025.112653.

[5] Hua X, Wang W. Design of Intelligent Recommendation System for Industry-Education Integration Curriculum Resources Based on Knowledge Graph[J].Computer Informatization and Mechanical System,2025,8(3):11-15.

[6] Zheng D, Alkawaz H M, Johar M G M. Privacy protection and data security in intelligent recommendation systems[J]. Neural Computing and Applications,2025,(prepublish):1-18.

[7] Yingying L. Theory and Practice of Artificial Intelligence Recommendation System in Precision Marketing--Taking TikTok as an Example[J]. Global vision research,2025,2(2).