Enhancing New-item Fairness in Dynamic Recommender Systems

Huizhong Guo
Zhejiang University
Hangzhou, China
huiz_g@zju.edu.cn

Zhu Sun
Singapore University of Technology
and Design
Singapore
sunzhuntu@gmail.com

Dongxia Wang*
Zhejiang University
Huzhou Institute of Industrial Control
Technology
Hangzhou, China
dxwang@zju.edu.cn

Tianjun Wei*
Nanyang Technological University
Singapore
tjwei2-c@my.cityu.edu.hk

Jinfeng Li
Alibaba Group
Hangzhou, China
jinfengli.ljf@alibaba-inc.com

Jie Zhang
Nanyang Technological University
Singapore
zhangj@ntu.edu.sg

Abstract

New-items play a crucial role in recommender systems (RSs) for delivering fresh and engaging user experiences. However, traditional methods struggle to effectively recommend new-items due to their short exposure time and limited interaction records, especially in *dynamic recommender systems* (DRSs) where **new-items get continuously introduced** and **users' preferences evolve over time**. This leads to significant unfairness towards new-items, which could accumulate over the **successive model updates**, ultimately compromising the stability of the entire system. Therefore, we propose FairAgent, a reinforcement learning (RL)-based new-item fairness enhancement framework specifically designed for DRSs. It leverages knowledge distillation to extract collaborative signals from traditional models, retaining strong recommendation capabilities for old-items. In addition, FairAgent introduces a novel reward mechanism for recommendation tailored to the characteristics of DRSs, which consists of three components: 1) a **new-item exploration reward** to promote the exposure of dynamically introduced new-items, 2) a **fairness reward** to adapt to users' personalized fairness requirements for new-items, and 3) an **accuracy reward** which leverages users' dynamic feedback to enhance recommendation accuracy. Extensive experiments on three public datasets and backbone models demonstrate the superior performance of FairAgent. The results present that FairAgent can effectively boost new-item exposure, achieve personalized new-item fairness, while maintaining high recommendation accuracy.

CCS Concepts

• **Information systems → Recommender systems**.

---

*Corresponding authors.

Keywords

Recommender Systems, Fairness, New items, AI Ethics

1 Introduction

Recommender systems (RSs) learn user preferences based on historical behavior, continuously providing high-quality items to users [6, 32]. These systems are widely used across various domains, such as e-commerce [31], job recommendation [4, 5], and short-video services [9], providing immense convenience to users and driving significant benefits for platforms and item providers.

In real-world scenarios, new-items are continuously introduced over time, user preferences for these items evolve dynamically, and RSs need to continuously collect user interactions for updates. However, in this dynamic recommender systems (DRSs), new-items suffer from limited exposure time, making it difficult to gather sufficient interaction data. This results in traditional recommendation models failing to effectively learn representations for them, leading to a bias toward over-recommending old-items [16, 40] and exhibiting unfairness toward new-items. Such unfairness is further amplified through the dynamic feedback loops of DRSs [35], ultimately compromising the long-term stability of the entire system.

Existing studies have recognized the importance of addressing the issue of new-item fairness [10, 43]. However, these studies fail to account for the dynamic nature of DRSs, leaving the challenge of addressing new-item fairness in DRSs unresolved. *1) First, the continuous introduction of new-items in DRSs requires optimization objectives to adapt over time.* Existing methods are primarily designed for static scenarios [10, 43], meaning they lack the flexibility to adjust to dynamic changes in item pools and user interactions. This inability to accommodate evolving conditions limits their effectiveness in achieving sustained improvements in new-item fairness. *2) Second, users' personalized preferences for new-items evolve over time.* Existing studies on item fairness neglect users' personalized preferences for new-items and also fail to account for the dynamic

evolution of these preferences in DRSs. *3) Third, users provide ongoing interaction feedback that directly influences subsequent model updates.* To maintain recommendation accuracy, it is essential to dynamically incorporate this feedback into model. Although some studies have addressed fairness issues in DRSs [18], such as mitigating popularity bias [33, 41] and ensuring user-side fairness [36], they overlook the critical issue of fairness for new-items. This aspect is essential for providing fresh user experiences and ensuring the long-term stability of DRSs. Consequently, the issue of new-item fairness in DRSs remains an open research problem that requires further exploration.

Therefore, we propose FairAgent, a reinforcement learning (RL)-based new-item fairness enhancement framework to tackle new-item fairness challenges in DRSs. Inspired by the idea of knowledge distillation (KD) [15], FairAgent inherit pre-trained embeddings from traditional recommendation models, preserving their strong ability to recommend old-items. On this basis, FairAgent specifically designs novel reward strategies tailored for recommending new-items in DRSs. To address the continuous introduction of new-items in DRSs, FairAgent incorporates a **new-item exploration reward** to consistently promote the exposure of newly introduced items, tackling the challenge of limited interaction data for new-items. To adapt to the dynamic changes in users' personalized preferences for new-items, FairAgent introduces a **fairness reward** that dynamically adjusts strategies to enhance fairness, ensuring recommendations align with users' evolving needs. Notably, this reward enables FairAgent to cater to personalized fairness requirements across different users, providing a more tailored and user-centric recommendation. Finally, to effectively leverage ongoing user interaction feedback, FairAgent introduces an **accuracy reward** designed to maintain recommendation accuracy by ensuring users are presented with items that align with their changing interests. This addresses the challenge of balancing fairness with accuracy in DRSs. By integrating these designs, FairAgent effectively addresses the key challenges of DRSs, which significantly increases new-item exposure, enhances new-item fairness by considering users' personalized preferences, and maintains high-accuracy recommendations for both new and old items. Moreover, FairAgent can integrate with any existing recommendation models to enhance new-item fairness, offering significant practical value.

In summary, our work makes the following contributions:

- We are the first to introduce the research problem of addressing new-item fairness in DRSs, emphasizing the the critical role of new-items in maintaining the stability of DRSs.
- We propose a dynamic RL-based new-item fairness enhancement framework that addresses three key challenges in DRSs: the continuous introduction of new-items, the dynamic evolution of user preferences, and the need for regular model updates. By tackling these challenges, FairAgent effectively mitigates unfairness accumulation within feedback loops of DRSs.
- We conducted extensive experiments on three public datasets and backbone models. The results demonstrate that FairAgent can effectively increase new-item exposure, enhance new-item fairness while maintaining high recommendation accuracy.

- We have released the code[1], establishing FairAgent as an open-source tool that can be integrated into any existing recommendation models.

2 Related work

New-item Fairness. Ensuring fairness for new-items without prior interaction is crucial in DRSs to enable equal opportunities for all item providers [10, 43]. The work [43] examines fairness in cold-start scenarios, formalizing it with equal opportunity and Rawlsian Max-Min fairness. It proposes a post-processing framework with two models to enhance fairness among new-items but overlooks unfairness between new and old items. This work [10] introduces a new-item exposure fairness definition considering item entry-time and presents a framework to address new-item fairness in RSs. Another approach tackles new-item fairness by addressing unfairness caused by varying interaction counts across items, with new-items being least interacted with. For instance, the inverse propensity scoring method [25] adjusted the training loss by re-weighting interactions according to the inverse of item popularity. Causal intervention methods [29, 37] aimed to mitigate the negative effects of popularity on prediction scores, while regularization-based techniques [23, 42] incorporated fairness constraints into the training loss to balance predictive scores across items. However, all of these work overlook the accumulation of new-item unfairness in dynamic feedback loops of DRSs.

Cold-start Recommender Systems. The cold-start recommendation problem aims to improve a system's ability to deliver relevant recommendations for new users or items. For new-item cold-start scenarios, existing research primarily follows two technical paradigms. The first leverages auxiliary item contents, such as category labels, textual descriptions, to reduce reliance on ID embeddings and instead utilize richer semantic features [2, 12, 28]. The second exploits graph-based structures [7, 14, 30], including user–item interaction graphs and knowledge graphs, to uncover high-order relational patterns that enhance recommendation quality for new-items. In this work, we also address the challenge of recommending new-items, with a particular focus on a novel and practical dimension—ensuring fairness in the exposure competition between new and existing items when user attention is limited. To support our investigation, we employ state-of-the-art cold-start method [12] as the backbone model.

RL-based Recommender Systems. Reinforcement learning has been extensively studied and applied in RSs, offering a powerful framework to optimize long-term user engagement and system objectives [3, 34]. The work [38] proposes a RL-based RS that optimizes strategies through continuous user interaction, effectively incorporating both positive and negative feedback. The work [39] introduces a Deep Q-Learning framework for news recommendation, modeling future rewards to address dynamic user preferences and news features. And the work [15] proposes a top-aware recommender distillation framework that uses RL to refine recommendation rankings, prioritizing top positions to improve user engagement. Inspired by the work [15], we also leverage KD to inherit traditional model's well-learned information about users and old-items. The aforementioned works confirm the effectiveness
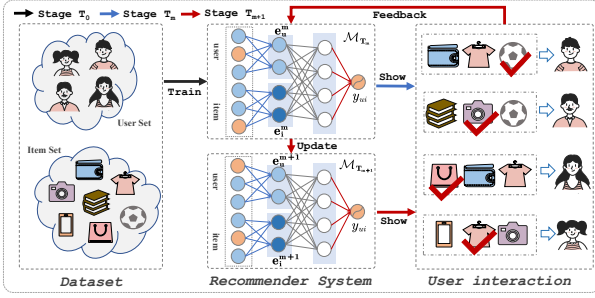
---

[1]https://github.com/Grey-z/FairAgent

Figure 1: Dynamic recommender system.

Figure 2: Proportion of items from different sets appearing in the ground-truth and recommendation sets generated by various backbone models.

of applying RL in RSs. In this work, we leverage RL techniques to enhance new-item fairness.

3 Preliminaries

As illustrated in Fig 1, we begin with a concise overview of the basic process of dynamic recommendation scenarios which involves multiple temporal stages [18, 36]. Suppose the DRS starts from T_0, where there are a set of users $\mathcal{U}$ and an item set $\mathcal{V}_{T_0}$. We collect the user-item interaction data before T_0 from all users, denoted as $\mathcal{D}_{T_0} = \{(u,v)|u \in \mathcal{U}, v \in \mathcal{V}_{T_0}\}$, to train the initial recommendation model, $\mathcal{M}_{T_0}$.

In each of the following recommendation stage T_m, a set of new-items $\mathcal{V}_{T_m}^n$ is introduced into the DRS[2]. The updated item set is expressed as:

$$\mathcal{V}_{T_m} = \mathcal{V}_{T_{m-1}} \cup \mathcal{V}_{T_m}^n, m = 1, 2, \ldots, M.$$

$\mathcal{M}_{T_m}$ takes user-item pairs (u,v), where $u \in \mathcal{U}, v \in \mathcal{V}_{T_m}$, as input to predict user preference probability $y_{uv} = \mathbf{e}_u^m \cdot \mathbf{e}_v^m$, where $\mathbf{e}_u^m$ and $\mathbf{e}_v^m$ are the embedding vectors learned by $\mathcal{M}_{T_m}$. A higher value of y_{uv} signifies a stronger preference of user u for the item v. We consider the top-K recommendation task, where the K items with the highest preference probability are included in the recommendation list, denoted as L_u, which is then presented to user u.

After receiving recommendations, users would typically interact with certain displayed items based on their preferences, such as clicking, purchasing, or adding them to a wish-list. Notably, affected by the position bias, items ranked higher on the list tend to receive more exposure, thereby is more probable to get user interactions. We define y_{uv}^* as the true preference of user u for item v, with $y_{uv}^* = 1$ indicating he/she is willing to interact with the item after observing it and $y_{uv}^* = 0$ indicating otherwise. Formally, we model user interaction behavior as follows [1, 18]:

$$\hat{y}_{uv} = \begin{cases} y_{uv}^* \cdot P_{obe}(r_{uv}|v \in L_u) & \text{if } r_{uv} \leq K \\ 0 & \text{otherwise} \end{cases}. \quad (1)$$

The observe probability $P_{obe}(r_{uv})$ that represents the probability of u observing the item v ranked at position r_{uv}, is modeled as:

$$p(r_{uv}) \sim \text{Bernoulli}\left(\frac{1}{\log_2(r_{uv}+1)}\right). \quad (2)$$

Noted that the observe probability decreases logarithmically with the value of the rank position.

At the start of the next recommendation stage T_{m+1}, user interaction data from the previous stage, $\mathcal{D}_{T_m} = \{(u,v,\hat{y}_{uv})|u \in \mathcal{U}, v \in \mathcal{V}_{T_m}\}$, is collected to update the model parameters: $\mathcal{M}_{T_m} \rightarrow \mathcal{M}_{T_{m+1}}$.

---

[2]In this work, we focus on fairness from the perspective of new-items. To avoid potential adverse effects, we temporarily exclude the introduction of new users.
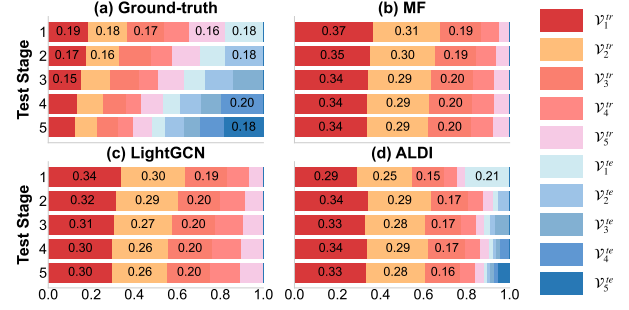
The updated model will generate the recommendation lists for users at the next recommendation stage. Notably, only items appearing in the top-K recommendation list of a user have the opportunity to receive feedback data, which consequently influences the direction of model updates in the subsequent stages. This highlights the need to ensure item (exposure) fairness in a DRS, as unfairness may accumulate over stages and eventually lead to system instability, as we will analyze in the following.

4 New-item Fairness Concern in DRSs

In this section, we perform a series of data analyses to reveal exposure fairness concerns of new-items in the existing recommendation models, such as collaborative filtering based models like Matrix Factorization (MF) [22], LightGCN [11] and cold-start based models like ALDI [12] on the Steam dataset [19]. We begin by simulating multiple recommendation stages in DRSs. We split the dataset into training and testing sets in a 1:1 ratio. The training set is then used to train the backbone model, while the testing set is divided into five subsets to be used in the five sequential stages to simulate a DRS. Items are grouped into ten sets based on their appearance time. Item sets ($\mathcal{V}_1^{tr} - \mathcal{V}_5^{tr}$) are included in the training set, while item sets ($\mathcal{V}_1^{te} - \mathcal{V}_5^{te}$) are introduced into the system at the start of their corresponding testing stage. For instance, items in $\mathcal{V}_2^{te}$ only become available when testing stage 2 begins and remains accessible in the subsequent stages. We train the backbone model follows the standard procedures outlined in the public library [26, 27]. Detailed experimental settings are provided in Section 6.

The findings are presented in Fig 2. In each subfigure, the y-axis shows the results across different recommendation stages, while the x-axis represents different item sets. Specifically, red (or orange) shades represent old-items from the training sets, while blue shades indicate new-items from each testing set. The numbers (or lengths) of the bars reflects the proportion of each item set in the ground truth (Fig 2(a)) or in the recommendation sets generated by various backbone models (Fig 2(b)-(d)).

From Fig 2(a), we observe that new-items introduced in each stage consistently attract a notable share of users in the following stages. By test stage 5, they collectively accounted for more market share (51%) than the old-items. We could expect that over time new-items may gradually replace more and more old ones in user interaction, which actually complies with lots of applications in reality such as in news/short video/e-commerce platform.

Huizhong Guo, Zhu Sun, Dongxia Wang, Tianjun Wei, Jinfeng Li, and Jie Zhang

This implies that: ***In general, users have increasing interest in exploring new-items over time.***

Comparing Fig 2(b)-(d) to Fig 2(a), we observe that in each stage, recommendations of MF, LightGCN and ALDI all deviate from what users actually prefer (i.e. proportion of items distributes differently), and that deviation gets worse along with the stages. While under ALDI, new-items can take a bit of market share (though still far from the ground-truth), there is barely chance for them under MF and LightGCN. The primary cause of this phenomenon is the lack of user interaction data for new-items. Models like MF and LightGCN struggle to capture sufficient information about new-items, resulting in a bias toward overexposing old-items with more interactions. While cold-start models like ALDI utilize additional content information to improve the representation of new-items, their exposure remains highly unfair compared to old-items. This unfairness is further amplified by the dynamic feedback loops in DRSs, ultimately leaving new-items with minimal exposure. ***This highlights a critical issue: under existing RS models, new-items are at a significant disadvantage when competing with old-items for limited exposure, failing to align with actual user needs. It underscores the importance of improving new-item exposure and fairness to enhance the stability and sustainability of dynamic recommender systems.***

5 The Framework of FairAgent

Aiming for improving the exposure fairness of new-items, in this section, we propose a RL-based new-item fairness enhancement framework, FairAgent. In this section, we first introduce the relevant fairness definitions and evaluation metrics related to fair exposure of new-items. Next, we elaborate on the design of the proposed FairAgent framework, and how it explores new-items and maintains fairness between new and old items.

5.1 User-Level New-Item Exposure Fairness

In DRSs, item providers are primarily concerned with whether their items are effectively exposed to users, as this is a prerequisite for potential user interactions. We define exposure resources that item v receives in recommendation list L_u during stage T_m as [18]:

$$Exp_m(L_u, v) = \frac{\mathbb{I}(v \in L_u)}{\log_2(r_{uv} + 1)}, \tag{3}$$

where function $\mathbb{I}(x)$ serves as an indicator, returning 1 if x is true and 0 otherwise. r_{uv} denotes the rank position of item v within L_u. Items ranked higher in the list receive more exposure and are more likely to attract user interactions.

We have observed in Section 4 that with new-items entering a system over time, proportion of user preference of old and new items changes dynamically. As shown in Fig 2, typically for new-items, those that entered more recently tend to attract more users, and for old-items, those relatively older ones attract more users. This enlightens us that to investigate whether exposure resources are fair between old and new items, their entry time should be considered. There exists an time-based item fairness metric (TGF) proposed in [10] that takes this into account. It weights items based on their entry time in measuring the exposure disparity between old and new items.

$$TGF(L_u) = \frac{1}{|\mathcal{V}_u^o|} \sum_{p=1}^{|\mathcal{V}_u^o|} w_p^o \cdot Exp_m(L_u, v_p^o) - \frac{1}{|\mathcal{V}_u^n|} \sum_{q=1}^{|\mathcal{V}_u^n|} w_q^n \cdot Exp_m(L_u, v_q^n), \tag{4}$$

$$w_p^o = |\mathcal{V}_u^o| + 1 - p, \quad w_q^n = 1 + (q - 1) \cdot \frac{|\mathcal{V}_u^o| - 1}{|\mathcal{V}_u^n| - 1}, \tag{5}$$

where $\mathcal{V}_u^o$ and $\mathcal{V}_u^n$ represent the sets of old and new items, respectively, within L_u at stage T_m. Items are ordered in descending sequence based on their entry time into the DRS. Older items in $\mathcal{V}_u^o$ with smaller indices p are assigned larger weights w_p^o considering their accumulated larger user base and entitlement to higher exposure resources. Newer items in $\mathcal{V}_u^n$ with larger indices q are assigned larger weights w_q^n to make up for their disadvantages in collecting interactions.

We want to follow the idea of TGF since it aligns with our observations, but there is a problem. Taking the average over all users, TGF cannot account for the personalized preferences for new and old items, which could vary between different users. To address this limitation, we propose a user-level personalized fairness metric:

DEFINITION 1. *User-level personalized New-item Fairness (UNF). For user $u \in \mathcal{U}$, let H_u denote his/her historical interaction list, while L_u still denotes recommendation list the user obtained. UNF is defined as the divergence between the TGF calculated by recommendation results and that of user's historical interactions.*

$$UNF = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} [TGF(L_u) - TGF(H_u)]^2. \tag{6}$$

The value of $TGF(L_u)$ reflects the exposure distribution of new and old items in the recommendation results, while $TGF(H_u)$ represents the user's historical preference for that, computed based on their past interactions.

A smaller value of UNF indicates that the recommendation results align more closely with the user's preference for new and old items, making the system *fairer at the individual user level*.

5.2 Fairness Enhancement with RL Framework

In this section we present the design details of our RL-based new-item fairness enhancement framework, FairAgent. As shown in Fig 3, FairAgent initializes from a backbone model, selects the candidate items based on user interaction and a carefully designed reward mechanism, and then produces a fairer top-K recommendation list. A core characteristic of FairAgent lies in its design of the reward mechanism, which offers three significant advantages, 1) appropriate new-item recommendation rate and 2) fairer exposure of new-items and 3) high recommendation accuracy. The implementation details will be elaborated in the following sections.

5.2.1 Setting of RL-based DRS. Following the paradigm of RL-based work [15, 39], we utilize the following settings in FairAgent:

- **State** s_u^t refers to a vector that captures the historical preferences of a specific user u at step t. It comprises of the embeddings of the users along with most recent N items they have interacted with, represented as $s_u^t = [e_u, e_{v_1}, e_{v_2}, \cdots, e_{v_N}]$.
- **Action** a_u^t refers to the item selected for user u at step t from the action space $\mathcal{A}_u^t$.
- **Reward** r_u^t represents the benefit obtained by selecting action a_u^t given the state s_u^t.
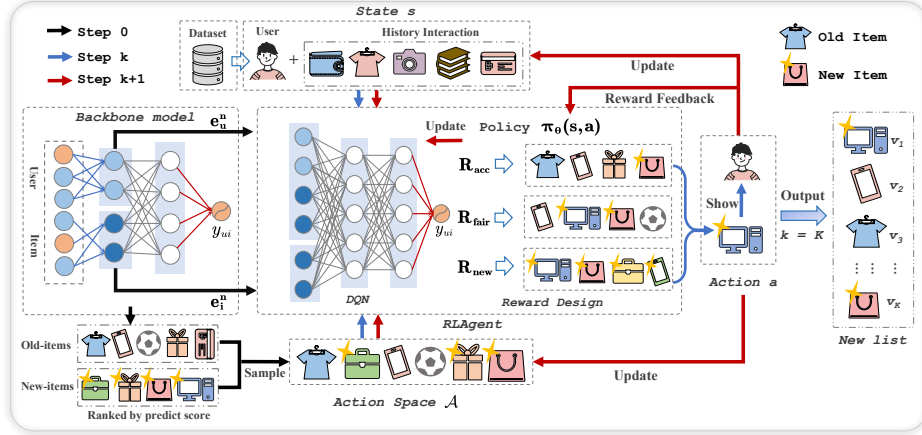
Figure 3: Our proposed RL-based new-item fairness enhancement framework, FAIRAGENT.

Deep Q-Network (DQN) [17] is a RL algorithm that integrates Q-learning with deep neural networks, enabling it to approximate the Q-value function for high-dimensional state-action spaces. In DRSs, DQN learns the optimal policy $\pi_\theta(s_u^t, a_u^t)$ for recommendations by modeling user-item interactions as a Markov Decision Process (MDP). The Q-value, $Q(s_u^t, a_u^t)$, represents the expected cumulative reward of selecting an item v (action a_u^t) given the user's preferences (state s_u^t), guiding the system to recommend items that maximize user satisfaction or system objectives.

5.2.2 Detailed Step to Construct FAIRAGENT. Based on the aforementioned settings, we now provide a detailed explanation of the steps involved to construct FAIRAGENT, as illustrated in Algorithm 1.

(1) **Initialize Model** (lines 1-3): FAIRAGENT employs DQN, denoted as Q_θ, as its core component to predict user preferences for items. To accelerate convergence, DQN is initialized with pretrained embeddings $e_{\mathcal{U}}$ and $e_{\mathcal{V}_{T_0}}$ from the backbone model $\mathcal{M}_{T_0}$. This enables DQN to retain the backbone model's strong recommendation capability for old-items and build upon it to enhance recommendations for new-items.

(2) **Construct initial state and action space** (lines 5-11): We combine the embeddings of the user and the N items he/she most recently interacted with during the previous recommendation stage to construct the initial state, s_u^0. For the construction of action space $\mathcal{A}_u^0$, we introduced a preference-aware sampling strategy to dynamically adjust the ratio of old and new items. Specifically, we define a *Bernoulli* distribution over the binary variable "item type" (old or new), parameterized by the ratio of new-items in user's historical interactions p_{new}. Each item added to the action space is sampled from this distribution: with probability p_{new} it is drawn from the set of new-items, and with probability $(1 - p_{new})$ it is drawn from the set of old-items. For instance, if the estimated probability of the user preferring a new-item is $P_{new} = 0.4$, the distribution is configured to yield an old-to-new item ratio of 6:4. For the selection within the new and old item sets, we adopt the same greedy search strategy: select the item with the highest score remaining in the item set, where the score is calculated from the pre-trained backbone models. By repeating this process for K steps, we construct a fixed-length initial action space $\mathcal{A}_u^0$.

Algorithm 1: Training Process of FAIRAGENT at stage T_m

Input: $\mathcal{D}_{T_m}, \mathcal{M}_{T_m}, \mathcal{V}_{T_m}, \mathcal{U}, H_{\mathcal{U}}$
Output: Updated model $Q_\theta, Q_{\theta'}, L_u$

1 **if** $m == 0$ **then**
2    $e_{\mathcal{U}}, e_{\mathcal{V}_{T_0}} \leftarrow \mathcal{M}_{T_0}$; // Get pre-trained embeddings
3    $Q_\theta, Q_{\theta'} = \text{Initialize}(e_{\mathcal{U}}, e_{\mathcal{V}_{T_0}})$;
4 **for each** $u \in \mathcal{U}$ **do**
5    $s_u^0 = [e_u, e_{v_1}, e_{v_2}, \cdots, e_{v_N}]$;
6    $P_{new} = \text{GetPreference}(u, H_u)$;
7   **while** $len(\mathcal{A}_u^0) < N_{act}$ **do**
8   **if** $\mathbb{I}(x = 1 | x \sim Bernoulli(P_{new}))$ **then**
9          $\mathcal{A}_u^0 \leftarrow \text{Sample}(\mathcal{V}_{T_m}^n)$; // Sample a new-item.
10   **else**
11          $\mathcal{A}_u^0 \leftarrow \text{Sample}(\mathcal{V}_{T_{m-1}})$; // Sample an old-item.
12    $L_u^0 \leftarrow \varnothing$;
13   **for** $t \leftarrow 1$ *to* K **do**
14   **if** train == True **and** $\mathbb{I}(x = 1 | x \sim Bernoulli(\varepsilon))$ **then**
15          $a_u^t = \text{Random}(\mathcal{A}_u^t)$; // Choose action randomly
16   **else**
17          $a_u^t = \text{ChooseAction}(s_u^t, \mathcal{A}_u^t, Q_\theta)$;
18       $L_u^t \leftarrow L_u^{t-1} \cup a_u^t(v)$;
19       $r_u^t = \text{GetReward}(s_u^t, a_u^t, \mathcal{V}_{T_m}^n)$;
20       $\mathcal{A}_u^t = \text{Update}(\mathcal{V}_{T_m})$;
21   **if** a_u^t returns positive feedback **then**
22          $s_u^{t+1} = \text{UpdateState}(a_u^t, s_u^t)$; // Update state
23   **else**
24          $s_u^{t+1} = s_u^t$;
25       $\mathcal{D}_{buffer} \leftarrow (s_u^t, a_u^t, r_u^t, s_u^{t+1})$; // Memory mechanism
26   **if** $len(\mathcal{D}_{buffer}) > N_{mem}$ **then**
27          $Q_{\theta'} = \text{Train}(\mathcal{D}_{buffer})$; // Update parameters
28       $Q_{\theta'} = \text{UpdateNetwork}(Q_\theta)$;
        // Update target network every five iterations
29    $L_u \leftarrow L_u^K$;
30 **return** $Q_\theta, Q_{\theta'}, L_u$;

---

(3) **Choose action** (lines 14-17): At each step t, FAIRAGENT takes an action a_u^t (i.e., selects an item v to user u) from action space $\mathcal{A}_u^t$ based on the current user state s_u^t. During training, we use a strategy combined with ε-greedy exploration. Specifically, with a probability of $1 - \varepsilon$, the model selects the item with the highest Q-value from the action space, while with a probability of ε, it randomly selects an item to encourage exploration. This approach balances exploitation and exploration, allowing the model to avoid local optima and improve its generalization capabilities.

(4) **Calculate reward** (lines 18-19): After taking an action, the selected item v is added to the recommendation list L_u. The reward is subsequently calculated based on the updated L_u^t at step t.

(5) **Update action space and state** (lines 20-24): After step t completed, both the action space $\mathcal{A}_u^t$ and the state $\mathbf{s}_u^t$ need to be updated for the next step $t + 1$. For the action space, the selected item v is removed from $\mathcal{A}_u^t$, and another item is sampled and added into $\mathcal{A}_u^{t+1}$, ensuring the action space size remains constant. If the selected item v receives user's positive feedback, i.e., $\hat{y}_{uv} = 1$, the state is updated in a similar manner. The earliest interacted item in the $\mathbf{s}_u^t$ is removed, and $\mathbf{e}_v$ is added to the $\mathbf{s}_u^{t+1}$. If no positive feedback is received, the state $\mathbf{s}_u^{t+1}$ remains the same as $\mathbf{s}_u^t$.

(6) **Update DQN** (lines 25-28) Inspired by the existing works [15, 20, 24], we utilize two key techniques to improve stability and performance of DQN, including: 1) *Experience Replay*. A buffer is used to store past experiences $(\mathbf{s}_u^t, \mathbf{a}_u^t, r_u^t, \mathbf{s}_u^{t+1})$, which are sampled randomly during training to break the correlation between consecutive updates. 2) *Target Network*. A separate target network $Q_{\theta'}$ is maintained and periodically updated to stabilize the Q-value estimates, preventing rapid oscillations during training. In FairAgent, DQN is adapted to model user decision-making processes and optimize the ranking of items based on a reward mechanism, ensuring accurate and fair recommendations. Specifically, the parameters θ of DQN are trained using the following loss function:

$$\mathcal{L}_\theta = \mathbb{E}_{\mathbf{s}_u^t, \mathbf{a}_u^t, r_u^t, \mathbf{s}_u^{t+1}} \left[\left(Q_{\text{target}} - Q\left(\mathbf{s}_u^t, \mathbf{a}_u^t; \theta\right) \right)^2 \right], \tag{7}$$

where $Q(\mathbf{s}_u^t, \mathbf{s}_u^t)$ represents the Q-value for the current state-action pair, Q_{target} denotes the target Q-value. For each training step, the target Q-value is defined as:

$$Q_{\text{target}} = \mathbb{E}_{\mathbf{s}_u^{t+1}} \left[r_u^t + \lambda \max_{\mathbf{a}_u^{t+1}} Q(\mathbf{s}_u^{t+1}, \mathbf{a}_u^{t+1}; \theta') \mid \mathbf{s}_u^t, \mathbf{a}_u^t \right], \tag{8}$$

where θ' indicates the parameters of the target network, which are periodically updated from the main Q-network, such as in every 5 iterations. The constant λ, ranging between 0 and 1, determines the balance between current and future rewards. FairAgent updates the recommendation policy $\pi_\theta(\mathbf{s}_u^t, \mathbf{a}_u^t)$ to find out the optimal policy parameter θ that can maximize the expected cumulative rewards.

5.2.3 Reward Mechanism. As previously described, FairAgent employs a reward mechanism to generate the recommendation list for all users. The objective is to produce a refined list that satisfies the following three key properties: 1) *Appropriate new-item recommendation rate.* The continuously introduced new-items receive sufficient exposure to meet users' needs. 2) *Fair exposure allocation for new and old items.* Exposure resources are distributed fairly between new and old items, aligning with users' preferences for new-items. 3) *High accuracy.* The recommended items closely reflect users' true preferences for both new and old items.

The three rewards below are designed to guide model update to achieve the above properties respectively.

New-item exploration reward (R_{new}) aims to encourage the exploration of continuously introduced new-items, thus improving the exposure rate of new-items and ensuring sustainability of the whole DRS.

$$R_{new} = \gamma \cdot \mathbb{I}(v \in \mathcal{V}_{T_m}^n) + (1 - \gamma) \cdot \mathbb{I}(v \in \mathcal{V}_{T_m}^n) \cdot \mathbb{I}(\hat{y}_{uv} = 1) \tag{9}$$

We use parameter γ to control the distribution of reward assigned to recommending new-items that have received user positive feedback and that have not.

Fairness reward (R_{fair}) aims to adjust the distribution of new and old items in the recommendation list to align with users' evolving preferences for them, thereby achieving personalized new-item fairness. Let

$$UNF_u^t = \left| TGF(L_u^t) - TGF(H_u) \right|,$$
$$R_{fair} = \frac{2 \cdot \tanh(UNF_u^t - UNF_u^{t+1})}{1 + \tanh(2)}, \tag{10}$$

where L_u^t denotes the generated recommendation list at step t, while H_u represents the user's historical interaction list, which reflects their historical preferences for new-items. The fairness reward R_{fair} ranges between $(-1, 1)$, taking a positive value when $TGF(L_u)$ aligns with $TGF(H_u)$ and a negative value otherwise. This reward is designed to optimize the recommendation list to better align with the user's true preferences for new-items.

Accuracy reward (R_{acc}) leverages ongoing user interaction feedback, dynamically adjusting to ensure the system adapts to evolving preferences and effectively utilizes this feedback to enhance recommendation accuracy over time.

$$R_{acc} = \frac{\mathbb{I}(\hat{y}_{uv} = 1)}{\log_2(r_{uv} + 1)} \tag{11}$$

where $\hat{y}_{uv} = 1$ denotes a positive user feedback to item v.

Taking accuracy as the basic goal, new-item fairness and exploration as the additional ones, we define the total reward as:

$$R_{total} = R_{acc} + \alpha R_{fair} + \beta R_{new}, \tag{12}$$

where α and β regulate the influence of R_{fair} and R_{new} respectively.

6 Experiments

In this section, we conduct extensive experiments on three publicly available datasets and backbone models. By addressing the following three research questions consecutively, we demonstrate the superior performance of FairAgent in improving exposure rate of new-items, achieving new-item fairness while aligning with user's personalized preference for new-items and maintaining high recommendation accuracy.

RQ1: Compared to state-of-the-art (SOTA) baselines, can FairAgent more effectively improve exposure rate of new-items, address unfairness between new and old items, while maintaining high recommendation accuracy?

RQ2: Can FairAgent keep up with user's personalized dynamic preferences for old and new items in DRSs?

RQ3: How effective are the different reward components in the design of FairAgent?

Datasets. To be align with real-world situations, we selected three publicly available datasets to construct varying DRS scenarios. Each dataset reflects distinct user behavior patterns. **KuaiRec-Small** [8] is a dense dataset collected from the Kuaishou platform, containing user interactions with short videos. In each stage, numerous new-items enter the DRS, and user interest shift towards these items rapidly. **KuaiRec-Large** [8] is a larger version of the previous dataset. This dataset features a DRS with relatively smoother expansion of item set and slower shifts of user interests. For these two datasets, we filter interactions where users' watch ratio is greater
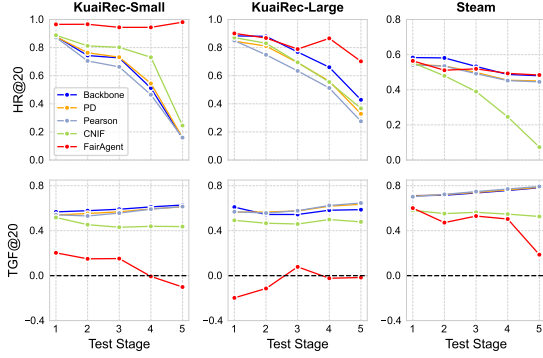
Figure 4: Dynamic changes in HR and TGF metrics across 5 test stages for different methods using the MF backbone.

Figure 5: Dynamic changes in HR and TGF metrics across 5 test stages for different methods using the ALDI backbone.

than 1.0 as positive samples. **Steam** [19] is a game dataset containing user interaction data from the Steam gaming platform. It encompasses a larger pool of users and items, with relatively stable item set expansion and more consistent changes in user interests. To create a DRS setting, we split all interaction data into training and test sets at a 1:1 ratio in chronological order. We train the backbone model on the entire training set, and utilize the last 20% to train FAIRAGENT and all the baselines. The test set is then divided into 5 sets to construct different test stages, each introducing a set of new-items to reflect dynamic introduction of new-items. For all datasets, we filter out users with fewer than 10 interactions.

Backbone Model. Following the settings of the existing works [10, 23], we employ widely used models, including Matrix Factorization (MF) [22], the graph-based recommendation algorithm LightGCN [11], and the cold-start recommendation algorithm ALDI [12], as backbone models to validate the effectiveness of FAIRAGENT. Noted that both MF and LightGCN rely solely on user-item interaction data, while ALDI requires content information to recommend new-items. For fair comparison, we utilize additional content information only when ALDI was used as the backbone model for all the baseline methods. We constructed the content information using a pre-trained language model [21], extracting features from the titles and categories of short videos (for KuaiRec-Small and KuaiRec-Large), as well as the names and labels of games (for Steam).

Comparison Baselines. To the best of our knowledge, this is the first work to address new-item fairness in DRSs. We opted to adapt methods designed to enhance item fairness in static scenarios as baselines for comparison. **PD** [37] employs causal intervention to adjust the final prediction scores based on the amount of each item's interactions, thereby balancing the overall recommendation rate. **Pearson** [42] introduces a regularization term that incorporates the correlation between prediction scores and item popularity to enhance item fairness during training. These baselines primarily focus on increasing the recommendation exposure of items with limited interactions, where here new-items are considered as those with no interaction. **CNIF** [10] is the first work that explicitly considers the time at which an item enters a system, and introduces a time-based fairness loss function to optimize new-item fairness during training. The parameters of these baselines were tuned based on the configurations in the existing work [10].

Implementation Details. We follow the training protocol from the Open-Source Library [26] to train all backbone models with
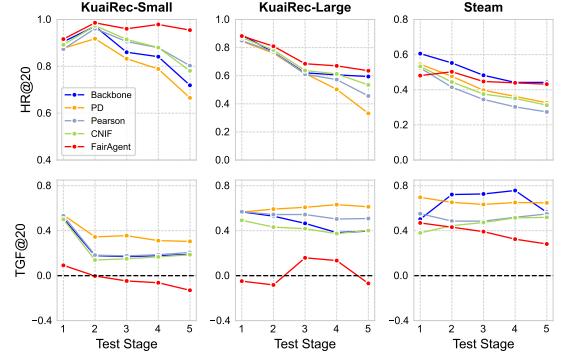
BPR loss. Then we employ Bayesian optimization to tune the relevant hyper-parameters, including learning rate, regularization coefficient, and embedding dimension. For fair comparison, all experiments use a batch size of 8, 192 and a negative sampling rate of 4 for training. For the parameters mentioned in Section 5.2.3, we tune α within [0.5, 2.5], β within [0, 1], and set $\gamma = 0.1$.

Evaluation and Metrics. Following the setting of [15, 26], at each test stage, we sample negative items for users to construct a fixed-length candidate set of 1000 items for evaluation. To measure recommendation accuracy, we adopt Hit Rate (HR) [26, 34] and NDCG [5, 13], with larger values indicating better performance. We also report TGF (introduced in [10]) at all stages, to evaluate overall new-item fairness (considering all the users), with larger values denoting more unfair exposure distribution against new-items. Furthermore, we use New-item Coverage (NC) to represent the ratio of new-items in recommendation lists for all users and a trade-off metric δT to quantify the balance between fairness improvement and any potential loss in recommendation accuracy following the work [10]. A higher δT indicates more fairness enhancement and less accuracy loss. Due to page limitations, we report results for all the metrics with $K = 20$. Note that similar conclusions can be drawn for other K values.

6.1 Experimental Results and Analysis

6.1.1 Results of RQ1. We present how recommendation accuracy and new-item fairness of different methods change along with five stages, across different datasets in a dynamic recommendation scenario. Fig 4 and Fig 5 present the results with MF and ALDI as the backbone model respectively[3], with sub-figures in a column representing the results under a dataset (3 in total in each figure). In both the figures, the first row represents recommendation accuracy (*HR*, the higher the better), while the second row represents the degree of unfairness on new-item exposure (*TGF*, the closer to 0 the better). We also present some quantitative results obtained in the same experiments in Tab 1, where each value represents the average performance of the corresponding method across the five test stages, providing a measure of its overall effectiveness in DRS settings. Bold values indicate the highest in each column, while underlined values represent the second highest. We make the following observations.

---

[3]The trend for LightGCN is similar to MF and is omitted here due to space constraints.

Table 1: Average results across all test stages for different methods on three backbone models.

Method	MF						LightGCN						ALDI					
K=20	HR↑	NDCG↑	TGF↓	UNF↓	NC↑	%δT↑	HR↑	NDCG↑	TGF↓	UNF↓	NC↑	%δT↑	HR↑	NDCG↑	TGF↓	UNF↓	NC↑	%δT↑
									KuaiRec-Small									
Backbone	0.6035	0.2741	0.5949	0.1252	0.0000	-	0.6379	0.2972	0.5694	0.1156	0.0000	-	0.8593	0.4269	0.2448	0.0385	0.0183	-
PD	0.6154	0.2781	0.5729	0.1195	0.0000	102.9%	0.4749	0.2142	0.6717	0.1425	0.0000	80.8%	0.8163	0.4040	0.3711	0.0691	0.0000	72.9%
Pearson	0.5729	0.2584	0.5667	0.1170	0.0000	99.8%	0.4736	0.2132	0.6741	0.1432	0.0000	80.5%	0.8845	0.4558	0.2564	0.0420	0.0036	99.1%
CNIF	0.6957	0.3211	0.4552	0.0903	0.0000	120.9%	0.4770	0.2302	0.6458	0.1419	0.0000	82.9%	0.8883	0.4538	0.2286	0.0357	0.0054	105.1%
FairAgent	0.9600	0.5082	0.0793	0.0061	0.3003	197.5%	0.9413	0.5025	0.0866	0.0124	0.3306	182.7%	0.9591	0.5540	-0.0304	0.0096	0.3097	160.3%
									KuaiRec-Large									
Backbone	0.7241	0.4021	0.5732	0.1033	0.0000	-	0.7120	0.3900	0.5834	0.1020	0.0000	-	0.6956	0.3382	0.4679	0.0762	0.0062	-
PD	0.6478	0.3388	0.5917	0.1062	0.0000	93.5%	0.5648	0.2644	0.7003	0.1241	0.0000	81.6%	0.6114	0.2961	0.6017	0.0994	0.0000	80.9%
Pearson	0.6041	0.3079	0.5945	0.1054	0.0000	90.7%	0.5642	0.2630	0.7043	0.1249	0.0000	81.3%	0.6515	0.3129	0.5334	0.0812	0.0000	90.2%
CNIF	0.6638	0.3477	0.4789	0.0836	0.0000	103.9%	0.5588	0.2890	0.6480	0.1231	0.0000	85.3%	0.6877	0.3408	0.4232	0.0692	0.0000	104.2%
FairAgent	0.8252	0.4280	-0.0547	0.0224	0.3440	161.1%	0.8445	0.4609	-0.0377	0.0251	0.3168	163.9%	0.7365	0.3530	0.0184	0.0277	0.2708	149.0%
									Steam									
Backbone	0.5328	0.2882	0.7401	0.0669	0.0000	-	0.5258	0.2835	0.7278	0.0621	0.0000	-	0.5049	0.2518	0.6540	0.0454	0.0788	-
PD	0.4966	0.2441	0.7431	0.0652	0.0000	96.5%	0.4884	0.2511	0.7666	0.0673	0.0000	94.0%	0.3719	0.1914	0.6334	0.0400	0.0009	93.2%
Pearson	0.4929	0.2385	0.7471	0.0657	0.0000	95.9%	0.4905	0.2528	0.7686	0.0676	0.0000	94.0%	0.3719	0.1545	0.5178	0.0339	0.0374	97.6%
CNIF	0.3485	0.1713	0.5537	0.0453	0.0000	96.1%	0.5137	0.2762	0.7095	0.0608	0.0000	100.1%	0.4031	0.2071	0.4658	0.0453	0.0528	103.9%
FairAgent	0.5143	0.2674	0.4584	0.0325	0.1348	116.8%	0.4458	0.2129	0.3905	0.0370	0.1597	114.1%	0.4602	0.2151	0.3794	0.0321	0.1391	115.6%

First, FairAgent can be effectively applied to different types of recommendation backbone models to increase exposure rate of new-items. From the analysis of NC (new-item coverage) metric (Tab 1), we found that with all backbones, the existing methods fail to improve the exposure rate of new-items (NC persistently remains near 0). These results highlight their inability to handle the continuous introduction of new-items in DRSs. In contrast, FairAgent consistently achieves significant improvements in NC across all the backbone models and datasets (with NC approaching 30% on KuaiRec-Small and KuaiRec-Large, and exceeding 13% on Steam). This demonstrates its robust ability to enhance new-item exposure and effectively tackle the challenge of continuously introducing new-items in DRSs.

Second, FairAgent is the most effective in improving the exposure fairness of new-items against old-items. It almost always achieves the lowest values of TGF under different backbone models and datasets, while the other baselines typically show much higher TGF values (more serious unfairness), as observed from Fig 4, Fig 5 and Tab 1. A TGF value closer to 0 indicates smaller exposure disparities between new and old items, reflecting greater fairness in the DRS. For KuaiRec-Small and KuaiRec-Large, FairAgent effectively captures users' strong preferences for new-items, significantly improving fairness. Specifically, its TGF values reaches close to 0, and decrease by an average of 86.34% and 93.35% compared to all the backbone models on the two datasets, respectively. In contrast, the best baseline, CNIF, only achieves an decremental of 5.56% and 4.98%. For Steam, FairAgent adopts a more balanced fairness optimization strategy to aligning with users' weaker preferences for new-items. Under this condition, TGF still improves by an average of 42.13% compared to the backbone models, outperforming CNIF's 18.83% improvement.

Third, FairAgent maintains the highest recommendation accuracy in most of the cases, highlighting its ability in well balancing new-item fairness with accuracy in DRSs. Specifically, for KuaiRec-Small, FairAgent maintains the highest recommendation accuracy across all the test stages, accurately recommending preferred new-items for users in dynamic recommendation. Compared to all the backbone models, the HR and NDCG metrics got improved by an average of 39.41% and 61.42%, respectively. In contrast, to achieve fairness, the best baseline CNIF sacrifices 2.19% in HR. For KuaiRec-Large, a similar pattern can be observed. Compared to the backbone models, FairAgent achieves an improvement of 12.82% in HR and 9.66% in NDCG on average, while the best baseline suffers the losses of 10.33% and 12.88%, respectively. In Steam-like scenarios, where users have weaker interest in new-items, improving fairness often reduces recommendation accuracy by limiting old-item exposure. This especially requires balancing well fairness and accuracy, as neither severe unfairness nor low accuracy suits practical needs. We use δT metric to evaluate how well different methods balances these two. The results shows that *FairAgent achieves the best balance in all the scenarios, enhancing fairness with minimal loss in accuracy, thus maintaining long-term stability of a system.*

The analysis of the dataset and backbone model reveals several key insights. First, datasets where new-items and user preference evolve more rapidly seem to be more challenging. For instance, the accuracy and fairness of the backbone models decrease relatively more sharply with KuaiRec-Small, where user interest tend to quickly shift towards the numerous new-items entering overtime. This confirms our concern raised in Section 4 that if unfairness is not addressed promptly, it may quickly accumulate in the dynamic feedback of DRSs, leading to reduced exposure of new-items and a sharp decline in recommendation accuracy. And for the fairness enhancement methods, it seems more difficult to keep up with the dynamics in KuaiRec-Small. For KuaiRec-Large and especially Steam, where the expansion of new-item set and also shift of user interest happen more slowly, the backbone models perform slightly better and more stably. The same goes for the baseline fairness methods. Second, despite being a cold-start model, ALDI does not consistently enhance the new-item exposure and fairness as expected. Third, TGF of FairAgent has negative values sometimes, meaning the reversal of exposure resources. This improves the chances of users encountering new-items. And its degree can be controlled to near zero through the use of adaptable parameters in the reward function.

In summary, we answer RQ1: *FairAgent most effectively improves new-item exposure and enhances the fairness between new and old items, while maintaining high recommendation accuracy in DRSs.*
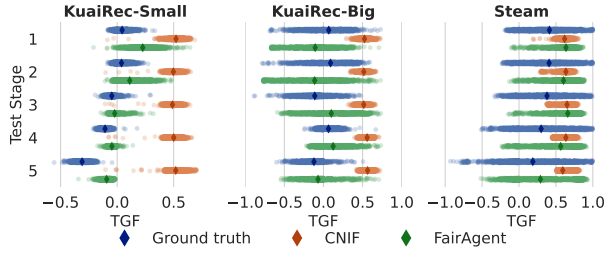
Figure 6: Performance of FairAgent and the best baseline in adapting to users' true preferences for new-items across different recommendation stages.

6.1.2 Results of RQ2. Tab 1 presents the average UNF (user-level new-item fairness) values across the five test stages, with smaller values indicating that the recommendations better adapting to the dynamic changes in users' personalized preferences for new-items. The results show that FairAgent consistently achieves the best UNF in all the scenarios. To illustrate this more intuitively, we visualize the phenomenon in Fig 6. Each point in the blue area represents a user's true preference for new-items at a specific recommendation stage, with the value calculated using the TGF metric based on ground-truth interactions. The orange and green areas represent the recommendation results of CNIF[4], and FairAgent, respectively, with values calculated using the TGF metric based on their respective recommendation lists. When the distribution along the x-axis closely matches the blue region (with bars of equal length) in the same test stage, it signifies that the recommendation results better align with the user's dynamic fairness requirements.

From the results on KuaiRec-Small, we observe that most users' preferences shift with the introduction of new-items (TGF shifts from near zero to negative), and FairAgent effectively tracks these changes (green areas closely follow the movement of blue areas). This shows that FairAgent captures the dynamic evolution of users' personalized preferences for new-items and adjusts its fairness strategies accordingly. In contrast, CNIF fails to do so, leading to significant mismatches between recommendations and users' actual preferences. On KuaiRec-Large, users show diverse preferences for new-items, with some favoring them (TGF < 0) and others preferring older ones (TGF > 0). FairAgent adapts to this diversity by tailoring fairness strategies to individual users, demonstrating its ability to achieve new-item fairness at the user level in DRSs. In the Steam scenario, most users prefer older items (TGF > 0), and FairAgent successfully captures this. It adopts a milder fairness strategy that not only improves new-item fairness but also respects users' preferences for older items, striking an effective balance between accuracy and fairness.

In summary, we answer RQ2: ***FairAgent can effectively adapt to users' personalized dynamic changes in preferences for old and new items within DRSs by flexibly tailoring its optimization strategies.***

6.1.3 Results on RQ3. To analyze the roles of different reward components, we conducted ablation studies. As similar conclusions were drawn across three backbone models, we present only the MF

---

[4]CNIF is the best performed baseline for improving new-item fairness, so we choose it for comparison. Since the conclusions are similar across different backbone models, we present the results for MF only due to space constraints.

Table 2: Results of ablation study on three datasets.

Method	Dataset - KuaiRec-Small				
	HR↑	NDCG↑	TGF↓	UNF↓	NC↑
FairAgent	**0.9600**	**0.5082**	**0.0793**	**0.0061**	**0.3003**
FairAgent$_{w/o\ f}$	0.9340	0.4617	0.1985	0.0576	0.1945
FairAgent$_{w/o\ n}$	0.9315	0.4605	0.1824	0.0370	0.1699
FairAgent$_{w/o\ f\&n}$	0.7946	0.3461	0.4649	0.0410	0.0450
	Dataset - KuaiRec-Large				
FairAgent	0.8445	0.4609	-0.0377	0.0251	0.3168
FairAgent$_{w/o\ f}$	**0.8686**	**0.4897**	0.0786	0.0312	**0.3562**
FairAgent$_{w/o\ n}$	0.8311	0.4537	-0.0685	0.0270	0.2704
FairAgent$_{w/o\ f\&n}$	0.7574	0.4052	0.3161	0.0513	0.1343
	Dataset - Steam				
FairAgent	0.4477	0.2369	0.4438	**0.0250**	0.1733
FairAgent$_{w/o\ f}$	0.4055	0.1768	0.4584	0.0396	**0.1918**
FairAgent$_{w/o\ n}$	**0.5143**	**0.2674**	**0.4051**	0.0325	0.1348
FairAgent$_{w/o\ f\&n}$	0.5028	0.2593	0.5401	0.0425	0.0938

results in Tab 2. Specifically, FairAgent$_{w/o\ f}$ removes the fairness reward R_{fair} (let $\alpha = 0$), FairAgent$_{w/o\ n}$ removes the new-item exploration reward R_{new} (let $\beta = 0$), and FairAgent$_{w/o\ f\&n}$ removes both (let $\alpha = 0, \beta = 0$). We observe that removing R_{fair} leads to poorer fairness performance, as indicated by increased |TGF| and UNF values. This validates the effectiveness of the fairness reward in improving new-item fairness in DRSs by considering users' personalized preferences for new-items. Similarly, removing R_{new} results in a significant drop in new-item recommendation rates, as reflected by the reduced NC values. This demonstrates the effectiveness of R_{new} in enhancing the exposure of dynamically introduced new-items. In scenarios where users prefer new-items (e.g., KuaiRec-Small), R_{fair} and R_{new} can work synergistically to both ensure new-item fairness and significantly enhance recommendation accuracy.

In summary, we answer RQ3: ***Our designed fairness reward and new-item exploration reward effectively enhance new-item fairness and increase new-item recommendation rates.***

7 Conclusion and Discussion

In this work, we proposed FairAgent, a RL-based new-item fairness enhancement framework, designed to effectively boost new-item exposure, keep up with users' personalized preferences for new-items, and meanwhile maintain high recommendation accuracy within DRSs.

While FairAgent shows promising performance, several directions remain for future improvement. First, its reliance on a backbone model may constrain performance due to the model's inherent limitations, whereas RL-based approaches often suffer from slow convergence and poor scalability. Enhancing computational efficiency would also improve applicability in large-scale settings. Finally, extending fairness considerations to other stakeholders, such as content creators and platform providers, could lead to more comprehensive and equitable DRSs.

8 Acknowledgment

References

[1] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In *Proceedings of the twelfth ACM International Conference on Web Search and Data Mining*. 474–482.

[2] Hao Chen, Zefan Wang, Feiran Huang, Xiao Huang, Yue Xu, Yishi Lin, Peng He, and Zhoujun Li. 2022. Generative adversarial framework for cold-start item recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2565–2571.

[3] Xiaocong Chen, Siyu Wang, Julian McAuley, Dietmar Jannach, and Lina Yao. 2024. On the opportunities and challenges of offline reinforcement learning for recommender systems. *ACM Transactions on Information Systems* 42, 6 (2024), 1–26.

[4] Yingpeng Du, Hongzhi Liu, Hengshu Zhu, Yang Song, Zhi Zheng, and Zhonghai Wu. 2025. Quasi-Metric Learning for Bilateral Person-Job Fit. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).

[5] Yingpeng Du, Di Luo, Rui Yan, Xiaopei Wang, Hongzhi Liu, Hengshu Zhu, Yang Song, and Jie Zhang. 2024. Enhancing job recommendation through llm-based generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 8363–8371.

[6] Yingpeng Du, Ziyan Wang, Zhu Sun, Yining Ma, Hongzhi Liu, and Jie Zhang. 2024. Disentangled Multi-interest Representation Learning for Sequential Recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 677–688.

[7] Yuntao Du, Xinjun Zhu, Lu Chen, Ziquan Fang, and Yunjun Gao. 2022. Metakg: Meta-learning on knowledge graph for cold-start recommendation. *IEEE Transactions on knowledge and data engineering* 35, 10 (2022), 9850–9863.

[8] Chongming Gao, Shijun Li, Wenqiang Lei, Jiawei Chen, Biao Li, Peng Jiang, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. 2022. KuaiRec: A Fully-Observed Dataset and Insights for Evaluating Recommender Systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. 540–550. https://doi.org/10.1145/3511808.3557220

[9] Shiping Ge, Qiang Chen, Zhiwei Jiang, Yafeng Yin, Ziyao Chen, and Qing Gu. 2024. Short Video Ordering via Position Decoding and Successor Prediction. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2167–2176.

[10] Huizhong Guo, Dongxia Wang, Zhu Sun, Haonan Zhang, Jinfeng Li, and Jie Zhang. 2024. Configurable Fairness for New Item Recommendation Considering Entry Time of Items. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 437–447.

[11] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 639–648.

[12] Feiran Huang, Zefan Wang, Xiao Huang, Yufeng Qian, Zhetao Li, and Hao Chen. 2023. Aligning Distillation For Cold-Start Item Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1147–1157. https://doi.org/10.1145/3539618.3591732

[13] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[14] Jinri Kim, Eungi Kim, Kwangeun Yeo, Yujin Jeon, Chanwoo Kim, Sewon Lee, and Joonseok Lee. 2024. Content-based graph reconstruction for cold-start item recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1263–1273.

[15] Hongyang Liu, Zhu Sun, Xinghua Qu, and Fuyong Yuan. 2021. Top-aware recommender distillation with deep reinforcement learning. *Information Sciences* 576 (2021), 642–657.

[16] Zhongzhou Liu, Yuan Fang, and Min Wu. 2023. Mitigating popularity bias for users and items with fairness-centric adaptive recommendation. *ACM Transactions on Information Systems* 41, 3 (2023), 1–27.

[17] Volodymyr Mnih. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[18] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling fairness and bias in dynamic learning-to-rank. In *Proceedings of the 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval*. 429–438.

[19] Mark O'Neill, Elham Vaziripour, Justin Wu, and Daniel Zappala. 2016. Condensing steam: Distilling the diversity of gamer behavior. In *Proceedings of the 2016 Internet Measurement Conference*. 81–95.

[20] Joseph O'Neill, Barty Pleydell-Bouverie, David Dupret, and Jozsef Csicsvari. 2010. Play it again: reactivation of waking experience and memory. *Trends in Neurosciences* 33, 5 (2010), 220–229.

[21] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084

[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[23] Wondo Rhee, Sung Min Cho, and Bongwon Suh. 2022. Countering Popularity Bias by Regularizing Score Differences. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 145–155.

[24] Tom Schaul. 2015. Prioritized Experience Replay. *arXiv preprint arXiv:1511.05952* (2015).

[25] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *International Conference on Machine Learning*. PMLR, 1670–1679.

[26] Zhu Sun, Hui Fang, Jie Yang, Xinghua Qu, Hongyang Liu, Di Yu, Yew-Soon Ong, and Jie Zhang. 2022. DaisyRec 2.0: Benchmarking Recommendation for Rigorous Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2022).

[27] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *Proceedings of the 14th ACM Conference on Recommender Systems*.

[28] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. 2017. Dropoutnet: Addressing cold start in recommender systems. *Advances in neural information processing systems* 30 (2017).

[29] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded recommendation for alleviating bias amplification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1717–1725.

[30] Yaqing Wang, Hongming Piao, Daxiang Dong, Quanming Yao, and Jingbo Zhou. 2024. Warming Up Cold-Start CTR Prediction by Learning Item-Specific Feature Interactions. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3233–3244.

[31] Yuhan Wang, Qing Xie, Mengzi Tang, Lin Li, Jingling Yuan, and Yongjian Liu. 2024. Amazon-KG: A Knowledge Graph Enhanced Cross-Domain Recommendation Dataset. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 123–130.

[32] Tianjun Wei, Tommy WS Chow, and Jianghong Ma. 2024. FPSR+: Toward Robust, Efficient and Scalable Collaborative Filtering With Partition-aware Item Similarity Modeling. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[33] Tianjun Wei, Jianghong Ma, and Tommy WS Chow. 2023. Collaborative residual metric learning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1107–1116.

[34] Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. 2020. Self-supervised reinforcement learning for recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 931–940.

[35] Mengyue Yang, Jun Wang, and Jean-Francois Ton. 2023. Rectifying unfairness in recommendation feedback loop. In *Proceedings of the 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*. 28–37.

[36] Hyunsik Yoo, Zhichen Zeng, Jian Kang, Ruizhong Qiu, David Zhou, Zhining Liu, Fei Wang, Charlie Xu, Eunice Chan, and Hanghang Tong. 2024. Ensuring user-side fairness in dynamic recommender systems. In *Proceedings of the ACM on Web Conference 2024*. 3667–3678.

[37] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal intervention for leveraging popularity bias in recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 11–20.

[38] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with negative feedback via pairwise deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1040–1048.

[39] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 167–176.

[40] Huachi Zhou, Hao Chen, Junnan Dong, Daochen Zha, Chuang Zhou, and Xiao Huang. 2023. Adaptive popularity debiasing aggregator for graph collaborative filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 7–17.

[41] Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity bias in dynamic recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2439–2449.

[42] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-opportunity bias in collaborative filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 85–93.

[43] Ziwei Zhu, Jingu Kim, Trung Nguyen, Aish Fenton, and James Caverlee. 2021. Fairness among new items in cold start recommender systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 767–776.