Neurocomputing 704 (2026) 134768

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Dynamic interest drift detection: Leveraging bandit algorithm with UCB-based exploration

Huiying Wang [a, b], Shen Yang [a, b], Qifeng Zhou [a, b,*], Qing Wang [c]

[a] Department of Automation, Xiamen University, Xiamen, 361005, China
[b] Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-making, Xiamen, 361005, China
[c] Department of Cognitive Service Foundations, IBM T.J. Watson Research Center, Yorktown Heights, NY, USA

HIGHLIGHTS

- A new multi-armed bandit recommendation model is proposed to tackle dynamic and sparse-history cold-start in online systems.
- RNN-based interest drift detector tracks user preference changes over time via historical behavior, enriching user profiles.
- Neural bandit models the non-linear relationship and incorporates dynamic user representations into uncertainty estimation.
- Experiments show that integrating dynamic user modeling into the bandit mechanism leads to more adaptive and accurate recommendations.

ARTICLE INFO

Communicated by Z. Guan

Keywords:
Recommendation system
Interest drift detection
Multi-armed bandit model

ABSTRACT

Online recommendation systems play a crucial role in helping users discover items that match their interests and preferences. However, the performance of existing systems is often limited in sparse-history cold-start scenarios, where only a small number of interactions are available to infer user preferences. Meanwhile, user interests are inherently dynamic and may evolve over time, further increasing the difficulty of accurate recommendations. This paper aims to alleviate the sparse-history cold-start problem by integrating sequential user behavior into online recommendations. We propose MABID, a Multi-Armed Bandit-based recommendation framework integrated with an Interest Drift Detection network, designed to deliver accurate and adaptive recommendations in dynamic environments. Specifically, we design an interest drift detection module to capture temporal changes in user preferences from interaction sequences. Furthermore, recognizing the non-linear relationship between user-item features and rewards, we employ a neural bandit model to quantify uncertainty through gradient-based information for effective exploration and exploitation. Extensive experiments on three real-world datasets, i.e., LastFM, MovieLens, and Yelp, demonstrate the effectiveness of the proposed method.

1. Introduction

Amidst the exponential growth of digital information and the increasing demand for enhanced user experiences, online recommendation systems have gained unprecedented popularity. These systems now play crucial roles in the information services of modern society. The primary goal of an online recommendation system is to align users with their interests from vast volumes of information quickly and accurately, thus providing online personalized recommendations in various online services, including E-commerce and online advertising [1].

Within online recommendation systems, the continuous growth and change in both users and items often give rise to challenges, especially with respect to new users or items—a phenomenon known as the cold-start problem [2–4]. Due to the lack of consumption history and feature information for new users and new items, providing efficient personalized recommendations is recognized as a complex task. In some

* Corresponding author at: Department of Automation, Xiamen University, Xiamen, 361005, China.
Email address: zhouqf@xmu.edu.cn (Q. Zhou).

https://doi.org/10.1016/j.neucom.2026.134768
Received 12 November 2024; Received in revised form 19 July 2026; Accepted 7 August 2026
Available online 14 August 2026

studies, such cold-start recommendations are modeled as a Multi-Armed Bandit (MAB) problem [5,6]. The major problem faced by MAB is the "Exploitation-Exploration" dilemma, which is to strike a trade-off between satisfying observable interests and exploring potential interests to maximize the long-term benefits. In recent years, some MAB-based methods [7] have been proposed and successfully leveraged in online recommendation systems including ϵ-greedy strategy [8], Thompson Sampling [9], Upper Confidence Bound algorithm (UCB) [10] as well as the context-based bandit [11], e.g., LinUCB [12].

Context-based bandits optimize decision-making by incorporating contextual information with user feedback. However, the context-based bandit problem presents challenges from two key aspects. One challenge lies in the *non-linear relationship between the feature vector and the expected reward*, introducing complexity in practice. Another significant challenge arises from the *dynamic nature of user interests and demands*, which evolve over time due to external influences and changing environments. Some classical context-based bandit approaches, such as LinUCB and LinTS [13], are derived to employ linear models to estimate the expected reward thus reducing the complexity of the solutions. To overcome this limitation, researchers integrate neural networks to enhance feature expressiveness and capture non-linear relationships in the bandit setting, e.g., NeuralUCB [14] and NeuralTS [15]. These approaches challenge the linearity assumption in reward estimation and provide theoretical guarantees for cumulative regret bounds.

Although these neural network-based methods partially address the first challenge, certain difficulties remain unsolved. As highlighted in the second challenge, capturing the dynamical evolution of user interests over time is another pivotal factor in online recommendations. To illustrate this, Fig. 1 presents an example of interest drift using the Movielens dataset. In this example, a user sequentially rates ten movies with corresponding IDs from time T0 to T9. This dataset includes seven common movie genres, each represented by a binary indicator (1 if the movie belongs to the genre, 0 otherwise).

Initially, from T0 to T1, User 10 shows a strong interest in comedy and romance genres. However, from T2, a noticeable interest shift occurs towards horror and thriller movies. From T4, User 10 begins to focus more on suspense and action genres. After T8, his/her preference returns to comedy and romance movies. This phenomenon suggests that accumulating user dynamic feedback and regularly updating historical behavior information are also important in online recommendations. A review of recent user interaction sequences can gain valuable insights into user preferences, enabling the system to capture both general interests and the dynamic shifts in their preferences over time.

However, the available methods are less effective in capturing both the nonlinear relationships between user features and the expected reward, as well as dynamic user interest drifts. On the one hand, the existing MAB-based interest drift detection models, such as PSLinUCB [16] and TVUCB [17], can effectively balance the "Exploitation-Exploration" problem and achieve a trade-off between long-term and short-term gains, but these methods struggle to capture sequential information (i.e., users' recent interactive sequences) and user behavior trends because they mainly rely on the current click records for recommendation.

Time		T0	T1	T2	T3	T4	T5	T6	T7	T8	T9
Movie ID		4361	3361	3863	1347	2944	659	1186	2707	3868	2572
Movie Genre	Romance	1	1	0	0	0	0	0	0	1	1
	Comedy	1	1	0	0	0	0	0	0	1	1
	Action	0	0	0	0	1	0	0	0	1	0
	Crime	0	0	0	0	0	1	0	0	0	0
	Drama	0	1	1	0	1	1	1	0	0	0
	Thriller	0	0	1	1	0	1	0	1	0	0
	Horror	0	0	1	1	0	0	0	0	0	0

Fig. 1. An interesting drift example of user 10 on Movielens dataset.

Moreover, both methods rely on the linear assumption between rewards and features that limits their capability to model the intricate relationships in real-world problems. On the other hand, Recurrent Neural Networks (RNNs) [18] have been employed to capture evolving patterns in sequential data due to their strong expressiveness and high adaptability aligned with data characteristics. However, their application in online recommendation scenarios remains underexplored. RNNs can leverage known user preferences to generate accurate recommendations, but they struggle with exploration, limiting their ability to identify new items that users might be interested in.

Building on these insights, in this paper, we further investigate the evolution of user preferences in interaction sequences and propose a Multi-Armed Bandit-based recommendation model with an Interest Drift Detection network, named MABID. The model adopts a Multi-Armed Bandit framework with a UCB-based exploration strategy, selecting items according to their upper confidence bounds to balance exploration and exploitation. To model the dynamic nature of user preferences, we introduce an interest drift detection network that captures temporal changes in user behavior and estimates reward expectations under non-linear feature interactions. This detection module consists of two components: a dynamic interest extraction layer and a feature fusion layer. The former leverages sequential interaction data to learn time-aware user representations, while the latter integrates these representations with static user features to form a comprehensive user profile.

In summary, the main contributions of this work can be summarized as follows:

- We propose a novel online context-aware recommendation model that integrates sequential user behavior to better capture evolving user interests. This model leverages a non-linear, time-varying multi-armed bandit framework to deliver precise and timely recommendations, particularly in dynamic and sparse-history cold-start scenarios.
- We present an interest drift detection module that combines a Gated Recurrent Unit with a two-tower model to capture dynamic evolution of user interests over time from recent interaction sequence. By extracting sequential information this module can obtain richer user features to discern the shifts in user preferences over time.
- Extensive experiments on three real-world datasets demonstrate the effectiveness of the proposed framework, and validate that incorporating dynamic user representations into the bandit decision process leads to more adaptive and accurate recommendations.

2. Related work

In this section, we delve into several research directions related to our work on online recommendation, including contextual linear bandit models, non-linear bandit models, and interest drift detection models.

2.1. Linear bandit models

The primary goal of MAB models is to address the crucial challenge of balancing exploration and exploitation in cold-start recommendation. Various methods, including UCB [10], Thompson Sampling [9], and ϵ-greedy [8], have been proposed to solve this challenge. UCB algorithm optimizes decisions through uncertain information, which comprises the estimated means of rewards and the widths of confidence bounds. Thompson Sampling utilizes a Beta distribution to model the expected reward of each action within a Bayesian setting. ϵ-greedy mostly exploits the best-known option and occasionally explores a random one. Given the importance of contextual information in recommendations, researchers have increasingly focused on context-based algorithms [19]. Some classic methods, such as LinUCB [12], enhance the use of user and item information in modeling and achieve success in Yahoo! online news recommendation. This work employs a linear model to estimate

the upper confidence bound of the reward for each news, selecting the news item with the highest upper confidence bound for recommendation. Building upon the Thompson Sampling technique, LinTS [13] also utilizes a linear payoff function in the contextual bandit. Instead of the linear model, LogUCB [20] and LogisticTS [9] use the logistic regression on features to estimate the expected rewards. More recently, linear bandit frameworks have been extended to interactive and conversational recommendation scenarios. For example, recent work [21] integrates the Expected Value of Information (EVOI) with linear contextual bandits using stochastic gradient descent and smoothed key term contexts to efficiently elicit user preferences.

2.2. Non-linear bandit models

Although the linear-reward assumption has proven successful in both theoretical analysis and some applications, it rarely holds in practice due to its limited ability to handle complex features. To overcome this limitation and enhance feature expressiveness, several neural contextual bandit models have been proposed to model non-linear relationships more effectively.

NeuralUCB [14] combines UCB with a two-layer neural network to estimate the upper confidence bound, which challenges the assumption of a linear relationship and provides a theoretical guarantee for the confidence interval and cumulative regret. Subsequently, NeuralTS [15] employs a neural network to estimate the reward instead of the linear model in LinTS. DeepLinUCB [22] adopts a deep neural network to convert the contextual features in reward estimation for better representations. Ref. [23] deploys a two-tower deep model NRMS to generate the non-linear representations of users and items, and uses the UCB strategy as the foundation for recommendation. In Ref. [24], EE-Net algorithm is designed for a better resolution of the typical "Exploitation-Exploration" dilemma. It employs a novel strategy using an Exploitation network to learn the expected reward and another Exploration network to estimate the potential gains, and then it utilizes a hybrid decision-maker to generate recommendations. In Ref. [25], GNB constructs two user graphs for exploitation and exploration, respectively. It extracts correlations within each user graph through graph neural network-based methods. To overcome the limitations of strictly linear assumptions, recent studies have explored hybrid contextual bandit frameworks. For instance, a reward-informed semi-personalized approach [26] employs decision trees to partition the user context space into homogeneous segments and performs Thompson Sampling within each segment, improving recommendation quality through non-parametric context modeling.

The contextual MAB algorithms discussed above, whether based on linear or non-linear assumptions, rely on static context information, i.e., assuming a fixed yet unknown reward mapping function, which limits their ability to capture the drifts in user interests over time.

2.3. Interest drift detection models

In real-world scenarios, user interests evolve dynamically over time [27]. Sequence models, motivated by the analysis of sequential data such as text sentences, time series, and other discrete sequence data, have been utilized to detect such changes. Recurrent Neural Network (RNN) [18] is a typical sequence model. The fundamental structure of the RNN comprises a set of hidden state vectors, which are updated at each time step of the input sequence; by this means, the RNN can convey information over time. However, the basic RNN model still struggles to capture long-term dependencies within a sequence. Additionally, its back-propagation algorithm often suffers from the issue of gradient vanishing. To mitigate these two problems, RNN variants including Long Short-Term Memory (LSTM) [28] and Gated Recurrent Unit (GRU) [29] have emerged.

In the field of MAB issues, some research also tries to deal with the contextual MAB problem by using the dynamic behaviors of reward. In Ref. [30], a windowed change detection algorithm to capture an abrupt shift in the mean of a process is proposed. The work of Ref.[31] predicts sudden variations in users' behavior by computing the distance between the windows of two intervals and then aggregating the old and new preferences for recommendation. In Ref. [17], a time-varying context drift model is presented to track the dynamic context change and to infer the latent parameters and state random variables via particle learning. PSLinUCB [16] is developed to characterize the dynamically changing user interests under the piecewise-stationary assumption. This work estimates the preference vectors and employs a sliding window for each arm consisting of the latest reward observations to detect the preference change. In Ref. [32], dLinUCB detects and adapts to changes in the non-stationary environment by maintaining an admissible suite of contextual bandit models as slave bandits, to estimate the reward distribution. When a change is detected with high confidence, the master bandit discards the out-of-date slave bandits and creates new ones to fit the new environment. A dynamic ensemble of bandit models, DenBand in Ref. [33] capitalizes on the context-dependent property of environmental changes to conquer the non-stationary environment. It comprises bandit experts providing reward estimations for a given arm and a bandit auditor evaluating whether the monitored bandit expert is admissible. In Ref. [34], HyperBandit is proposed, which focuses on the periodicity of reward drift over time and employs Multi-Layer Perceptron (MLP) to generate the parameters from time features, in order to estimate the time-varying rewards for streaming recommendations. In Ref. [35], GRC uses the MLP to learn the cross-modal user-item interactions with the metric learning technique in a graph to capture users' dynamic preferences.

Existing MAB-based interest drift detection models are typically linear or parametric contextual bandits, which do not explicitly model users' historical behavior sequences. As a result, they fail to capture contextual temporal dependencies for online recommendations. In this work, we introduce an interest drift detection module under a non-linear assumption to learn the underlying reward function from sequential behavior data, thereby providing more accurate and timely recommendations.

3. Problem formulation

In this section, we formulate the MAB problem in the context of online recommendation systems. A glossary of key notations mentioned in our model is listed in Table 1.

The multi-armed bandit problem is a classic problem in probability theory, statistics, and machine learning [36,37]. The multi-armed bandit machine acts as an agent, and its selection of arms is referred to as an action. The agent takes an action and then receives a corresponding reward generated by the unknown distribution [38]. The multi-armed bandit problem can be formulated mathematically as follows. It involves making a series of decisions over a limited, but possibly unknown round horizon T. Assume that there are K items in the online recommendation system for each user to select. We consider it as a K-armed contextual bandit problem with the known number of total rounds T, where each item corresponds to an arm. Let $\mathbf{A}$ be a set of arms at each round $t \in [1, 2, \dots, T]$, denoted as $\mathbf{A} = \{a_{t,1}, a_{t,2}, \dots, a_{t,K}\}$. A d-dimensional feature vector $\mathbf{x}_{t,a}$ represents the context of each arm in A. At each round t, the learning agent observes the context of K arms, consisting of K feature vectors: $\mathbf{X}_t = \{\mathbf{x}_{t,a} \in \mathbb{R}^d | a \in [1, 2, \dots, K]\}$. The feature vectors can be computed by the matrix factorization of the user-item interaction matrix. Based on the context $\mathbf{X}_t$, the agent makes a decision to select an arm a_t and then obtains a reward r_{t,a_t}. A binary variable r_{t,a_t} represents the feedback received by pulling the arm a_t of the agent at round t. Since the distribution of this reward is unknown, it needs to be estimated by the multi-armed bandit solutions (e.g., UCB, Thompson sampling). The agent collects feedback at each round to update its parameters and determine the probability distribution, reducing cumulative regret across T rounds. Thus, the goal of the multi-armed bandit problem described above is to select a proper policy $\pi = \{a_1, a_2, \dots, a_T\}$ to minimize the

Table 1
Notations.

Notation	Description
M	the number of users
N	the number of items
K	the number of arms
T	the number of rounds
$\mathbf{A}$	the set of arms
$\mathbf{x}_{t,a}$	the context of the arm a represented by a d-dimensional feature vector at round t
$\mathbf{X}_t$	the context of K arms at round t
a_t	the pulled arm at round t
a_t^*	the optimal action at round t
r_{t,a_t}	the reward received by pulling the arm a_t
π	the policy selected sequentially of total T rounds
$Regret_\pi$	the cumulative regret of policy π
$h(\mathbf{x}_{t,a})$	the unknown reward distribution of the arm a at round t
ϵ_t	Gaussian noise
$\boldsymbol{\theta}$	the parameter of the model
$f(\mathbf{x}_{t,a};\boldsymbol{\theta})$	the reward prediction of the arm a at round t
$CB(\mathbf{x}_{t,a};f(\mathbf{x}_{t,a};\boldsymbol{\theta}))$	the confidence interval width of the arm a at round t
$U_{t,a}$	the estimated upper confidence bound of the arm a at round t
$\mathbf{P},\mathbf{Q}$	the user matrix and the item matrix
$\mathbf{h}$	the initial user feature vector
$\mathbf{v}$	the initial item feature vector
$\mathbf{s}_t$	the feature vector of user's recent interactions at round t
$\mathbf{z}_t$	the update gate of GRU at round t
$\mathbf{e}_t$	the reset gate of GRU at round t
$\mathbf{c}_t$	the memory cell of GRU at round t
$\mathbf{s}'_t$	the hidden state vector of user interest at round t
$\mathbf{h}'$	the user feature vector containing the information of interest drifts
$\boldsymbol{\mu}$	the transformed user feature vector
$\boldsymbol{\eta}$	the transformed item feature vector
$\mathbf{D}_t$	the records of the selected arm and its corresponding reward at round t

following cumulative regret [39]:

$$Regret_\pi = \sum_{t=1}^{T}(r_{t,a_t^*} - r_{t,a_t}) \tag{1}$$

where $a_t^* = \arg\max_{a\in[1,2,\ldots,K]} \mathbb{E}[r_{t,a}]$ is the optimal action at round t with the maximum reward expectation.

In the UCB algorithm, it is assumed that the rewards r_t of K arms are drawn independently from a fixed yet unknown distribution in each round [10]. This algorithm is founded upon an upper confidence bound that is expressed as the sum of an estimate for the true expected reward and a deviation. Therefore, we make the following assumption of context combination about reward generation to predict the reward: for any round t,

$$r_{t,a_t} = h(\mathbf{x}_{t,a_t}) + \epsilon_t \tag{2}$$

where function $h(\cdot)$ represents the unknown reward distribution satisfying $0 \le h(\mathbf{x}) \le 1$ and ϵ_t is Gaussian noise satisfying $\mathbb{E}(\epsilon_t) = 0$.

To simplify the solution process, the expected reward on an arm a is often assumed to be a linear combination of the contextual vector $\mathbf{x}_{t,a}$ and a coefficient vector $\boldsymbol{\theta}$:

$$h(\mathbf{x}_{t,a}) = \mathbf{x}_{t,a}^T\boldsymbol{\theta} \tag{3}$$

In rapidly changing dynamic environments, the non-linear function can quickly adapt to shifts in reward distributions due to its robust feature-learning capability, compared to the linear function 3. To overcome the inadequacy of the linear assumption, in this work, we make the non-linear assumption to model complicated relationships between rewards and features and design a neural network-based module $f(\mathbf{x}_{t,a};\boldsymbol{\theta})$ to learn the above reward function $h(\mathbf{x}_{t,a})$. By incorporating the UCB exploration strategy, the model can more accurately estimate the potential benefits of unfamiliar items, effectively balancing exploration and exploitation. Accordingly, the arm's upper confidence bound $U_{t,a}$, consisting of reward expectation and reward deviation, is obtained as:

$$U_{t,a} = f(\mathbf{x}_{t,a};\boldsymbol{\theta}) + \alpha \cdot CB(\mathbf{x}_{t,a}; f(\mathbf{x}_{t,a};\boldsymbol{\theta})) \tag{4}$$

where $f(\cdot)$ estimates the unknown reward expectation, $CB(\cdot)$ calculates the width of confidence interval as the reward deviation, $\boldsymbol{\theta}$ represents the parameters of function $f(\cdot)$, and exploration factor α is a hyper-parameter to balance the trade-off between exploitation and exploration. Ultimately, the arm with the highest upper confidence bound will be pulled:

$$a_t = \arg\max_{a\in[1,2,\ldots,K]} U_{t,a} \tag{5}$$

Consequently, the final objective function of predicting the most likely action for the given contexts is re-formulated as follows which strives for the minimum cumulative regret in (1).

$$a_t = \arg\max_{a\in[1,2,\ldots,K]} (f(\mathbf{x}_{t,a};\boldsymbol{\theta}) + \alpha \cdot CB(\mathbf{x}_{t,a}; f(\mathbf{x}_{t,a};\boldsymbol{\theta}))) \tag{6}$$

4. Solution & algorithms

In this section, we first present an overview of the proposed online recommendation model, MABID. Following this, we provide a detailed explanation of each component within the model.

4.1. Framework overview

The overall framework of MABID is illustrated in Fig. 2. It contains three major components: *initialization module*, *interest drift detection module*, and *prediction module*. MABID first employs matrix factorization to obtain initial features for both users and items. Then, it integrates an *interest drift detection module* to track the evolving interests of users. Finally, the model estimates item rewards and adopts the UCB strategy to select the most promising arm (i.e., the arm with the highest certainty) from the available K arms, and then delivers it to the current user. For training and optimizing the model, users' feedback is collected to update the model's parameters. These steps are detailed in the following sections, respectively.

4.2. Initialization module

Matrix factorization [40], as a latent factor model, has demonstrated superior performance in model-based collaborative filtering tasks [41]. It maps users and items into a shared latent factor space, where user-item interactions are modeled as inner products between the user and item matrices. Users' preferences for items are captured by the user-item co-occurrence matrix. In practical scenarios, this matrix is often sparse. Matrix factorization offers advantages, particularly when handling sparse data, and exhibits superior generalization ability. In this work, user and item embeddings are pre-trained offline using matrix factorization on historical interaction data. During the online interaction phase, these embeddings are treated as fixed features and are not updated using future observations. For newly arriving users or items without prior interactions, we assign a default embedding and rely on the online learning component to refine decisions based on observed rewards. This design ensures that no future information is utilized at decision time. Accordingly, the proposed framework is designed for sparse-history recommendation settings rather than strict zero-interaction cold-start scenarios, since the offline matrix factorization step requires historical interaction data to obtain the initial embeddings.

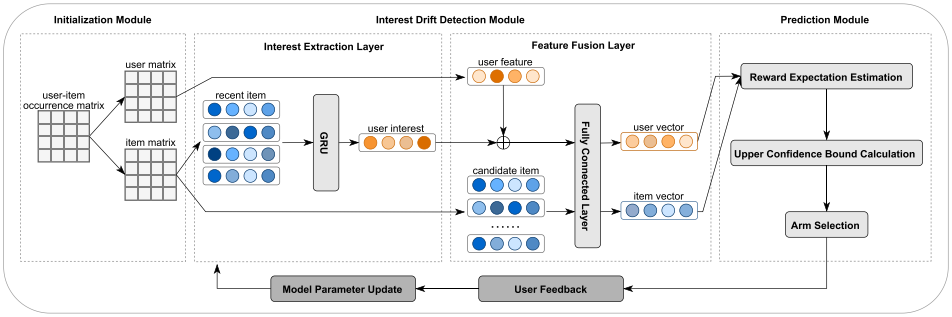

Fig. 2. The overall framework of the proposed model MABID. After obtaining static features of users and items via an *initialization module*, MABID introduces an *interest drift detection module* to capture dynamic preference changes based on historical interactions, and then recommends an item through a *prediction module* to balance exploitation and exploration.

Each element in the co-occurrence matrix indicates whether the corresponding user clicks the respective item, denoted as 1 for clicked and 0 for not clicked. The factorization technique will produce low-dimensional hidden vectors that can be utilized to express the potential characteristics of users and items. In the space defined by these latent vectors, the users or items with similar characteristics are positioned closer together. Assume that the system has M users and N items. Given a user-item interaction matrix $\mathbf{R} \in \mathbb{R}^{M\times N}$, the matrix factorization technique decomposes it into a user matrix $\mathbf{P} \in \mathbb{R}^{M\times d}$ and an item matrix $\mathbf{Q} \in \mathbb{R}^{N\times d}$, where d refers to the dimension of the latent feature vectors. Furthermore, each user is associated with a feature vector $\mathbf{h}_u \in \mathbb{R}^d$ and each item is associated with a feature vector $\mathbf{v}_i \in \mathbb{R}^d$. The probability of user u clicking on item i, denoted by $\hat{y}_{u,i}$, can be predicted by:

$$\hat{y}_{u,i} = \mathbf{v}_i^T \mathbf{h}_u \tag{7}$$

To further enhance the representation capability of features for different users and items, we introduce biases into the matrix factorization. For any user u and item i, the predicted interaction $\hat{y}_{u,i}$ between user u and item i can be computed by:

$$\hat{y}_{u,i} = \mathbf{v}_i^T \mathbf{h}_u + b_{u,i} + b_u + b_i \tag{8}$$

where $b_{u,i}$, b_u, and b_i are the average occurrences for the interaction matrix, user bias, and item bias, respectively.

The core of matrix factorization is to adjust the latent feature vectors of users and items in the training process to minimize the discrepancy between the predicted interaction $\hat{y}$ and the real user-item interaction y for all user-item pairs. Accordingly, the training process can be optimized by minimizing the squared loss error function:

$$\min_{\mathbf{h}^*,\mathbf{v}^*} \sum_{(u,i)\in O} (y_{u,i} - \mathbf{v}_i^T \mathbf{h}_u - b_{u,i} - b_u - b_i)^2 \tag{9}$$

where O refers to the user-item pairwise training set in which $y_{u,i}$ is known. During the training process of matrix factorization, the algorithm relies on iterative computation to optimize parameters until it converges or a stopping criterion is met.

We employ the stochastic gradient descent technique to update the latent variables:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \gamma \cdot ((y_{u,i} - \mathbf{v}_i^T \mathbf{h}_u) \cdot \mathbf{h}_u) \tag{10}$$

$$\mathbf{h}_u \leftarrow \mathbf{h}_u + \gamma \cdot ((y_{u,i} - \mathbf{v}_i^T \mathbf{h}_u) \cdot \mathbf{v}_i) \tag{11}$$

where γ is the learning rate dictating the pace of parameter adjustments, it influences the speed of convergence. If the user matrix and item matrix are fitted perfectly with the loss function (9), the model will obtain better initial representations of users and items.

4.3. Interest drift detection module

Current bandit-based recommendation algorithms consistently assume the existence of a linear or fixed combination within the feature context vectors and rewards. However, in practice, this assumption rarely holds as features often exhibit more complexity (i.e., are often dynamic and non-linear) in their reward structures. To better capture such intricate relationships, the proposed model MABID leverages the strong representation power of the Two-tower recommendation model and the Gated Recurrent Unit, constructing the feature context vectors to learn the unknown reward function with a UCB exploration strategy.

Specifically, we introduce an interest drift detection module with a two-tower architecture to learn intricate contextual information and fuse it with initial features in Section 4.2. The classic two-tower recommendation model DSSM [42] constructs user and item towers to represent user and item features. These two towers independently learn representations for users and items by computing similarity scores via inner products. The score reflects the user's preference for a particular item. Inspired by this, we design a two-tower structure in the proposed interest drift detection module to learn user and item feature vectors, respectively, by extracting user interests from historical interaction sequences. This module comprises two key components: the *Interest Extraction Layer* and the *Feature Fusion Layer*.

4.3.1. Interest extraction layer

For user u, his/her recent interactions $\{a_1, a_2, \dots, a_t\}$ are gathered chronologically according to the feedback received at each round, we can construct a sequence $\mathbf{s} = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_t\}$ accompanied by their contextual information in $\mathbf{X}$. The sequence model GRU surpasses standard RNNs by employing fewer parameters while efficiently capturing long-range dependencies and maintaining comparable performance. GRU can effectively handle the temporal dependency by incorporating the hidden states from previous information and the current interactions. Therefore, in this work, GRU is adopted to detect the interest drifts from $\mathbf{s}$ and extract the feature information of user interest, denoted as $\mathbf{s}' = \{\mathbf{s}'_1, \mathbf{s}'_2, \dots, \mathbf{s}'_t\}$:

$$\{\mathbf{s}'_1, \mathbf{s}'_2, \dots, \mathbf{s}'_t\} = GRU(\{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_t\}) \tag{12}$$

Further, we elaborate on the network structure of GRU in (12) to illustrate the way of interest extraction at a time step. At time t, GRU controls information flow using update gate $\mathbf{z}_t$, reset gate $\mathbf{e}_t$, and memory cell $\mathbf{c}_t$:

$$\mathbf{z}_t = \sigma(\mathbf{W}_{sz}\mathbf{s}_t + \mathbf{W}_{s'z}\mathbf{s}'_{t-1}) \tag{13}$$

$$\mathbf{e}_t = \sigma(\mathbf{W}_{se}\mathbf{s}_t + \mathbf{W}_{s'e}\mathbf{s}'_{t-1}) \tag{14}$$

$$\mathbf{c}_t = \tanh(\mathbf{W}_{sc}\mathbf{s}_t + \mathbf{W}_{ec}(\mathbf{s}'_{t-1} \odot \mathbf{e}_t)) \tag{15}$$

where W_{sz}, $W_{s'z}$, W_{se}, $W_{s'e}$, W_{sc}, and W_{ec} are trained parameters of GRU's update gate, reset gate, and memory cell. Activation function $\sigma(\cdot)$

resizes the value to the interval (0,1), and $\tanh(\cdot)$ scales the value to the range (-1,1). Then, the output of GRU's hidden layer can be worked out:

$$\mathbf{s}'_t = (1 - \mathbf{z}_t) \odot \mathbf{c}_t + \mathbf{z}_t \odot \mathbf{s}'_{t-1} \tag{16}$$

where the hidden state vector $\mathbf{s}'_t$ represents the feature of the user's recent preference derived from the interaction sequence.

Additionally, in sparse-history cold-start recommendation scenarios, some users may have no interactions at the beginning. The MABID algorithm addresses this issue by randomly initializing users' historical interactions, ensuring the algorithm functions properly.

4.3.2. Feature fusion layer

The feature fusion layer aims to integrate the user's recent interest information into the user vector, yielding transformed user and item vectors. As the user's characteristics change over time, the user tower needs to be computed online to update the user vector. The user interest vector $\mathbf{s}'_t$, obtained in the interest extraction layer, as well as the initial user feature vector h, are taken as the inputs of the user tower. To capture a more comprehensive and up-to-date user representation, we concatenate the two vectors $\mathbf{s}'_t$ and $\mathbf{h}$, denoted as $\mathbf{h}'$, and then get a transformed user vector $\boldsymbol{\mu}$ through a fully connected layer:

$$\mathbf{h}' = \mathbf{h} \oplus \mathbf{s}'_t \tag{17}$$

$$\boldsymbol{\mu} = \tanh(\mathbf{h}'\mathbf{W}_1) \tag{18}$$

where $\oplus$ indicates concatenation and $\mathbf{W}_1$ is a learnable transformation weight compressing the combined vector $\mathbf{h}'$ into the latent space $\mathbb{R}^c$. The feature fusion layer considers both inherent user features and evolving user interests reflected in recent interactions, serving the purpose of enriching the user feature representation to capture dynamic drifts in user preferences over time.

Besides, since the attributes of items are relatively fixed, the item tower can be computed offline. We directly use the initial item feature vector $\mathbf{v}$ as the input to the item tower, and it is also transformed in the latent space. Thus, the transformed item vector $\boldsymbol{\eta}$ can be learned by a fully connected layer:

$$\boldsymbol{\eta} = \tanh(\mathbf{v}\mathbf{W}_2) \tag{19}$$

where $\mathbf{W}_2$ is a learnable transformation weight compressing the item feature vector $\mathbf{v}$ into the latent space $\mathbb{R}^c$.

4.4. Prediction module

According to the UCB-based exploration strategy, the reward consists of the reward expectation related to the feature information and the reward deviation determined by the degree of uncertainty. Aiming to estimate the reward expectation of each item in candidate pool, we calculate the similarity score of the user vector $\boldsymbol{\mu}$ and item vector $\boldsymbol{\eta}$:

$$f(\mathbf{x};\boldsymbol{\theta}) = \boldsymbol{\mu}^T\boldsymbol{\eta} \tag{20}$$

To balance the trade-off between exploration and exploitation, we need to estimate the reward deviation by calculating the confidence interval width, which can be interpreted as the uncertainty (i.e., the larger the deviation, the more exploration). The confidence interval width can be calculated as follows:

$$CB(\mathbf{x}; f(\mathbf{x};\boldsymbol{\theta})) = \sqrt{g(\mathbf{x};\boldsymbol{\theta})^T\mathbf{J}^{-1}g(\mathbf{x};\boldsymbol{\theta})/j} \tag{21}$$

where $g(\mathbf{x},\boldsymbol{\theta})$ is the gradient of $f(\mathbf{x};\boldsymbol{\theta})$ at $\boldsymbol{\theta}$, which means $g(\mathbf{x},\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} f(\mathbf{x};\boldsymbol{\theta})$. Matrix $\mathbf{J}$ fuses all the gradient information and parameter j depends on the number of model parameters $\boldsymbol{\theta}$. If an item has a wide confidence interval, it is likely to be selected multiple times. If the confidence interval is narrow, the items with large rewards tend to be selected more frequently. Consequently, based on (24) and (27), we can derive the reward estimation U on any arm at each round:

$$U = \boldsymbol{\mu}^T\boldsymbol{\eta} + \alpha \cdot \sqrt{g(\mathbf{x};\boldsymbol{\theta})^T\mathbf{J}^{-1}g(\mathbf{x};\boldsymbol{\theta})/j} \tag{22}$$

The item a with the highest U, as the ultimate arm selection, will be recommended to the current user:

$$a = \underset{a\in[1,2,\ldots,K]}{\arg\max}\ U \tag{23}$$

Algorithm 1 sums up all the aforementioned steps and outlines the MAB-based interest drift detection approach.

Algorithm 1 MABID.

Input: total rounds T, exploration factor α, regularization parameter λ, history sequence length L_h.
Initialization: Randomly initialize network parameter $\boldsymbol{\theta}_0$ and recent interactions $\{a_1, \ldots, a_{L_h}\}$ for each new user
for $t = 1, \ldots, T$ **do**
  Observe K-armed feature context $\mathbf{X}_t = \{\mathbf{x}_{t,a} | a \in [1, 2, \ldots, K]\}$
  for $a = 1, \ldots, K$ **do**
    Learn the transformed user vector $\boldsymbol{\mu}$ and item vector $\boldsymbol{\eta}$ through the *interest drift detection module*
    Calculate the current arm's reward $f(\mathbf{x}_{t,a};\boldsymbol{\theta}_{t-1}) = \boldsymbol{\mu}_{t,a}^T\boldsymbol{\eta}_{t,a}$
    Compute the current arm's upper confidence bound $U_{t,a} = f(\mathbf{x}_{t,a};\boldsymbol{\theta}_{t-1}) + \alpha \cdot \sqrt{g(\mathbf{x}_{t,a};\boldsymbol{\theta}_{t-1})^T\mathbf{J}_{t-1}^{-1}g(\mathbf{x}_{t,a};\boldsymbol{\theta}_{t-1})/j}$
    Let $a_t = \arg\max_{a\in[1,2,\ldots,K]} U_{t,a}$
  end for
  Play the arm a_t and observe the reward r_{t,a_t}
  Compute $\mathbf{J}_t = \mathbf{J}_{t-1} + g(\mathbf{x}_{t,a_t};\boldsymbol{\theta}_{t-1})g(\mathbf{x}_{t,a_t};\boldsymbol{\theta}_{t-1})^T/j$
  Update the network parameter $\boldsymbol{\theta}_t$ by calling Training Process
end for

4.5. Prediction module

According to the UCB-based exploration strategy, the reward consists of the reward expectation related to the feature information and the reward deviation determined by the degree of uncertainty. Aiming to estimate the reward expectation of each item in the candidate pool, we calculate the similarity score of the user vector $\boldsymbol{\mu}$ and item vector $\boldsymbol{\eta}$:

$$f(\mathbf{x};\boldsymbol{\theta}) = \boldsymbol{\mu}^T\boldsymbol{\eta} \tag{24}$$

Although $f(\mathbf{x};\boldsymbol{\theta})$ is nonlinear with respect to the input features due to the use of neural networks and sequential modeling, it remains smooth with respect to the model parameters $\boldsymbol{\theta}$. Therefore, we apply a first-order Taylor expansion around the current parameter $\boldsymbol{\theta}$:

$$f(\mathbf{x};\boldsymbol{\theta}') \approx f(\mathbf{x};\boldsymbol{\theta}) + g(\mathbf{x};\boldsymbol{\theta})^T(\boldsymbol{\theta}' - \boldsymbol{\theta}), \tag{25}$$

where $g(\mathbf{x};\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} f(\mathbf{x};\boldsymbol{\theta})$ is the gradient with respect to the model parameters. This approximation allows us to linearize the reward function in the gradient space.

This formulation is consistent with prior neural contextual bandit methods (e.g., NeuralUCB [14]), where gradients are used to construct confidence estimates for nonlinear models. Based on the linearized form, we estimate the uncertainty using the accumulated gradient information. Specifically, the covariance matrix is approximated by the outer products of historical gradients:

$$\mathbf{J} = \sum_{\tau=1}^{t} g(\mathbf{x}_\tau;\boldsymbol{\theta})g(\mathbf{x}_\tau;\boldsymbol{\theta})^T. \tag{26}$$

To balance the trade-off between exploration and exploitation, we estimate the reward deviation by calculating the confidence interval width, which represents the model's uncertainty. The confidence

interval width is then formulated as follows:

$$CB(\mathbf{x}; f(\mathbf{x};\theta)) = \sqrt{g(\mathbf{x};\theta)^T \mathbf{J}^{-1} g(\mathbf{x};\theta)/j} \tag{27}$$

where j is a normalization factor related to the parameter dimension.

Intuitively, $g(\mathbf{x};\theta)$ measures how sensitive the prediction is to parameter updates. If a direction is not well covered by historical data, the uncertainty remains large, leading to a wider confidence interval and encouraging exploration.

The uncertainty is defined in the parameter space rather than the input space. Therefore, the use of nonlinear and sequential architectures does not affect the validity of the confidence estimation.

Consequently, based on (24) and (27), we can derive the reward estimation U:

$$U = \boldsymbol{\mu}^T \boldsymbol{\eta} + \alpha \cdot \sqrt{g(\mathbf{x};\theta)^T \mathbf{J}^{-1} g(\mathbf{x};\theta)/j} \tag{28}$$

The item a with the highest U is selected for recommendation:

$$a = \underset{a \in [1,2,\ldots,K]}{\arg\max}\ U \tag{29}$$

Algorithm 1 summarizes the aforementioned steps and outlines the MAB-based interest drift detection approach.

4.6. Optimization

4.6.1. Loss function

The core of our model is to predict the expected reward accurately to determine the optimal action; hence a squared loss function is used to measure the difference between the estimated and actual rewards of the selected arm. Given a record $\mathbf{D}_t = (\mathbf{x}_{\tau,a_\tau}, r_{\tau,a_\tau})_{\tau=1}^{t}$, the loss function is defined as follows:

$$\mathcal{L}(\theta) = \sum_{\tau=1}^{t} (f(\mathbf{x}_{\tau,a_\tau};\theta) - r_{\tau,a_\tau})^2 + \lambda \|\theta\|_2^2 / 2 \tag{30}$$

where $f(\mathbf{x}_{\tau,a_\tau};\theta)$ denotes the predicted reward, r_{τ,a_τ} denotes the actual reward, and λ is the regularization parameter to avoid over-fitting problem.

4.6.2. Training process

MABID is essentially a reinforcement learning algorithm that requires continuous interactions to gain training samples. We choose the replay method [43] to train our model, and then utilize the gradient descent approach to update the model parameters θ_t. The training process is described in Algorithm 2. All experiments are conducted under an offline replay protocol using historical interaction logs, rather than in a live interactive environment. At each step, the model makes a decision based only on past observed interactions, and no future information is used. This setup simulates a sequential decision process and enables consistent comparisons across different methods using the same logged data. At each decision round, the candidate pool consists of one observed positive item and nine negative items sampled from the user's non-clicked item set. The position of the positive item is randomly assigned within the candidate list to avoid position bias. Negative items are sampled using user-specific random sampling from precomputed non-clicked items.

Algorithm 2 Training Process.

1: **Input**: reward record $\mathbf{D}_t$, initial model parameter θ_0, regularization parameter λ, learning rate β.
2: Define loss function $\mathcal{L}(\theta)$ as (30).
3: **for** $i = 1, \ldots, |\mathbf{D}_t|$ **do**
4: $\quad \theta_i = \theta_{i-1} - \beta \nabla L(\theta_{i-1})$
5: **end for**
6: **return** $\theta_t = \theta_{|\mathbf{D}_t|}$.

5. Experiments

To demonstrate the effectiveness of the proposed method, we conduct a series of experiments on three real-world datasets including the music streaming service data LastFM, the movie rating data Movielens, and the local business review data Yelp. We detail the experimental settings in Section 5.1. Subsequently, in Section 5.2, we conduct a comprehensive comparative analysis of the proposed MABID and other baselines to assess their performance. The detailed analysis of parameter sensitivity and the sequence model is presented in Sections 5.3 and 5.4, respectively. Moreover, the ablation study is conducted in Section 5.5. Finally, the algorithm complexity of the proposed model is provided in Section 5.6.

All experiments reported in this section follow the offline replay protocol described in Section 4.6.2. Since the evaluation relies on historical interactions collected under an unknown recommendation policy, the replay protocol cannot fully account for counterfactual feedback bias. Consequently, the reported recommendation performance and regret should be interpreted as comparative results under a common replay environment rather than direct estimates of real online recommendation performance. Nevertheless, because all methods are evaluated using the same historical interactions, candidate construction strategy, and reward definition, the protocol provides a consistent and fair basis for relative comparison.

5.1. Experimental settings

5.1.1. Dataset description

Since the recommendation process in MABID adheres to the sequential pattern of user history interactions, three real-world datasets LastFM, Movielens, and Yelp, adopted for empirical studies in this work, are first pre-processed according to the chronological order of user interactions with the items. Subsequently, the experiments are conducted on these datasets which contain temporal information.

- **LastFM** dataset is extracted from the music streaming service Last.fm, made available on the HetRec 2011 workshop. This dataset contains a social network with 1892 users and 17,632 artists. There are 11,946 tags used to tag artists by users. It contains information about the artists that the users have listened to, thus we treat the "listened artists" in each user as positive feedback.
- **Movielens** dataset contains 25 million ratings and more than a million tag applications for 62,423 movies by 162,541 users. The binary label is given by a 5-star rating: if the rating is more than 3, the label is 1, otherwise, the label is 0. We treat the "rated movies" with label 1 as positive feedback.
- **Yelp** dataset is availably adopted from the 2018 edition of the yelp challenge. Wherein local businesses like bars and restaurants are viewed as items. It includes 174,567 items and more than 5 million ratings by 1,326,101 users. The binary label is given by a 5-star rating: if the rating is more than 3, the label is 1, otherwise, the label is 0. We treat the "rated businesses" with label 1 as positive feedback.

5.1.2. Baselines

We conduct comparative experiments with the following ten baselines proposed in recent years to verify the effectiveness of the proposed MABID model in the dynamic and cold-start recommendation problem. Comparison methods include three types: (1) **models based on static and linear methods**, i.e., Random, Epsilon-greedy [8], LinUCB [12], and MABs [44]; (2) **models based on static and non-linear methods**, i.e., Neural ϵ-greedy [45], NeuralUCB [14], NeuralTS [15], and EE-Net [24]; (3) **models based on interest drift detection and linear methods**, i.e., PSLinUCB [16], dLinUCB [32], DenBand [33], and SCAPA-UCB [46]. Their descriptions are listed as follows:

- **Random.** It randomly selects one of all items in every timestep, without learning any contextual information.

- **ϵ-greedy.** It pulls a random arm with probability ϵ and selects a greedy arm with the highest empirical mean with probability $1-\epsilon$. A larger ϵ represents a greater degree of exploration.
- **LinUCB.** It assumes a linear combination between arm vectors and rewards, such as ridge regression, to estimate the expected reward. It selects the arm with the highest upper confidence bound of reward for recommendation.
- **MABs.** It groups arms into clusters and utilizes the dependencies among arms within the same cluster to share reward information and accelerate the learning process.
- **PSLinUCB.** It adopts a piecewise-stationary assumption on the unknown preference vector and a linear assumption on the expected reward estimation.
- **dLinUCB.** It chooses LinUCB as the slave bandit model and designs a strategy for slave model creation and abandonment based on the defined concept of "badness". The master bandit model selects one active slave model of lower "badness" with higher confidence.
- **DenBand.** It introduces a bandit auditor for each bandit expert to monitor its reward estimation quality and determine if the expert is admissible. An arm is selected by the upper confidence bound of reward estimation according to an ensemble of its admissible bandit experts set.
- **SCAPA-UCB.** It monitors reward distributions to detect changes and anomalies in non-stationary environments, allowing the model to adaptively update its policy.
- **Neural ϵ-greedy.** It utilizes the Multi-Layer Perceptron (MLP) to predict the reward and employs the ϵ-greedy to explore the potential reward.
- **NeuralUCB.** It uses a neural network-based random feature mapping to construct an upper confidence bound of reward for efficient exploration.
- **NeuralTS.** It adopts the deep neural network for exploitation and Thompson sampling strategy for exploration on the proposed novel posterior distribution of the reward.
- **EE-Net.** It employs a neural network as an Exploitation network for the reward function and uses another neural network as an Exploration network to estimate potential gains.

5.1.3. Evaluation metric

The performances of the proposed method and other baselines are evaluated in terms of the commonly used metric cumulative regret $Regret_\pi$:

$$Regret_\pi = \sum_{t=1}^{T} (r_{t,a_t^*} - r_{t,a_t})$$

which is also described concretely in Section 3. The recorded reward r_{t,a_t} is 1 if the selected arm a_t is equal to the optimal arm a_t^*, and 0 otherwise.

In our evaluation, the optimal arm a_t^* corresponds to the observed positive item within the constructed candidate set in each replay round. Therefore, the reward is 1 only when the selected item matches this positive item, and 0 otherwise. This implies that regret is incurred whenever the selected item does not match the positive item. This definition reflects decision quality within the replay environment. We note that this regret should be interpreted as an offline proxy for comparing different methods under the same evaluation protocol, rather than as an unbiased estimate of cumulative regret in a real online system. The cumulative regret quantifies the accumulated difference between the total rewards obtained and the maximum achievable rewards over the evaluation period. A lower cumulative regret signifies better performance.

In addition to cumulative regret, we adopt standard ranking-based metrics to evaluate recommendation quality. Specifically, a compact yet comprehensive set of metrics, including HR@1, HR@5, NDCG@5, and MRR, is adopted. These metrics jointly capture complementary aspects of recommendation performance without introducing redundant evaluation measures. HR@1 evaluates the accuracy of the top-ranked recommendation, while HR@5 measures the ability to retrieve the relevant item within a small candidate list. NDCG@5 further evaluates ranking quality by accounting for the position of the relevant item, and MRR summarizes how early the relevant item appears in the ranked list. Together, these metrics provide complementary perspectives on recommendation performance by evaluating both top-K retrieval effectiveness and position-aware ranking quality.

At each round t, the model ranks a candidate set of items, where the ground-truth positive item (i.e., the optimal arm a_t^*) is regarded as the only relevant item.

The Hit Ratio at cutoff K is defined as:

$$HR@K = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(\text{rank}_t < K) \tag{31}$$

The Normalized Discounted Cumulative Gain at cutoff K is defined as:

$$NDCG@K = \frac{1}{T}\sum_{t=1}^{T} \frac{\mathbb{I}(\text{rank}_t < K)}{\log_2(\text{rank}_t + 2)} \tag{32}$$

The Mean Reciprocal Rank is defined as:

$$MRR = \frac{1}{T}\sum_{t=1}^{T} \frac{1}{\text{rank}_t + 1} \tag{33}$$

Here, rank_t denotes the rank position of the ground-truth positive item in the ranked list at round t, and $\mathbb{I}(\cdot)$ is the indicator function.

5.1.4. Hyper-parameters settings

We use the grid search method for the hyper-parameters optimization of MABID. The hidden state size of the interest extraction layer and feature fusion layer are both set to 32. A GRU layer is employed to extract user interests from a historical sequence of user interactions of length 10. The Adam optimizer is utilized to optimize the parameters and the learning rate is set to 0.001 or 0.01. The exploration factor is set to 0.1 or 0.01 when calculating the upper confidence bound. There are 10,000 rounds set to train the proposed model MABID. The specific hyper-parameters settings of experiments on three datasets are shown in Table 2. At each decision round, the candidate pool consists of one observed positive item and nine negative items sampled from the user's non-clicked item set. The position of the positive item is randomly assigned within the candidate list to avoid position bias. Negative items are sampled using user-specific random sampling from precomputed non-clicked items.

5.2. Overall performance

To reduce the impact of randomness, the experiments are repeated five times for all methods and we report their averaged performance for comparison. Our model is configured with the optimal hyper-parameter settings recorded in Table 2. The main experimental results on three datasets are presented in Table 3 and Figs. 3–5. We additionally report ranking-based recommendation results in Table 4 for a more comprehensive evaluation. Note that in Tables 3 and 4 the best and second-best results are marked in bold and underlined, respectively.

Table 2
Hyper-parameters Settings.

Hyper-parameter	Dataset		
	LastFM	Movielens	Yelp
Numer of GRU layers	1	1	1
Hidden state size	32	32	32
Length of historical interaction sequence	10	10	10
Exploration factor	0.1	0.1	0.01
Number of rounds	10,000	10,000	10,000
Optimizer	Adam	Adam	Adam
Learning rate	0.001	0.001	0.01

Table 3
Overall Experimental Results.

Method	Dataset											
	LastFM				Movielens				Yelp			
	mean	std	max	min	mean	std	max	min	mean	std	max	min
Random	9021	10.61	9034	8996	8998	28.10	9045	8966	9001	23.64	9025	8961
ϵ-greedy	8408	70.32	8494	8303	7742	70.05	7853	7650	8663	42.08	8715	8602
LinUCB	6324	19.87	6340	6286	4938	36.94	4978	4872	6978	82.99	7139	6905
MABs	6529	49.23	6614	6469	5193	72.85	5290	5104	7367	72.43	7419	7225
dLinUCB	6539	31.37	6598	6508	4994	41.62	5050	4922	6572	15.47	6596	6550
DenBand	6111	34.64	6161	6065	4672	55.27	4730	4599	6174	62.13	6236	6057
PSLinUCB	7968	33.20	8012	7918	6848	121.80	7005	6713	8150	41.40	8219	8103
SCAPA-UCB	4377	36.98	4444	4345	4078	42.98	4150	4032	6713	48.65	6760	6628
Neural ϵ-Greedy	6319	172.12	6549	6097	5089	83.36	5247	5009	7026	266.32	7528	6768
NeuralTS	5036	207.86	5414	4807	4081	137.99	4308	3875	6145	110.27	6318	5976
NeuralUCB	4936	108.86	5042	4739	3961	55.56	4017	3866	6114	236.06	6322	5688
EE-Net	4833	118.35	4975	4688	4252	135.47	4419	4094	5521	248.15	5890	5126
MABID (ours)	**4080**	56.06	4158	4007	**3698**	51.78	3762	3629	**5156**	72.04	5288	5084

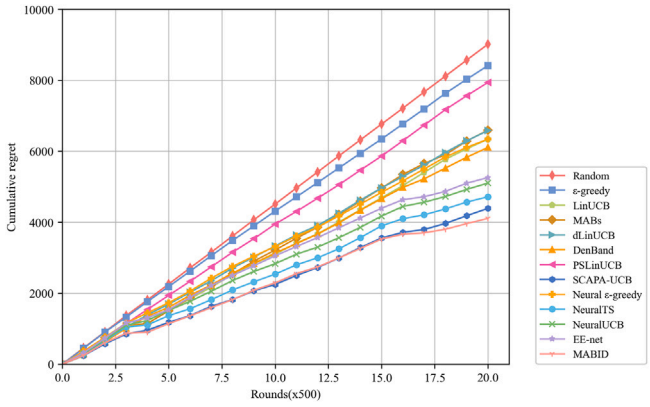

Fig. 3. Cumulative regret with different baselines on LastFM dataset.

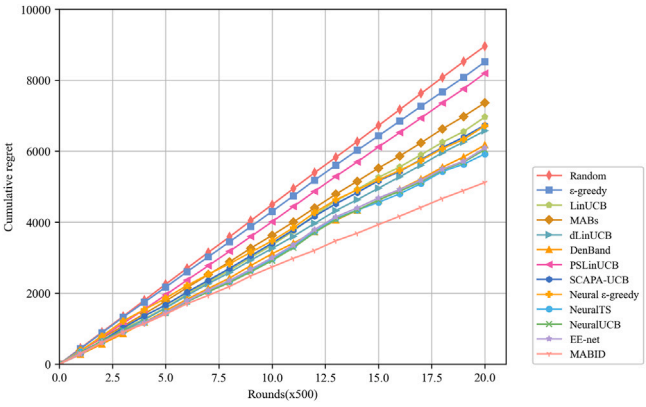

Fig. 5. Cumulative regret with different baselines on Yelp dataset.

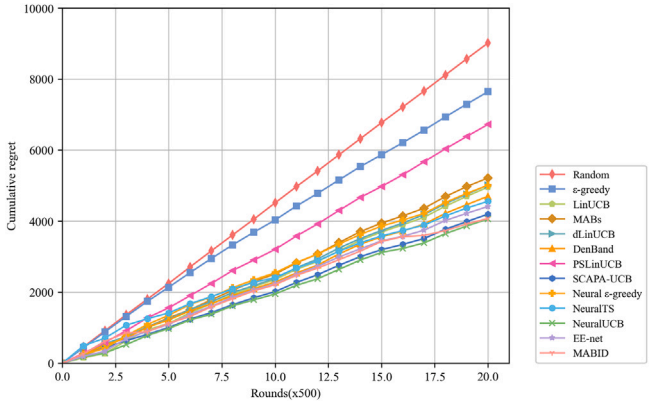

Fig. 4. Cumulative regret with different baselines on Movielens dataset.

From the main experimental results we can observe that conventional methods, *Random* and ϵ-greedy, exhibit relatively inferior performances due to their high level of randomness. Notably, the ϵ-greedy method outperforms the Random method, indicating the significance of the balance between exploitation and exploration in recommendations. ϵ-greedy allocates a probability of 1-ϵ for exploitation and the remaining for exploration. However, this strategy might result in suboptimal or locally optimal solutions since it does not consider the contextual information.

Compared with aforementioned two context-free methods, LinUCB achieves better performance. This may result from two reasons: one is that LinUCB considers the uncertainty of expected reward to seek the global optimal solutions, and the other is that it utilizes the contextual information of arms and estimates rewards under the linear assumption between context and reward. MABs show comparable but slightly inferior performance to LinUCB, indicating that simple bandit extensions without modeling user dynamics are insufficient in this setting.

Experimental results further show that, with the strong expressive power inherent in neural networks, neural network-based bandit models outperform linear assumption-based methods significantly. Methods like Neural ϵ-greedy, NeuralUCB, and NeuralTS calculate the reward distribution under the realistic non-linear assumption to enhance prediction performance. In particular, EE-Net utilizes two neural networks for exploitation and exploration respectively, leading to superior performance against the conventional and linear methods. SCAPA-UCB further improves performance by explicitly modeling non-stationarity through change and anomaly detection mechanisms, enabling it to adapt to shifts in reward distributions and achieve competitive results among advanced bandit baselines. Nevertheless, these contextual models make efforts to learn fixed yet unknown reward mapping functions thus ignoring the dynamic changes of the users' preferences. dLinUCB, DenBand, and PSLinUCB consider the time-varying interests of users to estimate the rewards. Their performances surpass the static and linear models, e.g., LinUCB, but are inferior to the non-linear models.

From Table 5, we further observe that while strong baselines such as SCAPA-UCB achieve competitive performance on ranking metrics,

Table 4
Recommendation performance comparison.

Method	Dataset											
	LastFM				Movielens				Yelp			
	HR@1	HR@5	NDCG@5	MRR	HR@1	HR@5	NDCG@5	MRR	HR@1	HR@5	NDCG@5	MRR
Random	0.0988	0.4981	0.2933	0.2916	0.1002	0.4985	0.2943	0.2927	0.0979	0.4975	0.2923	0.2908
ϵ-greedy	0.1584	0.5342	0.3423	0.3403	0.2295	0.5773	0.4010	0.3984	0.1359	0.5202	0.3241	0.3226
LinUCB	0.3676	0.7588	0.5708	0.5401	0.5062	0.8508	0.6913	0.6573	0.3022	0.6849	0.5029	0.4809
MABs	0.3472	0.6877	0.5238	0.5096	0.4804	0.7987	0.6519	0.6280	0.2633	0.6204	0.4476	0.4389
dLinUCB	0.3461	0.7211	0.5382	0.5147	0.5006	0.8184	0.6700	0.6439	0.3428	0.7487	0.5531	0.5215
DenBand	0.3889	0.7724	0.5873	0.5564	0.5328	0.8609	0.7091	0.6768	0.3826	0.7905	0.5948	0.5578
PSLinUCB	0.2032	0.5508	0.3731	0.3743	0.3152	0.5905	0.4536	0.4622	0.1850	0.5289	0.3554	0.3592
SCAPA-UCB	0.5672	**0.9441**	**0.7733**	**0.7241**	0.5973	**0.9237**	0.7771	0.7377	0.3308	0.7760	0.5610	0.5200
Neural ϵ-Greedy	0.3968	0.7945	0.6056	0.5700	0.5334	0.8752	0.7184	0.6820	0.3185	0.6411	0.4895	0.4817
NeuralTS	0.4964	0.8284	0.6731	0.6439	0.5919	0.8973	0.7586	0.7252	0.3855	0.7524	0.5836	0.5572
NeuralUCB	0.5064	0.8330	0.6801	0.6510	0.6039	0.9104	0.7713	0.7363	0.3885	0.7526	0.5859	0.5598
EE-Net	0.5167	0.8291	0.6823	0.6557	0.5748	0.8671	0.7342	0.7062	0.4479	0.8534	0.6668	0.6236
MABID (ours)	**0.5920**	0.8804	0.7458	0.7170	**0.6302**	0.9197	**0.7894**	**0.7559**	**0.4844**	**0.8911**	**0.7054**	**0.6576**

Table 5
Experimental results on different sequence Models.

Method	Dataset		
	LastFM	Movielens	Yelp
RNN	4623	3760	5237
LSTM	4465	3779	5128
GRU	**4224**	**3438**	**5103**

their improvements are not consistently reflected in cumulative regret. This discrepancy indicates that achieving high-quality rankings within a single round does not necessarily translate into effective long-term decision-making in bandit settings.

A possible reason is that ranking metrics evaluate the relative ordering of items within a fixed candidate set, without considering how uncertainty estimation influences future decisions. In contrast, cumulative regret explicitly captures the sequential decision process, where both reward estimation and uncertainty modeling jointly affect exploration behavior.

Therefore, methods that rely on static or decoupled representations may achieve strong ranking performance but fail to adapt their exploration strategy to evolving user preferences. In comparison, MABID integrates dynamic user representations into both reward prediction and uncertainty estimation, leading to more consistent improvements across both ranking-based metrics and cumulative regret.

The proposed MABID achieves the best overall performance in terms of cumulative regret across all three datasets, and consistently obtains strong results on ranking-based recommendation metrics.

5.2.1. LastFM dataset

On LastFM, MABID achieves the lowest cumulative regret (4080), consistently outperforming all baselines. It significantly improves over static and linear methods (*Random*, ϵ-greedy, LinUCB), as well as MABs, indicating the benefit of incorporating sequential user dynamics. Compared with drift-aware linear models (dLinUCB, DenBand, PSLinUCB), MABID maintains clear advantages. Although SCAPA-UCB performs competitively on ranking metrics, MABID achieves lower regret, suggesting that modeling user-level preference evolution provides more informative signals than relying on global non-stationarity detection. In Table 5, MABID achieves strong overall ranking performance.

5.2.2. Movielens dataset

On Movielens, MABID again achieves the lowest regret (3698), outperforming all categories of baselines. It consistently improves over static/linear methods and MABs, confirming the effectiveness of sequential contextual modeling under sparse interaction data. Compared with drift-aware baselines, MABID achieves consistent gains over dLinUCB (4994), DenBand (4672), and PSLinUCB (6848), and also outperforms SCAPA-UCB in regret despite its strong ranking performance. Specifically, SCAPA-UCB achieves competitive ranking results such as HR@1 = 0.5973 and NDCG@5 = 0.7771 in Table 5, but its regret remains higher than MABID, suggesting that strong one-step ranking quality does not necessarily imply better decision quality under sequential interaction. Table 5 further shows that MABID achieves the best performance on HR@1, NDCG@5, and MRR, and remains close to the best on HR@5, demonstrating consistent ranking effectiveness across multiple evaluation settings.

5.2.3. Yelp dataset

On Yelp, MABID achieves the best cumulative regret (5156), outperforming all baselines across different categories. Significant gains are observed over static/linear methods and MABs, highlighting the importance of modeling sequential behavior. Compared with drift-aware models, MABID consistently outperforms dLinUCB, DenBand, PSLinUCB, and SCAPA-UCB in regret. SCAPA-UCB remains competitive on ranking metrics, but MABID is more stable in decision quality under sequential interaction. MABID achieves the best performance on all ranking metrics in Table 5.

Overall, results across all datasets demonstrate that modeling sequential user behavior within the bandit framework consistently improves decision quality. Compared with static, neural, and drift-aware baselines, MABID achieves better regret and strong ranking performance, validating the effectiveness of jointly modeling user preference evolution and uncertainty in online recommendation.

The objective of MABID is to minimize the cumulative regret by optimizing the model parameters. The optimization process of the model is iterated until the parameter θ converges or the iteration number exceeds a preset limit T. The convergence curves on LastFM and Movielens datasets are respectively shown in Fig. 6(a) and (b). We can observe that the trends of learned objective curves tend to become constant after approximately 15,000 rounds on LastFM dataset and 27,000 rounds on Movielens dataset.

5.3. Further analysis

We further analyze the impact of three major hyper-parameters on the model's performance, including the number of GRU layers, the length of historical interaction sequence, and the exploration factor α.

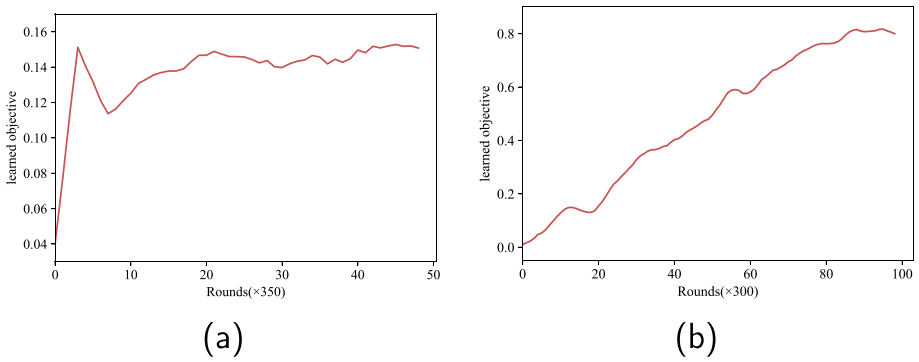

Fig. 6. Convergence curves on LastFM (a) and Movielens (b) datasets.

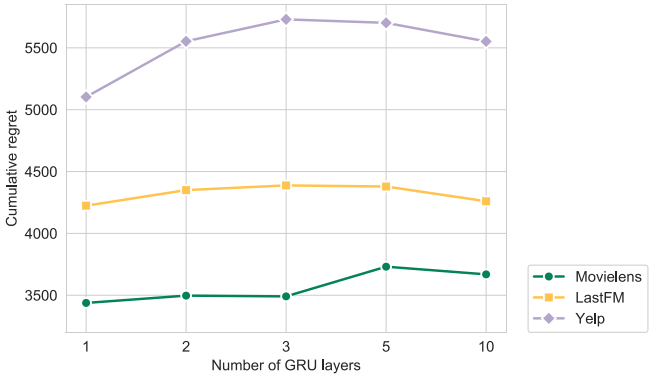

Fig. 7. Cumulative regret values of different GRU layers.

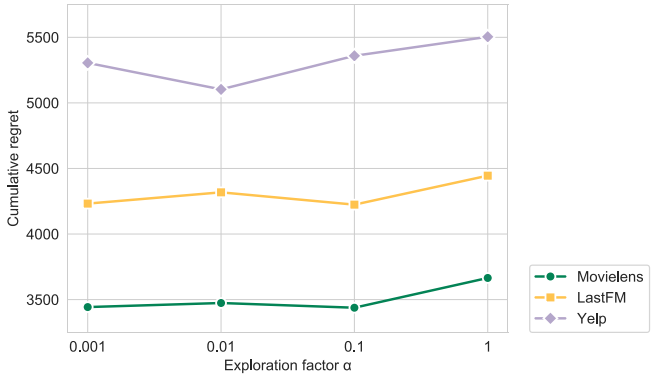

Fig. 8. Cumulative regret values for different exploration factor values.

5.3.1. *Number of GRU layers*

As illustrated in Fig. 7, when the number of GRU layers is set to 1 MABID obtains the best performance (i.e., the lowest cumulative regret) on all three datasets. When the number of GRU layers increases to 3 and 5, the model training parameters significantly increase, which makes the model training more challenging, thus increasing the cumulative regret. Subsequently, when the number of GRU layers is increased to 10, the cumulative regret exhibits a slight decline but is still higher than that of using fewer GRU layers. Therefore, in this work, we choose one layer of GRU, a simple but effective structure to train the MABID model.

5.3.2. *Exploration factor α*

The observation from Fig. 8 highlights that the factor α is crucial to balance exploration and exploitation in computing the upper confidence bound. With a large α, the algorithm tends to "exploration", while with a small α the algorithm tends to "exploitation". A smaller exploration factor is more likely to result in a locally optimal solution, making it challenging for the algorithm to discover items with higher rewards over the long term. Conversely, a too large α might trigger over-exploration, potentially causing the algorithm to overlook the important interests reflected in some valuable user histories.

5.4. *Sequence model analysis*

To analyze the effect of sequence models on recommended performance, we compare three different sequence models, GRU, RNN, and LSTM. The experimental results using cumulative regret evaluation are shown in Table 5.

From Table 5, we can observe that the best experimental results are achieved when using the GRU sequence model, and the other two models RNN and LSTM have approximately similar but inferior performance. This indicates that GRU has a stronger capability to capture the interest changes of users.

To further investigate the model performance under sparse-data and dynamic conditions, we conduct a grouped analysis based on the length of historical interaction sequences. Specifically, users are divided into three groups, i.e., short, medium, and long, according to quartiles of the interaction length distribution. That is, the short group corresponds to the bottom 25% of users in terms of interaction history length, the medium group corresponds to the middle 50%, and the long group corresponds to the top 25%. This setting allows us to explicitly evaluate the model's behavior in cold-start and data-rich scenarios.

From Fig. 9, we observe that MABID consistently outperforms the compared methods across all groups on both datasets. More importantly, the advantage is particularly evident in the short group, where only limited user interactions are available. For instance, on Yelp, MABID achieves a regret of 1459 in the short group, outperforming EE-Net (1517) and SCAPA-UCB (1674), while also obtaining the highest NDCG@5 of 0.6314. A similar trend can be observed on Movielens, where MABID achieves the lowest regret (809) and the highest NDCG@5 (0.8045) under the short setting.

As the interaction length increases, all methods show performance improvements, indicating that richer historical information benefits user preference modeling. However, MABID maintains competitive or superior performance in both medium and long groups. Notably, the performance gap is more significant in the short and medium groups, suggesting that MABID is more effective in leveraging limited interaction signals and adapting to evolving user preferences.

These results provide empirical evidence that the proposed model is capable of handling sparse-history scenarios, which commonly correspond to cold-start or early-stage interactions in real-world systems. By

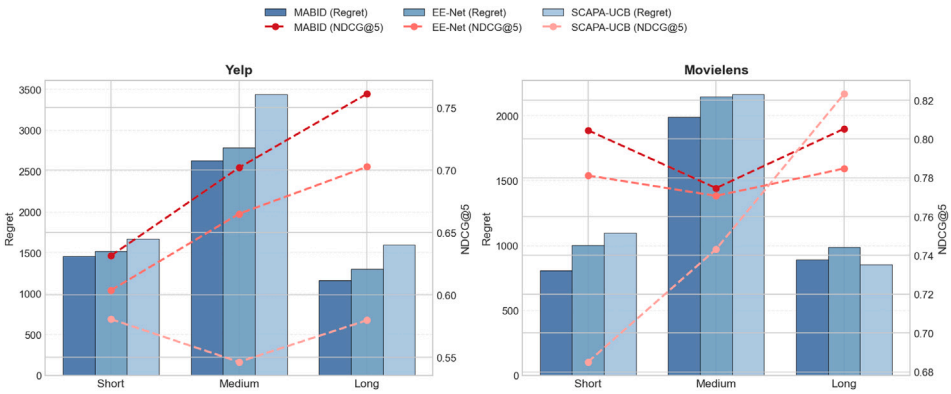

Fig. 9. Grouped Analysis by Historical Interaction Length.

Table 6
Ablation experimental Results.

Method	Dataset		
	LastFM	Movielens	Yelp
MABID-Seq*	4412	3647	5135
MABID-Seq	4464	3716	5185
MABID-User	5269	3977	5574
MABID-Greedy	4248	3682	5356
MABID-Decouple	5822	4284	6901
MABID	**4224**	**3438**	**5103**

effectively incorporating recent behavioral signals into the decision process, MABID demonstrates stronger adaptability when user information is insufficient, while still maintaining stable performance as more data becomes available.

5.5. Ablation study

MABID integrates multiple modules to extract the informative user features for reward estimation. In order to demonstrate the effect of different components, we design several variants of MABID and conduct the ablation study as follows:

- **MABID-Seq*.** It removes the interest extraction layer and only records the last one historical behavior of user to input into the feature fusion layer.
- **MABID-Seq.** It removes the interest extraction layer and only retains the feature fusion layer.
- **MABID-User.** It removes the user features from the feature fusion layer.
- **MABID-Greedy.** It keeps the same network architecture as MABID, but removes the UCB exploration bonus. At each round, the model selects the arm using only the predicted mean reward μ, instead of $\mu + \sigma$.
- **MABID-Decouple.** It separates sequential representation learning from uncertainty-based exploration. A GRU encoder first generates the user representation, which is then fed into a separate NeuralUCB head for reward prediction. Unlike the full MABID model, where the uncertainty is estimated from the gradients of all trainable parameters in an end-to-end manner, MABID-Decouple computes the uncertainty only from the gradients of the NeuralUCB head parameters.

As shown in Table 6, compared with the three variants, the original MABID achieves the best performance (i.e., the lowest cumulative regret) across all three datasets, validating the importance of each component in MABID. The MABID-Seq* model and the MABID-Seq model, excluding the interest extraction layer, exhibit decreased performance, this is attributed to their loss of essential user information. It also emphasizes the efficacy of adopting a sequence model to capture changes in user interests and to enhance the efficiency of sparse-history cold-start recommendations. Moreover, compared to MABID-Seq, MABID-Seq* (which considers only the user's most recent behavior) achieves slightly better performance, demonstrating the impact of users' recent interactions on preference prediction. A comparison between MABID-User and MABID indicates that incorporating user features enriches user information, thereby improving recommendation performance. In addition, the MABID-Greedy variant, which removes the UCB exploration term and relies only on the predicted mean reward, shows consistently worse performance than the full model, indicating that the improvement of MABID is not solely due to representation learning but also benefits from effective exploration during decision-making. Furthermore, the MABID-Decouple variant performs significantly worse than the full model across all datasets. In this variant, the sequential encoder is used only as a feature extractor, and a separate NeuralUCB head conducts the exploration. The large performance gap suggests that a simple combination of sequential modeling and bandit learning is insufficient. Instead, MABID benefits from a tighter integration in which sequential representations directly support both reward estimation and uncertainty-aware exploration, leading to better decisions over time.

5.6. Complexity of implementation

Here, we analyze the time complexity of MABID at a single step. Supposing that there is d-dimensional feature vector, an m-dimensional hidden state vector, c-dimensional transformed vector, and historical sequence length L_h. The time complexity of GRU is $O(L_h(dm + m^2))$. The output of GRU and feature initialization have to be transformed into the same latent space, which takes $O((d + m)c)$ time to complete. After the interest drift detection steps, the current arm's upper bound confidence can be calculated. It costs $O(c)$ time to estimate the reward. Thus, the overall time complexity of MABID is

$$O(L_h m^2 + (L_h d + c)m + (d + 1)c).$$

Moreover, we analyze the time complexity of other related algorithms for comparison. At a single step, the time complexity of both LinUCB and PSLinUCB is $O(d^2 + d)$. Both NeuralUCB and EE-Net have a time complexity of $O((L - 1)m^2 + (d + 1)m)$, where L represents the depth of fully connected neural networks ($L \geq 2$). If the dimensions of the input parameters, m and d, are taken into account, the time complexity of our proposed model is slightly higher than that of the linear static model (i.e., LinUCB) and the linear interest drift detection model (i.e., PSLinUCB), yet its performance is demonstrably superior. Compared to the non-linear models (i.e., NeuralUCB and EE-Net), our model has a comparable time complexity $O(m^2 + m)$ and exhibits a better performance (i.e., lower cumulative regret).

From a practical deployment perspective, the additional computation of MABID mainly comes from two components. First, the GRU encoder performs sequential modeling over recent user interactions. In our implementation, the history length is fixed to 10 and the hidden dimension is 32, resulting in approximately 4.2K trainable parameters for the GRU module. Therefore, the additional cost of sequence modeling remains small and grows linearly with the history length. Second, the NeuralUCB exploration strategy requires one additional gradient computation to estimate the uncertainty for each candidate item. In our experiments, the candidate pool contains only 10 items, and the entire network has approximately 6K trainable parameters. Therefore, each recommendation requires one forward pass and one gradient computation for each candidate item. In addition, model updates are performed once every 50 interaction rounds rather than after every recommendation, which further reduces the online computational burden. Under these experimental settings, the additional deployment cost remains modest while retaining the performance improvements reported in our experiments.

6. Conclusion and future work

In this paper, we acknowledge the limitations associated with the linear assumption in b and the static modeling of users' interests within online recommendation systems. To address these constraints, we introduce an improved Multi-Armed Bandit model named MABID, which can detect the time-varying user interest by applying deep neural networks and an upper confidence bounds strategy. We design an interest drift detection module to dynamically capture the drifts in user interests over time, while extracting informative user features. Moreover, this module reflects underlying non-linear connections between feature information and reward estimation that are more in line with the actual situation. We utilize the UCB strategy to explore the preferences of users and their potential interests, ensuring a comprehensive understanding of user behaviors. Empirical studies on three real-world datasets demonstrate the effectiveness of our proposed method.

In the future, we will further extend MABID in two directions: (1) incorporating embedding layers in lieu of feature engineering to model the user and item features more accurately during the training process, thereby enhancing the representation of user interests and item attributes; (2) leveraging attention mechanisms to learn the diverse user interests and their dynamic evolution over time, thus facilitating the rich characterization of user profiles.

CRediT authorship contribution statement

Huiying Wang: Writing – original draft, Software, Methodology, Conceptualization. **Shen Yang:** Validation, Software, Methodology, Formal analysis, Conceptualization. **Qifeng Zhou:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Formal analysis, Conceptualization. **Qing Wang:** Writing – review & editing, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant No. 62171391).

Data availability

Data will be made available on request.

References

[1] S. Li, A. Karatzoglou, C. Gentile, Collaborative filtering bandits, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2016, pp. 539–548.
[2] J. Kawale, H.H. Bui, B. Kveton, L. Tran-Thanh, S. Chawla, Efficient thompson sampling for online matrix-factorization recommendation, Adv. Neural Inf. Process. Syst. 28 (2015).
[3] B. Lika, K. Kolomvatsos, S. Hadjiefthymiades, Facing the cold start problem in recommender systems, Expert Syst. Appl. 41 (4) (2014) 2065–2073.
[4] L. Song, C. Tekin, M. van der Schaar, Online learning in large-scale contextual recommender systems, IEEE Trans. Serv. Comput. 9 (3) (2016) 433–445, https://doi.org/10.1109/TSC.2014.2365795
[5] S. Pandey, D. Chakrabarti, D. Agarwal, Multi-armed bandit problems with dependent arms, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 721–728.
[6] Q. Wang, C. Zeng, W. Zhou, T. Li, S.S. Iyengar, L. Shwartz, G.Y. Grabarnik, Online interactive collaborative filtering using multi-armed bandit with dependent arms, IEEE Trans. Knowl. Data Eng. 31 (8) (2019) 1569–1580, https://doi.org/10.1109/TKDE.2018.2866041
[7] N. Silva, H. Werneck, T. Silva, A.C. Pereira, L. Rocha, Multi-armed bandits in recommendation systems: a survey of the state-of-the-art and future directions, Expert Syst. Appl. 197 (2022) 116669.
[8] C.J.C.H. Watkins, Learning From Delayed Rewards (Ph.D. thesis), King's College, Cambridge, United Kingdom, 1989.
[9] O. Chapelle, L. Li, An empirical evaluation of thompson sampling, Adv. Neural Inf. Process. Syst. 24 (2011).
[10] P. Auer, Using confidence bounds for exploitation-exploration trade-offs, J. Mach. Learn. Res. 3 (Nov) (2002) 397–422.
[11] X. Xu, H. Xie, J.C.S. Lui, Generalized contextual bandits with latent features: algorithms and applications, IEEE Trans. Neural Netw. Learn. Syst. 34 (8) (2023) 4763–4775, https://doi.org/10.1109/TNNLS.2021.3124603
[12] L. Li, W. Chu, J. Langford, R.E. Schapire, A contextual-bandit approach to personalized news article recommendation, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 661–670.
[13] S. Agrawal, N. Goyal, Thompson sampling for contextual bandits with linear payoffs, in: International Conference on Machine Learning, PMLR, 2013, pp. 127–135.
[14] D. Zhou, L. Li, Q. Gu, Neural contextual bandits with ucb-based exploration, in: International Conference on Machine Learning, PMLR, 2020, pp. 11492–11502.
[15] W. Zhang, D. Zhou, L. Li, Q. Gu, Neural thompson sampling, arXiv preprint arXiv:2010.00827, 2020.
[16] X. Xu, F. Dong, Y. Li, S. He, X. Li, Contextual-bandit based personalized recommendation with time-varying user interests, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 6518–6525.
[17] C. Zeng, Q. Wang, S. Mokhtari, T. Li, Online context-aware recommendation with time varying multi-armed bandit, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 2025–2034.
[18] L.R. Medsker, L. Jain, Recurrent neural networks, Des. Appl. 5 (64–67) (2001) 2.
[19] W. Chu, L. Li, L. Reyzin, R. Schapire, Contextual bandits with linear payoff functions, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.
[20] D.K. Mahajan, R. Rastogi, C. Tiwari, A. Mitra, Logucb: an explore-exploit algorithm for comments recommendation, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 6–15.
[21] Z. Li, M. Liu, X. Dai, J.C. Lui, Towards efficient conversational recommendations: expected value of information meets bandit learning, in: Proceedings of the ACM on Web Conference 2025, 2025, pp. 4226–4238.
[22] Q. Shi, F. Xiao, D. Pickard, I. Chen, L. Chen, Deep neural network with LinUCB: a contextual bandit approach for personalized recommendation, in: Companion Proceedings of the ACM Web Conference 2023, 2023, pp. 778–782.
[23] M. Zhang, T. Nguyen-Tang, F. Wu, Z. He, X. Xie, C.S. Ong, Two-stage neural contextual bandits for personalised news recommendation, arXiv preprint arXiv:2206.14648, 2022.
[24] Y. Ban, Y. Yan, A. Banerjee, J. He, EE-net: exploitation-exploration neural networks in contextual bandits, in: International Conference on Learning Representations, 2021.
[25] Y. Qi, Y. Ban, J. He, Graph neural bandits, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 1920–1931.
[26] L. De Kerpel, D. Benoit, A reward-informed semi-personalized bandit approach for enhancing accuracy and serendipity in online slate recommendations, ACM Trans. Recomm. Syst. 4 (3) (2026) 1–23.
[27] Q. Zhang, D. Wu, G. Zhang, J. Lu, Fuzzy user-interest drift detection based recommender systems, in: 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2016, pp. 1274–1281, https://doi.org/10.1109/FUZZ-IEEE.2016.7737835
[28] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[29] K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: EMNLP, 2014.

[30] J.Y. Yu, S. Mannor, Piecewise-stationary bandit problems with side observations, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 1177–1184.
[31] N. Hariri, B. Mobasher, R. Burke, Adapting to user preference changes in interactive recommendation, in: Proceedings of the 24th International Conference on Artificial Intelligence, 2015, pp. 4268–4274.
[32] Q. Wu, N. Iyer, H. Wang, Learning contextual bandits in a non-stationary environment, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 495–504.
[33] Q. Wu, H. Wang, Y. Li, H. Wang, Dynamic ensemble of contextual bandits to satisfy users' changing interests, in: The World Wide Web Conference, 2019, pp. 2080–2090.
[34] C. Shen, X. Zhang, W. Wei, J. Xu, HyperBandit: contextual bandit with hypernewtork for time-varying user preferences in streaming recommendation, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 2023, pp. 2239–2248.
[35] X. Wu, S. Cetintas, D. Kong, M. Lu, J. Yang, N. Chawla, Learning from cross-modal behavior dynamics with graph-regularized neural contextual bandit, in: Proceedings of the Web Conference 2020, 2020, pp. 995–1005.
[36] J. Langford, T. Zhang, The epoch-greedy algorithm for contextual multi-armed bandits, in: Proceedings of the 20th International Conference on Neural Information Processing Systems, 2007, pp. 817–824.
[37] S. Bubeck, N. Cesa-Bianchi, et al., Regret analysis of stochastic and nonstochastic multi-armed bandit problems, Found. Trends® Mach. Learn. 5 (1) (2012) 1–122.
[38] Y. Lin, Y. Liu, F. Lin, L. Zou, P. Wu, W. Zeng, H. Chen, C. Miao, A survey on reinforcement learning for recommender systems, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–21, https://doi.org/10.1109/TNNLS.2023.3280161
[39] T. Lattimore, C. Szepesvári, Bandit Algorithms, Cambridge University Press, 2020.
[40] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[41] S. Xu, BanditMF: multi-armed bandit based matrix factorization recommender system, arXiv preprint arXiv:2106.10898, 2021.
[42] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using clickthrough data, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, 2013, pp. 2333–2338.
[43] L. Li, W. Chu, J. Langford, T. Moon, X. Wang, An unbiased offline evaluation of contextual bandit algorithms with generalized linear models, in: Proceedings of the Workshop on on-Line Trading of Exploration and Exploitation 2, JMLR Workshop and Conference Proceedings, 2012, pp. 19–36.
[44] R. Singh, F. Liu, Y. Sun, N. Shroff, Multi-armed bandits with dependent arms, Mach. Learn. 113 (1) (2024) 45–71.
[45] M. Rawson, R. Balan, Convergence guarantees for deep epsilon greedy policy learning, arXiv preprint arXiv:2112.03376, 2021.
[46] E. Austin, L.E. Morgan, Detecting changes and anomalies in nonstationary contextual bandits with an application to task categorisation, Inf. Sci. 717 (2025) 122270.

Author biography

Huiying Wang is currently a master's student in the Department of Automation, Xiamen University, Xiamen, China. Her research interests include recommender systems and multi-armed bandit algorithms.

Shen Yang is currently a master's student in the Department of Automation, Xiamen University, Xiamen, China. His research focuses on recommender systems and multi-armed bandit algorithms.

Qifeng Zhou is currently a Professor with the Department of Automation, Xiamen University, Xiamen, China. Her research interests include machine learning, data mining, recommendation systems, and Large Language Models.

Qing Wang is currently an honorary research associate, Department of Statistics and Actuarial Science, Hong Kong University, Her research interests include Large-Scale Data Mining. Multi-armed Bandits, Large Language Models, and Reinforcement Learning.