Stability-aware Preference Modeling for Sequential Recommendation

CHAOYONG WEI and WENJUN JIANG*, Hunan University, China
KENLI LI, Hunan University, China
JIE WU, Temple University, USA

Many researchers primarily rely on modeling user interests for sequential recommendation. However, dynamic user behaviors often accompany unstable interaction histories, and modeling user interests alone is insufficient for comprehensive user features. Some studies notice the problem of interest drift, but they are usually limited to modeling at the item-level, unable to perceive the subtle changes at the feature-level. To this end, we propose a Stability-aware Preference Model (SAPM), which consists of three modules. The LSI module for extracting long and short-term interests, the FLC module for extracting feature-level candidate information, and the SAF module for fusing them according to the stability score. In particular, we propose a Multi-head GRU (MHGRU) structure in the LSI module, which is more efficient than the general GRU and has stronger expression ability. Through extensive experiments, our framework shows significant mitigation of the impact of unstable interactions. On the two real data sets, we improve MRR by 5.7% and 14.0% compared with the recent baselines. Moreover, we conduct an in-depth analysis of user interaction stability and obtain several interesting findings that can benefit future studies.

CCS Concepts: • **Information systems → Information systems applications**; Data mining; Recommender systems.

Additional Key Words and Phrases: sequential recommendation, preference modeling, stability-aware

ACM Reference Format:
Chaoyong Wei, Wenjun Jiang, Kenli Li, and Jie Wu. 2025. Stability-aware Preference Modeling for Sequential Recommendation. *ACM Trans. Internet Technol.* , (2025), 25 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

User interest modeling is an essential part of recommendation models. It aims to capture user preferences to produce proper recommendations. Some studies [10, 61] focus on modeling long-term interests but overlook the temporal dynamics of user interests. With the widespread adoption of sequence models (e.g., GRU [7]), [3, 11, 13, 14, 45, 60] start extracting short-term interests and achieve some research outcomes. To comprehensively capture the evolution of user interests and changes in behavior, some researchers amalgamate the two and propose Long and Short-Term Interest recommendation models [41, 55, 59].

While existing models improve recommendation effectively, they still face three main challenges. First, interest drift has not been fully studied. Second, the modeling is usually at the item level or explicit aspect level, and the information on fine-grained dimensions, such as implicit feature level, hasn't been well explored. Third, existing methods either

*Wenjun Jiang is the Corresponding Author.

Authors' addresses: Chaoyong Wei, 1225305141@qq.com; Wenjun Jiang, jiangwenjun@hnu.edu.cn, Hunan University, College of Computer Science and Electronic Engineering, Changsha, China, 410082; Kenli Li, Hunan University, College of Computer Science and Electronic Engineering, Changsha, China, 410082, lkl@hnu.edu.cn; Jie Wu, Temple University, Department of Computer and Information Sciences, Philadelphia, USA, jiewu@temple.edu.

use a sequential structure which lacks expression ability, or quadratic computational complexity structures such as graph-based and transformer-based which are usually quite complex [19].

For the first challenge, some studies focus on the interest drift issue in the sequential recommendation, treating it as uncertainty or noise. Lin et al. [30] propose a modeling approach to integrating global and local preferences, succinctly addressing user intent uncertainty. Zhang et al. [57] devise a hierarchical sequence denoising model to eliminate noise items, while [32, 56] employ soft denoising techniques, allocating lower weights to reduce the impact of noises. However, denoising may compromise semantic information, weaken implicit feature signals, and destabilize sequences. Recent research [8, 9, 17, 44] employ Gaussian distribution modeling method for interaction sequences to counteract uncertainty. However, Gaussian-based studies have not explored interest drift from the perspective of stability. Complex user interaction behavior may lead to sequence instability. If the entire interaction history is regarded as the user's interest, it may lead to performance degradation. Meanwhile, most of these studies only focus on modeling at the item level but fail to perceive the subtle changes at the feature level. In fact, the problem of interest drift can be regarded as the stability of item-level and feature-level in the interaction history.

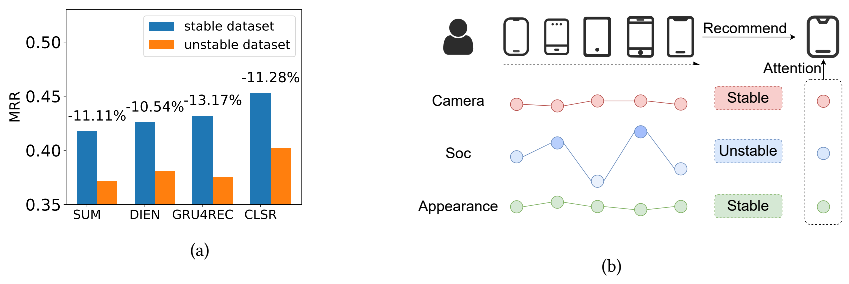


Fig. 1. (a): The negative effects of unstable sequences. (b): An illustrative example of fine-grained features.

To check the impact of unstable sequences on recommendation, we divide the *Taobao* dataset into two equal parts (Please refer to Section 4.1 for the processing): stable dataset and unstable dataset. Then, we use them as different testing data, along with the same training data, to train and test three representative models (i.e., SUM [25], DIEN [60], GRU4REC [11], and CLSR [59]). Figure 1(a) shows that existing models perform better on stable sequences (i.e., higher MRR), which validates the negative effects of unstable sequences.

As for the second challenge of fine-grained modeling, existing methods usually rely solely on item-level or explicit aspect-level interest modeling. Taking smartphones as an example, smartphones possess multiple attributes such as appearance, Soc (System on Chip), camera, and more. Modeling based solely on item-level interest results in embeddings that blend information from various attributes, thus failing to effectively capture the relationships between these attributes. Meanwhile, some researchers (e.g., Chin et al. [6]) model aspect level explicit interest, but in reality, the data set lacks enough labels to train. Considering this, Lin et al. [31] proposes implicit feature-level modeling and uses a routing mechanism to simulate the evolution of user behavior. Chen et al. [4] introduce feature-level memory channels to model users in more detail. They achieve good results in fine-grained modeling, but they still regard all learning as interest, without considering interest drift. The interaction history of some users may not be completely dominated by interest, because they may be exploring fresh items.

We provide an illustrative example in Figure 1(b), where a user clicks on five mobile phones in sequence. When evaluating the situation solely at the item level, we can only see the five devices and ignore deeper insights into the user preferences. By analyzing the smartphones' fine-grained features in terms of camera quality, Soc performance, and appearance level, we can see the stability of users' deep interest. Figure 1(b) shows the scores of the three features of the five mobile phones browsed by the user. It can be found that the Camera score and Appearance score of the five mobile phones are high and in a stable state, while the score of Soc fluctuates significantly. This indicates the importance of capturing fine-grained user interest stabilities and dealing with them carefully. Lin et al. [31] emphasizes the effectiveness of implicit feature-level modeling over explicit modeling. Considering the limited availability of real training data, we focus on interest drift at the implicit feature level.

As for the third challenge of the modeling technique, $RNNs$ (i.e., RNN, GRU, LSTM, etc.) is an efficient linear structure but with limited expression ability. Therefore, DIEN [60] and CLSR [59] employ dual $RNNs$ structures, which achieve good results. However, as the sequence length increases, the time required also increases linearly, making it unsuitable for scalability. Graph-based and Transformer-based recommendation models have quadratic computational complexity. To address this issue, AutoMLP [19] proposes a linear MLP model. Nevertheless, this model lacks the sequential relationship of $RNNs$. Thus there is an urgent need for a simple and efficient sequential model.

In this paper, we strive to address the above three challenges. Firstly, we find that interest drift can be transformed into a sequence stability problem and demonstrate that unstable sequences indeed impact interest recommendations (as shown in Fig. 1(a)). We also analyze interest drift at the feature level. Based on this, we propose a novel framework called Stability-aware Preference Modeling (SAPM). It comprises three modules: Long and Short-Term Interests (LSI), Feature-Level Channels (FLC), and Stability-aware Fusion (SAF). The LSI module effectively captures both the long-term and short-term interests of users, providing a simplified interest representation. The FLC module extracts feature-level information from interaction history to supplement the LSI module. The SAF module assesses sequence stability and exploits the stability score to control the degree to which candidate feature-level information supplements the long and short-term interests. In particular, to address the modeling challenge, we propose the MHGRU structure in LSI module, which exhibits parallelism and efficiency compared to traditional $RNNs$ models, while retaining the sequential nature of $RNNs$. It can reduce model inference time. Finally, we conduct extensive experiments to validate the effectiveness of the SAPM framework and conduct in-depth analyses of user interaction stability, providing novel insights into the sequential recommendation.

The contributions of this research can be summarized as follows:

- We discover that interest drift can be transformed into a sequence stability problem and demonstrate that unstable sequences indeed impact interest recommendations. Additionally, we analyze interest drift at the feature level.
- We propose a simple and efficient SAPM framework, which effectively extracts long and short-term interests while measuring interest drift through stability perception, subsequently complementing relevant feature-level information.
- To validate the effectiveness of the SAPM framework, we conduct extensive experiments and in-depth analyses of user interaction stability, thereby providing novel insights into the field of sequential recommendation.

The remainder of the paper is organized as follows. Section 2 describes the problem we address. The details of our SAPM framework are presented in Section 3. Extensive experiments with two real-life datasets are presented and analyzed in Section 4. The related work is discussed in Section 5. Section 6 concludes this paper and highlights future directions.

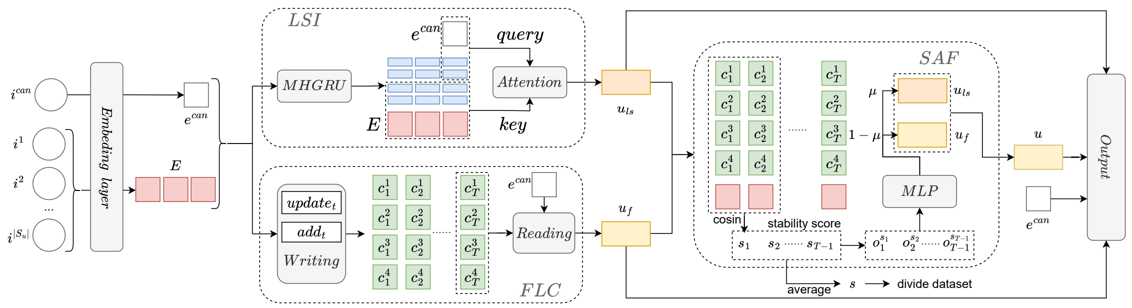


Fig. 2. The overview of the SAPM framework. The LSI module generates the long and short-term interests u_{ls}, and the FLC module obtains the feature-level information u_f, which is fused to u by the SAF module according to the stability score.

2 PROBLEM FORMULATION

In sequential recommendation tasks, there is a set of users U and a set of items I. The interaction sequence of a user $u \in U$ can be described as $S_u = \left[i^1, \ldots i^n, \ldots, i^{|S_u|}\right]$, where $i^n \in I$ represents the item of the user's n-th interaction, with a total of $|S_u|$ interactions, and the candidate item is i^{can}. Different user sequences exhibit varying degrees of stability. We denote the embeddings of S_u and i^{can} as $E = \left[e^1, \ldots, e^n, \ldots, e^{|S_u|}\right]$ and e^{can}, respectively, where $e^n, e^{can} \in \mathbb{R}^D$, $E \in \mathbb{R}^{|S_u| \times D}$, and D is the dimension. The goal of sequential recommendation is to predict the next item $i^{|S_u|+1} \in I$ that the user may interact with the known interaction sequence S_u. Our specific task can be expressed as:

$$P\left(i^{|S_u|+1} = i|S_u\right). \tag{1}$$

Owing to the negative impact of unstable sequences on recommendation quality, our objective is to maximize the probability of prediction P in Eq. 1. In the actual experiments, the maximum sequence length is T. If the sequence length L is less than T, it is padded on the left of T. If the sequence length L is greater than T, the most recent T items are used for the experiment.

We define the cosine similarity between e^n and e^{n+1} for $n, n+1 \in |S_u|$ as the local stability of the sequence. A higher similarity score indicates that two adjacent items are more similar and thus more stable. The average similarity score of the entire sequence represents the overall stability of the sequence.

3 THE SAPM FRAMEWORK

Our SAPM framework consists of three main modules (Figure 2): the LSI module for extracting long and short-term interests, the FLC module for extracting feature-level candidate information, and the SAF module for fusing them according to stability score.

3.1 Long and Short-Term Interests (LSI)

When it comes to interest recommendation, long and short-term interest modeling is widely used. Among these, SLi-Rec [55] and CLSR [59] are two recent representative models. We design the long and short-term interests (LSI) module, which enhances the expression ability and robustness of the model while maintaining the linear complexity. The module includes our proposed Multi-head GRU (MHGRU) and Attention pool, which can effectively extract users' long and short-term interests. To better illustrate our design, we compare the structures of SLi-Rec, CLSR, and our LSI in Figure 3.

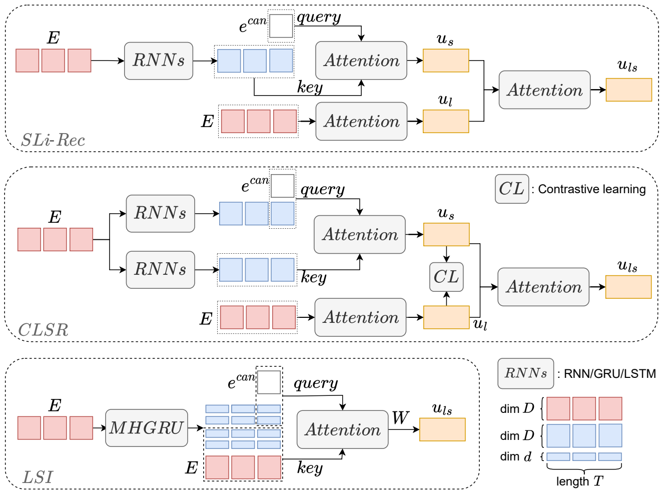


Fig. 3. Comparison of SLi-Rec, CLSR, and our LSI for extracting long and short-term interests. The first two need to generate u_s and u_l and then merge them into u_{ls}, while we generate u_{ls} at one time.

3.1.1 MHGRU.

3.1.1 MHGRU. Many researchers use *RNNs* to extract short-term interests because this structure can capture context information. However, compared to the transformer-based and the graph-based neural network, the effect of GRU with linear complexity is inferior. SLi-Rec utilizes an *RNNs* to extract interests and CLSR uses a dual *RNNs* structure to extract short-term interests, as shown in Figure 3. Although it achieves good performance, the efficiency of dual *RNNs* is low because general *RNNs* cannot be executed in parallel. Inspired by this and taking advantage of the Multi-head attention mechanism, we propose MHGRU to effectively aggregate several GRU into one, thus reducing the parameter quantity and improving the parallel efficiency. MHGRU incorporates multiple heads to offer diverse perspectives, with each head positioned differently in the sequence. This allows for independent viewpoints on short-term interests, enhances parallel processing, and improves the expression ability of the model.

First, let's review GRU's formula. GRU is a cyclic neural network structure optimized relative to LSTM, which contains only two gating units. In GRU, hide status h_t is used to transmit status information, but each GRU unit can only transmit one status information, which limits its expression ability to a certain extent.

$$z_t = \sigma(W_z \cdot [h_{t-1}, e_t]), \tag{2}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, e_t]), \tag{3}$$

$$\hat{h} = \tanh(W \cdot [r_t * h_{t-1}, e_t]), \tag{4}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h}. \tag{5}$$

Our MHGRU implementation is as follows. The status group at time t can be expressed as:

$$H_t = \left[h_t^1, h_t^2, \ldots, h_t^M\right], \tag{6}$$

where $H_t \in \mathbb{R}^{M \times d}$ contains states of M heads at time t and d represents the dimension of h_t^*. We use a trainable matrix W_z to generate an update gate group:

$$Z_t = \sigma(W_z \cdot [H_{t-1}, e_t]), \tag{7}$$

where σ is the sigmoid function and Z_t is used to control whether each head of the input information at the current time needs to be updated to the hidden state. It uses the input e_t and the hidden state group H_{t-1} is used as input and a sigmoid function is used to generate a value Z_t. When Z_t is close to 1, it indicates that the input information needs to be updated. Then we use a trainable matrix $W_r \in \mathbb{R}^{M\times d}$ to generate reset gate group:

$$R_t = \sigma(W_r \cdot [H_{t-1}, e_t]), \tag{8}$$

where R_t determines how much information from the previous time step hidden state group H_{t-1} should be retained. It also takes the e_t and the hidden state group H_{t-1} as input and a sigmoid function is used to generate a value R_t.

The candidate hidden state $\hat{h}$ in Eq. 4 in GRU is a temporary hidden state calculated based on the reset gate r_t and the current time step input e_t. It cannot directly be applied to Multi-head in MHGRU, so we design the independent candidate hidden states $\hat{H}$, which is expected to provide separate candidate states for each attention head. The hidden state updates of each head no longer depend on the same state $\hat{h}$; instead, they gather information from their respective independent candidate states. We use the trainable parameter matrix $W_{\text{alone}} \in \mathbb{R}^{(M\times d+D)\times(M\times d)}$ to obtain independent candidate state group:

$$\hat{H} = \tanh(W_{alone} \cdot [R_t \cdot H_{t-1}, e_t]), \tag{9}$$

where $\hat{H} = \left[\hat{h}^1, \hat{h}^2, \ldots, \hat{h}^M\right]$ contains the information of the previous time step reserved by each head. This design enables each head to independently acquire and process its own candidate states and improves the modeling ability of the model for the input sequence. We then update the final hidden state group H_t:

$$H_t = (1 - Z_t) \cdot H_{t-1} + Z_t \cdot \hat{H}. \tag{10}$$

The output of MHGRU can be expressed as:

$$\begin{aligned} MHGRU\left(e^1, e^2, \ldots, e^T\right) &= \left[H_1, H_2, \cdots, H_T\right] \\ &= \begin{bmatrix} h_1^1 & h_1^1 & \ldots & h_1^M \\ h_2^1 & h_2^2 & \ldots & h_2^M \\ \vdots & \vdots & \vdots & \vdots \\ h_T^1 & h_T^2 & \ldots & h_T^M, \end{bmatrix} \end{aligned}$$

which includes $M \times T$ small states output by M heads at T times.

Our design for MHGRU involves implementing GRU in a multi-head parallel manner. This approach can enhance execution efficiency as well as improve the overall performance. RNN, LSTM, and GRU share similar model structures, theoretically making it possible to design MHRNN and MHLSTM using the same principle. However, specific designs may vary with model structures, such as carefully determining which gating mechanism to parallelize. We chose GRU as the foundation because of its relatively simple structure and strong expression ability.

3.1.2 *Attention pool.*

Existing researches [1, 55, 58, 59] usually extract the long-term interest u_l and short-term u_s interest separately, and then fuse them to generate the final interest representation u_{ls}. Different from these methods, our design which obtains them simultaneously is more concise as shown in Figure 3, because long-term and short-term interest candidates are in the same semantic space. We use the attention pool to consider both long-term and short-term interest candidates through one-time query and generate the final interest representation. This design not only reduces the steps but also realizes interactive learning between long-term and short-term interest candidate states.

Similar to SLi-Rec [55] and CLSR [59], we also use the query embedding table and the output of $RNNs$ to capture different dynamics information over time and use them as query vectors. Different from them, we improve the modeling process of long and short-term interests in two aspects. First, our $RNNs$ is a Multi-head GRU. We choose m (where $m < M$) heads to extract the information from the query vector. Multi-head output can provide more independent query perspectives. According to the needs of different scenarios, different numbers of head channels can be selected to provide a wider selection range. Second, we use a unified query vector to capture the user's final interest expression u_{ls} at once. There will be no long-term interest u_l and short-term interest u_s in the middle steps, because u_{ls} integrate the characteristics of long and short-term interest, reducing the steps of fusion. The details are as follows.

The $query$ vector Q is expressed as:

$$Q = \left[h_T^1, h_T^2, \dots, h_T^m, e^{can}\right], \tag{11}$$

which includes the final states of m heads and candidate item e^{can}.

Similar to CLSR [59], we also use the outputs of $RNNs$ as the key for the attention pooling to effectively model the temporal dependencies of sequential data and capture the evolving relationships of short-term interests. We still employ the partial outputs of the Multi-head GRU model as representations of short-term candidate interests, where each head provides a unique and distinct perspective on interest evolution. When it comes to capturing long-term interests, we need to consider long-term candidate interests as well. Long-term interests represent the interests that remain relatively stable over time and are less influenced by factors t. Therefore, we use the encoded vectors of the historical sequences embedding $E = \left[e^1, e^2, \dots, e^T\right]$ as the features for long-term interest candidates. We only have $M - m$ heads to use on key and our candidate key can be represented as K:

$$K = \begin{bmatrix} h_1^{m+1} & h_1^{m+2} & \dots & h_1^M & e^1 \\ h_2^{m+1} & h_2^{m+2} & \dots & h_2^M & e^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ h_T^{m+1} & h_T^{m+2} & \dots & h_T^M, & e^T \end{bmatrix}, \tag{12}$$

which includes both long and short-term candidate states. We use an MLP layer to learn the similarity scores ρ_t between the query vector Q and the key vector K. We perform the query operation:

$$V = W_k K, \tag{13}$$

$$\{a_1, a_2, \dots, a_T\} = \text{MLP}(V||Q||(V - Q)||(V \cdot Q)) \tag{14}$$

$$\rho_t = \frac{exp(a_t)}{\sum_{i=1}^{T} exp(a_i)}, \tag{15}$$

where W_k is a transformation matrix and ρ_t represents the score at time t.

With Eq. 15, we obtain the score vector $A = \text{concat}\left(\rho_1, \rho_2, \dots, \rho_T\right)$. Then, the final interest representation u_{ls} can be expressed as:

$$u_{ls} = W_{to}(A \cdot K) \tag{16}$$

where W_{to} is a transformation matrix that converts the result to D dimensions, and u_{ls} contains the long and short-term interest information of the user.

3.1.3 Complexity. We analyze the complexity of MHGRU and M GRU models, supposing a sequence length of T. Since GRU is linear and is difficult to parallelize, the complexity of executing M GRU models is $O(M \times T)$. We set the number of heads in MHGRU to M, which can produce outputs of the same size as M GRU models, but with an approximate complexity of $O(T)$. This is because we only need to execute the $RNNs$ once, thus improving parallel efficiency. It is important to note that as M increases, the number of parameters also increases, which may introduce some additional time overhead. The impact of this additional overhead is influenced by GPU parallel computing. In real scenarios, the complexity of MHGRU falls in the $(O(T), O(M \times T))$. In summary, compared to DIEN [60] and CLSR [59] models, our MHGRU structure is more efficient, because those two models require executing two regular $RNNs$, such as GRU or other $RNNs$.

3.2 Feature-Level Channels (FLC)

Considering interests alone may not effectively capture the overall representation of the user, as user interests tend to drift over time. To model users more accurately, we need to consider multiple dimension features other than just interests, particularly user stability. When a user's choices of items have been unstable in past interactions, it may indicate that the user is exploring new items. Treating this exploration process as interest learning would be unreasonable. Furthermore, the feature-level features also matter, we need to consider fine-grained feature-level information beyond interests. Inspired by RUM [4] and SUM [25], we use memory networks to store feature-level information, which may reflect users' hidden and subtle preferences in interacting.

In this section, we propose the FLC memory network (As shown in Figure 2). Each channel represents a dimension of the low-dimensional features of the item. We update and utilize the memory network by writing and reading. As described in Figure 1(b), a series of user interactions can be transformed into feature-level dimensions such as Soc, appearance, and camera. However, many features are implicitly hidden, and we do not have ready-made explicit labels to train them. Therefore, we exploit the reading and writing operation methods of RUM and SUM to obtain multi-dimensional implicit features and make improvements and simplifications.

We use N channels as the memory matrix, denoted as $C = \{c_t^1, c_t^2, \ldots, c_t^N\}$, where c_t^n represents the embedding representation of feature n updated after the user interacting with the t-th item. Since each channel is also a hidden state, we set the dimension as d for ease of searching for optimal parameters. Meanwhile, we have a global feature map $F^w = \{f_1^w, f_2^w, \ldots, f_N^w\}$ for writing operations and $F^r = \{f_1^r, f_2^r, \ldots, f_N^r\}$ for reading operations. The details are the follows.

3.2.1 *Writing operation.*

The writing operation is to write the feature-level information of interaction history into the memory network. When the user interacts with the t-th item, the attention scores of e_t for each feature-level channel can be calculated as:

$$w_{tn}^w = e_t \cdot f_n^w, \tag{17}$$

$$z_{tn}^w = \frac{exp(\beta w_{tn}^w)}{\sum_j^N exp(\beta w_{tj}^w)}, \forall n \in \{1, 2, \ldots, N\} \tag{18}$$

where β represents the scaling factor, and z_{tn}^w denotes the attention score, which represents the weight of each item in each feature dimension at the t-th time step. Based on these scores, we synthesize the candidate memory state $\hat{c}$. In SUM [25], an additional Highway Channel is introduced to enhance the evolution of sequential relationships. Since we capture the interest evolution with our MHGRU structure, we delete the Highway Channel to simplify the network structure and operations. Therefore, the synthesis of $\hat{c}$ is as follows:

$$\hat{c} = \sum_n z_{tn}^w \cdot c_{t-1}^n. \tag{19}$$

We propose a simpler structure compared to RUM [4] and SUM [25], which achieves the same effects. Both SUM and RUM employ the operations "add," "erase," and "reset" to update the memory state. However, we notice redundancy in the functionality of the "erase" and "reset" operations. Additionally, since $\hat{c}$ already retains important memories from the previous time step, we introduce the operations "update" and "add" for memory updates. The new memory state is influenced by both the input of the new interaction and the past memory. The calculation is as follows:

$$\text{update}_t = \sigma(W_{update}\,[e_t, \hat{c}]), \tag{20}$$

$$\text{add}_t = tanh(W_{add}\,[e_t, \hat{c}]), \tag{21}$$

where W_{update} and W_{add} are trainable weight matrices. Next, we update memory as follows:

$$c_t^n = \text{update}_t \cdot \text{add}_t + (1 - \text{update}_t) \cdot c_{t-1}^n, \tag{22}$$

where c_t^n represents the updated state of the n-th feature-level memory after the user's t-th interaction.

3.2.2 Reading operation. The reading operation is to read feature-level information from the memory network. For the reading operation, we also utilize the global feature map $F^r = \left\{f_1^r, f_2^r, \ldots, f_N^r\right\}$ to obtain the attention scores for each channel:

$$w_{tn}^r = e^{can} \cdot f_n^r, \tag{23}$$

$$z_{tn}^r = \frac{exp(w_{tn}^r)}{\sum_j^N exp(w_{tj}^r)} . \forall n \in \{1, 2, \ldots, N\} \tag{24}$$

Then, we use the attention scores z_{tn}^r to weight the feature-level memory channels for reading and obtain the user's candidate feature u_f:

$$u_f = \sum_n z_{tn}^r \cdot c_T^n, \tag{25}$$

where u_f represents the aggregated feature-level memory from the user's interaction history and can serve as a complement for recommendation. It helps mitigate the influence of the user's evolving interests when exploring items.

3.3 Stability-aware Fusion (SAF)

The SAF module regulates the extent to which candidate feature-level information complements the long and short-term interests according to the stability score. By considering the stability of user interactions, we can better understand user behavior patterns and preferences, providing more accurate user representations for personalized recommendations and other tasks.

To obtain the stability score of the user sequences, we calculate the similarity between each pair of consecutive interactions using cosine similarity, to measure the distance between the user's new interaction and past ones. Based on this distance, we can ascertain whether the user's interactions are stable or unstable, thereby inferring whether they are consistently engaging with interested items or exploring new ones. If it is the latter case and the overall interaction stability is low, the computed overall similarity will also be lower. In such cases, we need to select some candidate feature memories that go beyond the user's interest and recommend them to the user, providing slightly surprising recommendations while still within the user's acceptance range of personalization.

We take into account the stability at both the item level and feature-level. The similarity score can be represented by calculating the cosine similarity between the embedded features and the channel features of two consecutive moments:

$$s_{t-1} = \text{cosine}\left(\left[c_{t-1}^1, c_{t-1}^2, \ldots, c_{t-1}^N, e_{t-1}\right], \left[c_t^1, c_t^2, \ldots, c_t^N, e_t\right]\right), \tag{26}$$

where s_{t-1} is the cosine similarity score between $t-1$ and t moment. We then input this score into the following formula to obtain the stability score μ:

$$\mu = \sigma\left(\text{MLP}\left(o_1^{s_1}, o_2^{s_2}, \ldots, o_{T-1}^{s_{T-1}}\right)\right), \tag{27}$$

where o_t are trainable parameters of dimension 1. Since there are $T-1$ intervals among T interactions, and each position has different importance, we set $T-1$ individual o_t parameters instead of a single unified base. $o_t^{s_t}$ represents the influence of stability at that position on the user's final representation learning, given the similarity score s_t. μ is the final stability score, which is exploited as the weight of LSI, and we perform weighting accordingly:

$$u = (1-\mu)\cdot u_f + \mu \cdot u_{ls}, \tag{28}$$

where u is the final user representation considering the stability of historical interactions. The insights of Eq. 28 are that the stable users take more weight on their interests, while unstable users take more weight on the implicit multi-dimensional features.

3.4 Output Layer

In this section, we introduce the final MLP output layer and loss function. To fully utilize the extracted user interest features, we not only input the fused interest representation u into the prediction layer but also consider the interest features before fusion. Although the fused interest representation has strong expression ability, the gating network may lose some feature information. Users' interests may drift, and although interest fused with stability features can reduce the impact of unstable sequences, there is still a particular degree of uncertainty. Therefore, we input both pre- and post-fusion interest features into the prediction layer, allowing the MLP to adaptively select, as follows:

$$\hat{y} = Predict(u_{ls} \odot e^{can} \| u_f \odot e^{can} \| u \odot e^{can} \| e^{can}), \tag{29}$$

where $\hat{y}$ represents the output score of the prediction layer, and $\odot$ denotes the Hadamard product.

We combine the three interest features with the candidate item using the Hadamard product to enhance the model's ability to capture the interactions between features. The optimization objective is as follows:

$$\mathcal{L} = -\frac{1}{|G|}\sum_{i=1}^{|G|}\left(y_i log\hat{y}_i + (1-y_i)log(1-\hat{y}_i)\right) + \lambda\|\Phi\|_2, \tag{30}$$

where Φ represents the set of trainable parameters, G is the training set, $|G|$ is the number of instances in the training set, λ controls the penalty strength, and $\lambda\|\Phi\|_2$ is the regularization term.

4 EXPERIMENTS

In this section, we conduct extensive experiments to validate the effectiveness of our SAPM framework. We first compare the model's performance with representative baselines. Then we conduct the ablation study on SAPM to evaluate the role of each component. Next, we perform sensitivity analysis on model parameters. We also analyze the role of the three modules of SAPM and the relationship between stability and user interaction behavior. Finally, we test the complexity of our MHGRU structure. We will share our codes upon the acceptance of this paper.

4.1 Experiment Setup

Datasets: We employ two real datasets of different platforms, *Taobao* and *Amazon*. The *Taobao* dataset is derived from user behavior sequence data on the Taobao e-commerce platform. Following the processing methods in [2, 3, 59], we select data extraction sequences from November 25, 2017, to December 3, 2017. The data in the first seven days serve as the training set, with the following two days as the validation set and test set, respectively. The *Amazon* dataset is chosen from Amazon Movies and TV files, where we process data from the last ten years and take the data within the 200 most recent days. The first 100 days serve as the validation set and the latter 100 days as the test set, with data from other periods as the training set. The statistical information for the two processed datasets is presented, as shown in Table 1.

Table 1. Statistics from the *Taobao* and *Amazon* datasets.

Dataset	Users	Items	Instances	Avg. Length	Sparsity
Taobao	37283	65217	1517912	40.71	99.94%
Amazon	33326	21901	958986	28.78	99.87%

Partition of Stable and Unstable Datasets: By training the SAPM model and reaching the convergence saturation stage, we obtain the weights of the model. We use these weights to recompute the testing set in the *Taobao* dataset and *Amazon* dataset. For each sequence of length T, we calculate the average of $T-1$ cosine similarity distance scores s_t in Eq. 26, denoted as s, representing the stability score of a sequence in the testing set. Next, we sort these scores in descending order, taking the top 50% scores (high scores) as the stable testing dataset and the bottom 50% scores (low scores) as the unstable testing dataset. We use the same training set and different test sets to verify the impact of unstable sequences.

Evaluation Metrics: We use three common metrics in sequential recommendation: MRR, NDCG, and HIT, and consider top-k recommendation ($k \in \{2, 4, 6\}$). All metrics are the higher the better.

Baselines: We carefully select different interest modeling methods as baselines. As our comparative research focuses on model structures, we mainly implement the baselines according to their original structure design. It is worth noting that most of the current recommendation models adopt empirical risk minimization (ERM) as the learning framework, which assumes that the training and testing data share the same distribution, neglecting the distribution drift between the two. Yang et al. [53] proposes a Distributionally Robust Optimization mechanism for Sequential recommendation (DROS) framework to enhance model robustness. In future work, it is promising to apply DROS to recommendation models.

- NCF [10]: NCF is a neural recommendation model for predicting user interactions using generalized matrix factorization.
- SASRec [16]: SASRec uses self-attention to balance the advantages of Markov Chains and $RNNs$, achieving good performance on both sparse and complex data.
- DIN [61]: DIN employs an attention mechanism to aggregate user interest representations from user historical behaviors.
- SUM [25]: SUM utilizes multiple channels to model users' multiple interests, compensating for the inadequacy of single embedding representations.
- Caser [45]: Caser embeds interaction sequences into images and then extracts local image features using convolutional networks to capture interests.

- GRU4REC [11]: GRU4REC applies GRUs to evolve user session sequences, extracting user interests.
- DIEN [60]: DIEN improves upon DIN by designing an interest extraction layer to capture temporal interests from historical behavior sequences.
- SLi-Rec [55]: SLi-Rec combines the attention framework with LSTM to extract long and short-term interests using time-awareness.
- SURGE [3]: SURGE uses graph neural networks to model interaction sequences, extracting actual interests from implicit and noisy signals.
- CLSR [59]: CLSR distinguishes long-term and short-term interests based on long and short-term modeling by leveraging contrastive learning.

Parameter Settings: We conduct experiments under the Microsoft Recommenders framework, setting the learning rate to 0.001. The maximum training length for *Taobao* and *Amazon* dataset is set to $T = 50$ and $T = 40$, respectively. The prediction layer hidden layers use feedforward neural networks with dimensions of [80, 40]. The ratio of positive to negative examples in the validation set is 1:4, and in the test set, it is 1:99, with the batch size being set to 500. We set λ to 1×10^{-6}.

4.2 Comparison study

Table 2. Comparison of the baselines and the SAPM. The best performance is indicated in bold, and the second-best performance is indicated in underlining. The *Imp.* means the percentage of improvement of the best compared with the second best.

Dataset	*Taobao*							*Amazon*						
Method/*Imp.*	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6
NCF	0.1417	0.0802	0.1128	0.1337	0.0963	0.1661	0.2222	0.1172	0.0631	0.0893	0.1072	0.0759	0.1322	0.1802
SASRec	0.2968	0.2280	0.2871	0.3166	0.2647	0.3907	0.4700	0.2047	0.1409	0.1850	0.2096	0.1658	0.2601	0.3260
DIN	0.3039	0.2284	0.2923	0.3263	0.2651	0.4014	0.4925	0.2428	0.1746	0.2246	0.2524	0.2034	0.3099	0.3846
SUM	0.3985	0.3392	0.3971	0.4230	0.3819	0.5049	0.5742	0.2541	0.1889	0.2372	0.2630	0.2185	0.3213	0.3905
Caser	0.3509	0.2886	0.3424	0.3696	0.3271	0.4415	0.5144	0.2057	0.1409	0.1843	0.2091	0.1656	0.2581	0.3248
GRU4REC	0.4043	0.3478	0.4017	0.4265	0.3893	0.5034	0.5701	0.2370	0.1711	0.2179	0.2440	0.1983	0.2981	0.3681
DIEN	0.3991	0.3380	0.3964	0.4244	0.3803	0.5041	0.5791	0.1982	0.1371	0.1750	0.1979	0.1592	0.2402	0.3017
SLi-Rec	0.3604	0.2963	0.3546	0.3834	0.3385	0.4624	0.5397	0.2718	0.2041	0.2567	0.2854	0.2364	0.3484	0.4253
SURGE	0.4260	0.3670	0.4250	0.4519	0.4099	0.5331	0.605	0.2723	0.2039	0.2553	0.2839	0.2352	0.3446	0.4213
CLSR	0.4312	0.3727	0.4307	0.4581	0.4164	0.5396	0.6129	0.2483	0.1798	0.2296	0.2577	0.2089	0.3150	0.3905
SAPM	**0.4558**	**0.4006**	**0.4583**	**0.4844**	**0.4475**	**0.5702**	**0.6399**	**0.3105**	**0.2416**	**0.2968**	**0.3270**	**0.2768**	**0.3943**	**0.4755**
Imp.	5.7%	7.5%	6.4%	5.7%	7.5%	5.7%	4.4%	14.0%	18.4%	15.6%	14.6%	17.1%	13.2%	11.8%

We present the performance comparison results of the baselines and our SAPM on the *Taobao* and *Amazon* datasets in Table 2. We obtain three main findings.

Our SPAM framework achieves the best performance. For instance, SAPM demonstrates improvements of 5.7% and 14.0% in MRR, 5.7% to 18.4% in NDCG, and 4.4% to 17.1% in HIT. This is because our LSI module has greater advantages in extracting long and short-term interests and stronger expression ability than traditional methods and we exploit feature-level information in FLC module to adaptively complement the interest modeling. Although SLi-Rec and CLSR can also extract long and short-term interests, SLi-Rec only uses one layer of GRU, so the effect is poor, while CLSR uses two layers of GRU and comparative learning, and achieves better results. However, they are still inferior to SAPM, because we use multiple GRU with small parameters to obtain stronger expression ability from different heads independently. Meanwhile, SURGE is close to CLSR in performance, but it has high secondary complexity and requires more resources.

The methods of considering evolution perform better than others. The graph network with quadratic structure performs better on both datasets. Among the original baseline methods, SURGE and CLSR demonstrate

Table 3. Ablation analysis on SAPM.

	Dataset	*Taobao*		*Amazon*	
Order	Method	MRR	*imp.*	MRR	*imp.*
(a)	SAPM	0.4558	-	0.3105	-
(b)	CLSR	0.4312	-5.40%	0.2483	-20.03%
(c)	*w/o* FLC & SAF	0.4423	-2.96%	0.3020	-2.74%
(d)	*w/o* LSI & SAF	0.3742	-17.90%	0.2773	-10.69%
(e)	*w* GRU	0.3913	-14.15%	0.3022	-2.67%
(f)	*w* SUM	0.4552	-0.13%	0.3032	-2.35%

relatively better performance on the *Taobao* dataset. SURGE extracts users' implicit interests through image modeling, while CLSR distinguishes long and short-term interests through self-supervised contrastive learning. Both of them model user representations from historical sequences. However, CLSR underperforms on the *Amazon* dataset, while SURGE remains relatively stable in terms of performance, indicating that SURGE's image modeling can better extract users' implicit interests in datasets with lower sequentiality.

Meanwhile, SLi-Rec, GRU4REC, and DIEN, which employ *RNNs* to evolve interaction sequences, and SUM, which models interests through multiple channels and empowers memory networks to capture the exact sequence of events, can capture users' interests as they change over time from history. These sequence modeling methods perform much better on *Taobao* than non-sequence modeling methods like NCF and DIN. We analyze the reason and find that users' behaviors are significantly influenced by previous interactions, and we cannot ignore their sequential nature.

The results are quite different on *Taobao* and *Amazon*, and all models perform better on *Taobao* than on the *Amazon*. Different from that on *Taobao*, SLi-Rec, and SURGE perform better than others, and DIN outperforms Caser, GRU4REC, DIEN, and even CLSR on *Amazon*. SURGE demonstrates excellent performance on both datasets, thanks to its quadratic graph structure. Moreover, SURGE performs much better than CLSR on *Amazon*. The reason for this may be that the *Amazon* dataset spans over ten years, and its sequentiality and stability are not as strong as that of *Taobao*. Moreover, Amazon users have shorter interaction sequences with weaker temporal patterns and less interaction information. Therefore, the overall predictive performance is better on *Taobao*.

4.3 Ablation Study

We conduct a series of ablation experiments in Table 3 and Figure 5(a), labeled as groups *(c)*-*(f)*, with groups *(a)* and *(b)* serving as the reference groups. In groups *(c)*-*(e)*, we eliminate specific structures to assess their impact on system performance. In group *(e)*, we replace the MHGRU structure with GRU, while in group *(f)*, we substitute FLC with SUM, further validating the effectiveness of our network architecture. We have the following four findings:

Our model performs better in extracting long and short-term interests. In group *(c)*, only the LSI module is active, while group *(b)* CLSR represents a similar long and short-term interest structure to LSI. By comparing groups *(c)* and *(b)*, our long and short-term interest modeling performs better on both datasets, demonstrating its ability to effectively extract user interests.

Although feature-level information cannot serve as the main factor for recommendations, it can mitigate the issue of interest drift. In group *(d)*, relying solely on feature-level information leads to a noticeable decrease in performance on both datasets, indicating that the sole application of feature-level information is insufficient. However, in group *(c)*, removing the FLC module leads to a performance drop of almost 3%. This suggests that feature-level information can effectively mitigate interest drift and serve as a valuable factor for interest recommendations.

Our model has a simple structure but stronger expression ability. By comparing group *(e)* with *(a)*, MHGRU exhibits greater expression ability compared to GRU, indicating the superiority of MHGRU which has a multi-head structure. In group *(f)*, we show that removing the Highway Channel and merging two operations into a single "update" operation is effective because our interest modeling already includes evolutionary information. In Figure 3, our LSI structure is simpler. We only perform one $RNNs$ (MHGRU) and one attention operation, while CLSR performs two $RNNs$ and three attention operations. However, comparing group *(b)* and *(c)*, it can be observed that our approach yields superior results in extracting long and short-term interests.

SAF is capable of effectively perceiving the stability of user interactions. In Figure 5(a), we vary the value of μ in Eq. 27 from 0.1 to 0.9 to replace the SAF structure, resulting in a significant decrease in performance on both datasets. This demonstrates that SAF effectively perceives the stability of sequences and balances the weights of LSI and FLC.

4.4 Parameter Sensitivity

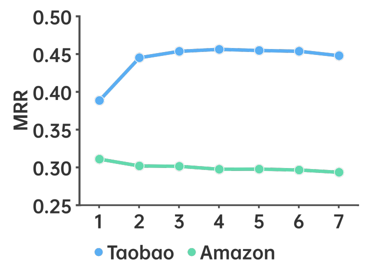
In this section, we study the impact of channel number N and hidden dimension d on the performance of the model, and we also discuss the role of MHGRU head numbers M and m. When we change one parameter, we set the other parameters to the best value to maintain consistency.

We search for the optimal value of N within the range of 1 to 7 in Figure 4(a). The results show that setting $N = 4$ achieves the best performance on the $Taobao$ dataset, while $N = 1$ is sufficient for the *Amazon* dataset. This is because the $Taobao$ dataset has longer sequence lengths and contains more interaction information, requiring a greater number of memory units to store. We then search for the optimal hidden layer dimension d within the range of 5 to 60 in Figure 4(b). According to the results, setting $d = 20$ achieves the best performance on the $Taobao$ dataset. When d is smaller than 20, the model lacks the expression ability to learn all the information, leading to underfitting. On the other hand, when d exceeds 20, the increased model complexity results in overfitting and decreases performance. For the Amazon dataset, which contains less information, the optimal value for d is 10.

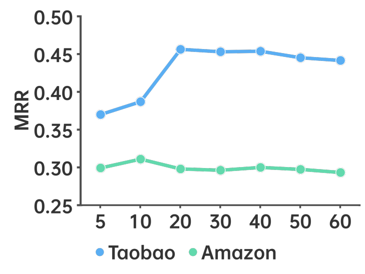
Figure 5(b) presents the results on the number of heads M in MHGRU and the number of heads m used for controlling attention. We conduct a two-dimensional search, where M is in the range of 2, 3, 4, 5, and m is in the range of 1 to M-1. Here, m represents the number of query heads, and M-m represents the number of key heads. We carry out experiments involving 10 groups of parameters. The experimental results show that the best performance is obtained when $M = 4$ and $m = 2$ in the $Taobao$ dataset. It should be noted that when $M = 2$ and $m = 1$, there is a 15% difference from the best performance, indicating that fewer parameters will result in underfitting. In the *Amazon* dataset, the best performance is achieved when $M = 5$ and $m = 1$. However, with fewer parameters, such as $M = 2$ and $m = 1$, users' interests can be fully learned, with only a 3% difference from the best performance.

4.5 Stability and SAPM

In this section, we first analyze the effectiveness of our SAPM framework in mitigating interest drift. Next, we delve into how our SAF module perceives sequence stability and adjusts the weights for interest recommendations. Additionally, we analyze the stability of behavioral relationships in the $Taobao$ dataset and *Amazon* dataset, providing new insights for sequence modeling and analysis. According to the partitioning method in Section 4.1, we compare all baselines on the stable and unstable datasets, and the results are shown in Tables 4 and 5. We further analyze the sequence stability in Figure 6. According to the experiment, we have the following four findings.

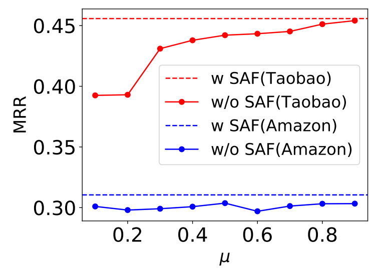


(a) Performance of different N

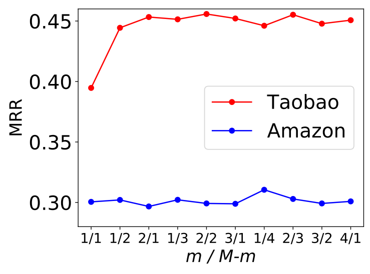


(b) Performance of different d

Fig. 4. The effects of channel number N and hidden dimension d.



(a) Performance of different μ



(b) Parameter Sensitivity on M and m

Fig. 5. The effects of interest weight μ and number of heads M and m.

Table 4. Comparison of the baselines in stable testing dataset. The best performance is indicated in bold, and the second-best performance is indicated in underlining. The *Imp*. means the percentage of improvement of the best compared with the second best.

Dataset	*Taobao*							*Amazon*						
Method/*Imp.*	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6
NCF	0.1635	0.0992	0.1375	0.1608	0.1185	0.2005	0.2632	0.1306	0.0743	0.1044	0.1238	0.0880	0.1524	0.2045
SASRec	0.3237	0.2544	0.3182	0.3487	0.2961	0.4321	0.5139	0.2450	0.1756	0.2266	0.2547	0.2054	0.3139	0.3894
DIN	0.3388	0.2639	0.3329	0.3665	0.3060	0.4528	0.5428	0.2803	0.2088	0.2668	0.2985	0.2437	0.3672	0.4522
SUM	0.4175	0.3572	0.4199	0.4481	0.4042	0.5375	0.6130	0.3183	0.2567	0.3086	0.3342	0.2924	0.4027	0.4712
Caser	0.3817	0.3189	0.3786	0.4081	0.3635	0.4903	0.5692	0.2532	0.1912	0.2375	0.2614	0.2210	0.3196	0.3837
GRU4REC	0.4319	0.3748	0.4332	0.4586	0.4201	0.5439	0.6118	0.2771	0.2107	0.2637	0.2894	0.2425	0.3556	0.4243
DIEN	0.4258	0.3649	0.4267	0.4543	0.4108	0.5416	0.6157	0.2102	0.1551	0.1918	0.2117	0.1783	0.2565	0.3099
SLi-Rec	0.3965	0.3334	0.3960	0.4251	0.3794	0.5124	0.5902	0.3104	0.2442	0.2998	0.3287	0.2802	0.3987	0.4763
SURGE	0.4385	0.3778	0.4392	0.4696	0.4238	0.5541	0.6353	0.3185	0.2499	0.3108	0.3413	0.2891	0.4185	0.5002
CLSR	0.4529	0.3949	0.4564	0.4849	0.4446	0.5748	0.6515	0.2722	0.2069	0.2599	0.2880	0.2410	0.3540	0.4291
SAPM	**0.4772**	**0.4211**	**0.4821**	**0.5111**	**0.4708**	**0.6002**	**0.6780**	**0.3323**	**0.2669**	**0.3221**	**0.3511**	**0.3041**	**0.4214**	**0.5008**
Imp.	5.4%	6.6%	5.6%	5.4%	6.3%	4.4%	4.1%	4.2%	4.0%	3.6%	2.9%	4.0%	0.7%	0.1%

All baselines perform poorly under unstable datasets. Comparing the performance of each baseline on the stable and unstable testing datasets, in Tables 4 and 5, we observe significant decrement in all metrics of MRR, NDCG, and HIT. Taking MRR for instance, on the *Taobao* dataset the baselines' performance drops by 8.6%-51% on unstable test sets compared to stable test sets. On the *Amazon* dataset, the drop is 16.9%-44.3%. This is due to the existence of interest drift, which increases the difficulty of model recommendations. Especially on unstable sequences, interest drift is more prominent, which also leads to poor test performance on unstable data.

Our SAPM effectively mitigates the impact of unstable datasets. As shown in Tables 4 and 5, we achieve the best results on both stable and unstable testing datasets. Our improvement percentage over the second baseline is greater on the unstable test set. On stable datasets, interest drift is not obvious, and other baselines can effectively

Table 5. Comparison of the baselines in unstable testing dataset. The best performance is indicated in bold, and the second-best performance is indicated in underlining. The *Imp*. means the percentage of improvement of the best compared with the second best.

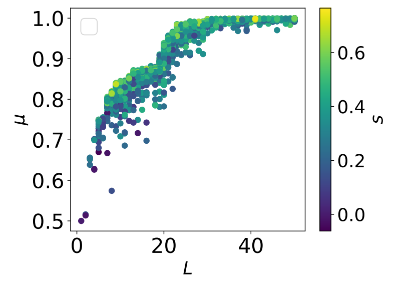
Dataset	*Taobao*							*Amazon*						
Method/*Imp.*	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6	MRR	NDCG@2	NDCG@4	NDCG@6	HIT@2	HIT@4	HIT@6
NCF	0.1079	0.0547	0.0789	0.0956	0.0654	0.1171	0.1622	0.1057	0.0528	0.0767	0.0932	0.0631	0.1142	0.1586
SASRec	0.2751	0.2095	0.2624	0.2897	0.2429	0.3556	0.4288	0.1794	0.114	0.1546	0.1791	0.1349	0.2217	0.2875
DIN	0.3022	0.2315	0.2909	0.3226	0.2682	0.3949	0.4801	0.2012	0.1338	0.1761	0.2029	0.1556	0.2456	0.3177
SUM	0.3711	0.3153	0.3645	0.3899	0.3554	0.4601	0.5279	0.2375	0.1709	0.2171	0.2432	0.1977	0.2961	0.3664
Caser	0.3306	0.2703	0.3187	0.3457	0.3047	0.4083	0.4805	0.1755	0.1107	0.1497	0.1733	0.1310	0.2144	0.2778
GRU4REC	0.3750	0.3191	0.3681	0.3918	0.3552	0.4593	0.5229	0.1967	0.1313	0.1729	0.1966	0.1544	0.2432	0.3066
DIEN	0.3809	0.3220	0.3758	0.4007	0.3609	0.4755	0.5423	0.1798	0.1214	0.1574	0.1781	0.1421	0.2190	0.2744
SLi-Rec	0.3415	0.2761	0.3343	0.3616	0.3135	0.4371	0.5104	0.2297	0.1599	0.2087	0.2369	0.1870	0.2912	0.3669
SURGE	0.4039	0.3460	0.4007	0.4248	0.3874	0.5036	0.5682	0.2262	0.1594	0.2049	0.2312	0.1852	0.2822	0.3528
CLSR	0.4018	0.3436	0.3975	0.4240	0.3836	0.4982	0.5690	0.2181	0.1498	0.1944	0.2228	0.1740	0.2695	0.3457
SAPM	**0.4267**	**0.3715**	**0.4256**	**0.4519**	**0.4166**	**0.5313**	**0.6020**	**0.2572**	**0.1892**	**0.2370**	**0.2655**	**0.2172**	**0.3191**	**0.3956**
Imp.	5.6%	7.3%	6.2%	6.4%	7.5%	5.5%	5.8%	8.3%	10.7%	9.2%	9.2%	9.9%	7.8%	8.0%

identify user interests, so every baseline performs well, and our baseline has a smaller percentage improvement of 0.1%-6.6%. However, on unstable datasets, other baselines fail to adequately address the issue of interest drift, while SAPM can alleviate this problem with the improvement of 5.5%-10.7%.

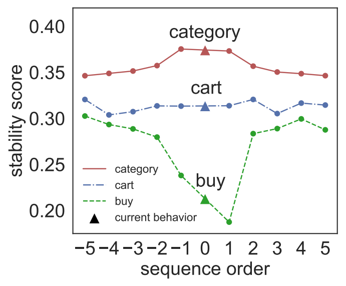
Shorter sequences are more prone to interest drift and SAPM effectively recognizes the stability of sequences and adjusts the weight μ of LSI to cope with interest drift. In Figure 6(a), we demonstrate the relationship between sequence length L, weight μ, and stability score s in *Taobao* dataset. When L is less than 20, there are more unstable sequences represented by the darker color. In this situation, the overall stability score μ is lower, indicating a decrease in the weight of interest recommendations because users may still be in the exploration phase. However, as the sequence length exceeds 20, stability increases, and the interaction history is more likely to be learned as user interest, resulting in higher μ values. Moreover, when the sequence length L is fixed, the darker points (representing lower stability) tend to be positioned lower, while the lighter points (representing higher stability) are positioned higher. This demonstrates that SAPM can identify stability differences among sequences of the same length and adjust μ accordingly.

Analyzing the stability of sequences helps in understanding user behavior and implementing different recommendation strategies. In Figure 6(b), we analyze the three behaviors and stability of "buy", "cart" and "category" in the *Taobao* dataset. In the figure, the sequence order value of 0 indicates the current behavior, -1 indicates the previous interactive product, and 1 indicates the next product. The triangle represents the corresponding interaction behavior of the current product. The ordinate value of three lines represents the stability score of adjacent items. The higher the stability score, the more similar the two adjacent items are. The "category" behavior indicates that the user continuously interacts with products of the same category. The stability before and after the "category" behavior is high, indicating that users interact with the same category continuously, are very interested in this kind of items, and can recommend similar items. Meanwhile, the "cart" behavior refers to the behavior of adding goods to the shopping cart. The stability score of interactive goods before and after the behavior changes little because the purchase link has not been completed. Finally, the stability before and after the "buy" behavior is low, which indicates that users are more inclined to explore new products after purchase.

We also conduct similar experiments on *Amazon* and obtain results similar to Figure 6(a). Due to the lack of behavior labels in *Amazon*, we do not conduct experiments in Figure 6(b).



(a)

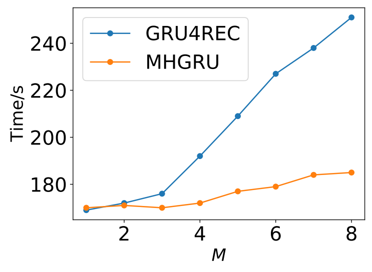


(b)

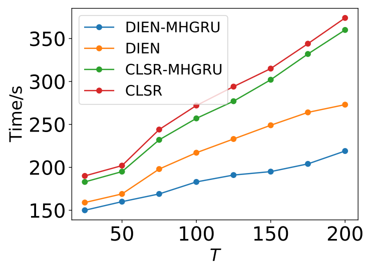
Fig. 6. (a): The relationship between L, s and μ. (b): The relationship between stability and user behaviors in $Taobao$.

4.6 Verification of MHGRU complexity

The inference time is a focal factor of industry attention. Delay in producing results would significantly impact user experience. If we can significantly reduce the inference time in the model phase, the industry can employ more complex models in the recall, coarse ranking, and re-ranking stages to enhance recommendation effectiveness. Therefore, we conduct two experiments to calculate the inference time required for the test dataset under different conditions, to verify the low complexity advantages of MHGRU.



(a) head number M



(b) sequence length T

Fig. 7. The impact of M and T on inference time.

The complexity of MHGRU is lower than that of a model that executes multiple general $RNNs$**.** While keeping other conditions unchanged, we vary the value of M to examine its correlation with runtime. In MHGRU, M represents the number of heads; while in GRU4REC, M represents the execution of M GRU operations, resulting in outputs of the same dimension. In Figure 7(a), the runtime of GRU4REC increases linearly, whereas MHGRU's runtime slightly increases. This demonstrates the improved parallelism of MHGRU. Ideally, considering only M, our complexity is $O(1)$, while GRU's is $O(M)$. However, in practice, influenced by GPU parallel acceleration, the complexity falls $O(1) < O_{MHGRU} << O(M)$.

MHGRU can effectively reduce inference time. CLSR and DIEN perform two $RNNs$ operations. We set MHGRU with 2 heads to replace the two $RNNs$ in these models and examine the impact on runtime for different sequence lengths T. As shown in Figure 7(b), replacing the two $RNNs$ structures with our MHGRU effectively reduces runtime, especially in the DIEN model. From the observations in the figure, when considering only T, the complexities of MHGRU and RNN are both $O(T)$. This validates the relationship $O(T) < O_{MHGRU} << O(M \times T)$ mentioned in Section 3.1.3.

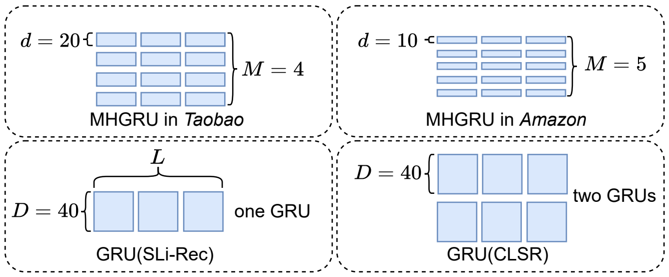


Fig. 8. The number of hidden layer parameters for MHGRU and GRU.

MHGRU does not significantly increase the dimensionality of hidden layers. Through experiments, it is found that the hidden layer dimension of our model is similar to the value of the baseline model. We randomly select two baselines to compare with this model in detail. For the values of the two parameters of dimension d and the number of heads M, we selected the optimal value in the parameter sensitivity analysis. For the Taobao dataset, we used $d = 20$, $M = 4$; and for Amazon, $d = 10$, $M = 5$. In addition, the optimal hidden layer dimension for the GRU is $D = 40$. We plot the hidden layers of the MHGRU model on the Taobao and Amazon datasets, as well as the hidden layer dimensions of the GRU in the SLi-Rec and CLSR models, as shown in Figure 8. The dimensions of the hidden layers of SLi-Rec and CLSR are the same on both datasets, and the L is the sequence length. It is found that SLi-Rec uses a single GRU with a hidden layer dimension of $40L$, while CLSR employs two GRUs with a resulting hidden layer dimension of $80L$. The dimension of the MHGRU on the Taobao dataset is $80L$, and on the Amazon dataset, the hidden layer dimension is $50L$. Therefore, our MHGRU model does not significantly increase the hidden layer dimension but performs exceptionally well in terms of performance.

Based on the above experimental results, our MHGRU demonstrates good parallelism and effectiveness. It is worth noting that this multi-head design is not limited to GRU but can also be applied to other $RNNs$ such as RNN and LSTM, as they share similar recurrent structures.

4.7 Case Study

We illustrate a case to explain the relationship between μ and s in Figure 9, where a higher μ value suggests recommending items with greater similarity to what the user has interacted with. A larger s value indicates a more uniform and stable nature of the user's historical sequence of interactions. Through analysis, it is found that User 1 and User 2 each interact with 5 items. The 5 items interacted with by User 1 are diverse and have lower similarity, while the items interacted with by User 2 are more uniform, all being tops. Hence, $s_1 < s_2$ and $\mu_1 < \mu_2$. This indicates that User 2's interaction sequence is more stable. The model will recommend items to User 2 that are more similar to his/her historical sequence based on the larger μ value. Likewise, User 1 will be recommended a more varied range of items because of their lower μ value, indicating a higher acceptance of diverse item types. Therefore, the model recommends shoes to User 1, a product type not seen in their historical sequence, and recommends tops, an item type that has appeared in User 2's historical sequence, to User 2.

5 RELATED WORK

This section briefly reviews the closely related works, including interest modeling, interest drift, and classical structures for the sequential recommendation.

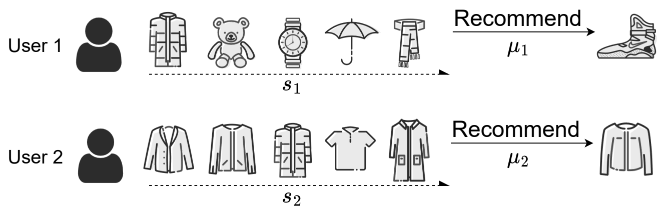


Fig. 9. Illustration of the interpretation of sequence stability.

5.1 Researches on interest modeling

This section introduces the main approaches in the field of interest modeling and classifies them into four categories: sequence recommendation, contrastive learning, multi-interest modeling, and diversified approaches for constructing interest representations using auxiliary information. Specifically, the sequence modeling technology can fully capture the diverse interests of users in the long and short term. Contrastive learning further optimizes the model performance based on interest modeling. Multi-interest modeling captures multiple interest points of the user to fully represent the complex interest structure of the user. By fusing other auxiliary information, the user's interest preferences can be studied and modeled in more detail.

Sequential recommendations [20, 28, 34, 35, 42] is a popular research direction aimed at using historical interaction sequences to predict the items. Interest modeling is a prevailing research area in the field of sequential recommendations. Several research [26, 54] use richer information to model interests. Targeting users' dynamic short-term interests and static long-term interests, a subset of studies [41, 43, 47, 55, 59] concurrently extracts long and short-term interests for a comprehensive user representation. Sun et al. [43] cleverly applies long short-term modeling to Point-of-Interest (POI) recommendation. Yu et al. [55] discuss the importance of user modeling in recommender systems and presents techniques to adapt to both long-and-short-term interests of users. [41, 59] enhance the accuracy of CTR prediction by disentangling the entanglement characteristics between long and short-term interests. Tran et al. [47] propose a novel approach to model user preferences in recommendation systems by utilizing quaternion space to encode both long-term and short-term interests, leading to improved performance over traditional Euclidean space methods.

Lots of research [27, 33, 37, 49] employ contrastive learning to optimize model performance based on interest modeling. For example, Lin et al. [27] adopt a dual-contrastive learning approach to handle data sparsity issues. Wang et al. [49] propose a framework of multilevel contrastive learning to achieve cross-view learning of user and item representations. Liu et al. [33] utilize contrastive learning to exploit item correlations and alleviate length skewness problems in sequential recommendation. Qin et al. [37] propose a novel anchor-guided contrastive learning process to perform basket denoising without requiring item-level relevance supervision.

Multi-interest modeling [12, 25, 29, 38] can capture the different interest points of users and model the rich interests of users more fully. For example, Lian et al. [25] introduce a memory matrix to store multiple interests of users and utilizes a high-speed channel to enhance the model. Qin et al. [38] explore multiple interests through interest evolution and then aggregates them to calculate user intent. Lin et al. [29] propose a dual-interest encoding layer to identify positive and negative feedback and untangles them. Huai et al. [12] introduce graph neural networks and meta-paths to learn and model multiple interests of users.

Regrettably, relying solely on interests is insufficient for a complete user representation, necessitating auxiliary modeling. For instance, [2, 5, 15, 18, 22, 40, 46] adopt intent to supplement user representation learning. Cai et al. [2]

use self-attention to identify transition patterns between categories and then infer user intent. Shen et al. [40] propose a model with three components for CTR to identify deeper user intent. Chen et al. [5] extract a user intent distribution function from unlabeled user behavior sequences and consider the learned intent. Tanjim et al. [46] learn item similarity from a user's interaction history using self-attention layers and obtain latent representations of user intent by using a time convolutional network layer on the user's behavior towards specific categories. Jin et al. [15] design a dual-intent network that learns user intent from attention mechanisms and historical data distributions to simulate the user's decision-making process and interaction with new items. Li et al. [18] combine multiple single-objective item lists with different user intents and learns personalized item-level weights in the model. Li et al. [22] combine the strength of hypergraph neural networks with disentangled representation learning to derive intent-aware representations of hyperedges to capture subtle differences in user purchase patterns; Wu et al. [51] unravel user characterization through social relationships. In addition, several research [21, 23, 24] employs novelty-based interest modeling to recommend innovative items. Some research [2, 39] explores item categories and attributes, utilizing their divergences to provide distinct perspectives for interest supplementation.

Although current interest modeling methods have achieved good results, existing methods have not fully considered the impact of user interest drift, and it is challenging to accurately model user interest with unstable interaction sequences.

5.2 Researches on interest Drift

This section explores research work related to interest drift. Interest drift is a common problem in e-commerce recommendations. The methods to deal with interest drift can be mainly divided into two categories. One is to treat interest drift as an uncertainty problem in the sequence. The other is to treat it as noise and deal with it accordingly.

Interest drift increases the difficulty of interest modeling. Some research treats interest drift as a sequence uncertainty problem[8, 9, 44, 48]. This approach describes uncertainty in items and sequences as Gaussian distributions.For instance,Fan et al. [8] utilize a Self-Attention module to capture item-item position-wise relationships in sequences, effectively incorporating uncertainty into model training. Sun et al. [44] employ uncertainty-aware graph convolutional networks to assist in uncovering and reducing uncertainty in unreliable scenarios. Fan et al. [9] use elliptical Gaussian distributions to describe uncertain items and sequences. Wang et al. [48] propose a two-branch VAE framework for sequential recommendation. It introduces model augmentation and variational augmentation to address the semantic inconsistency caused by traditional data augmentation. This type of method only coarsely handles interest drift, and fine-grained interest fluctuations remain unaddressed.

Most recently, researchers deal with interest drift between the training data and the testing data to enhance model robustness. Representative researches include [50, 53], which utilize and optimize the objective function of DRO (Distributionally Robust Optimization), to address the shortcomings of ERM (Empirical Risk Minimization), a widely used learning framework in recommendation models. Streaming-DRO [50] focuses on addressing the problem of group fairness by maximizing the worst-case performance, which reduces the variance in loss estimations due to data sparsity in recommendation systems. Distributionally Robust Optimization mechanism for SeqRec (DROS) [53] incorporates DRO into sequential recommendation to deal with the discrete data. It has a carefully-designed distribution adaption paradigm to achieve better generalization ability, handling the dynamics of data distribution and exploring possible distribution shifts between training and testing. These two methods belong to model-agnostic learning framework. Our work is different from them in twofold: (1) Our research primarily focuses on dealing with interest drift in the interaction sequence, while the two works propose the DRO methods that mainly study the robustness of the models to deal with the distribution shift between the training set and the testing set. (2) Both of them belong to model-agnostic frameworks, focusing on the design of loss functions or optimizer. While our study emphasizes the design of model structures.

Other methods consider interest drift as a noise issue[2, 30, 36, 52]. This method accounts for item transitions and similarity within interaction sequences, identifying noise in unstable sequences. Lin et al. [30] introduce a global representation learning module and a gating module. The former is used to improve the modeling of the user's global preference, while the latter balances local and global representations by considering candidate item information. Xie et al. [52] incorporate contrastive learning into a VAE model to learn distinctive features of different users using contrastive loss. It further optimizes the contrastive loss to ensure personalized and salient features across different users. Ma et al. [36] propose a hierarchical attention mechanism. Building upon personalized attention attribute perception, it enhances item representation learning by activating interactions caused by specific attributes. Soft denoising methods employ attention mechanisms to reduce focus on noisy items, while hard denoising [57] strictly filters and removes strong noise items to enhance the sequence. However, denoising methods cannot entirely resolve the issue of interest drift, and deletion operations may result in a loss of semantic information. In addition, they do not consider feature-level interest drift.

In our work, we propose a novel method to quantify interest drift with stability. At the same time, we can visually see user interest drift with stability indicators.

5.3 Research on classical structure for sequential recommendation

In this section, we analyze the two mainstream model architectures in the field of sequence recommendation: $RNNs$ and Transformer architectures, and summarize their respective advantages and limitations. Different methods show their advantages and challenges when utilizing these model architectures. By analyzing how these architectures are used, we can more easily explore where optimization approaches can start.

The $RNNs$ structure has been widely used in sequential recommender systems due to its inherent sequence processing ability. However, the $RNNs$ structure is unable to implement parallel computation, which limits their computational efficiency. The Transformer architecture is also widely used for sequence recommendation because it does not have this limitation, but it will encounter quadratic time complexity when dealing with large-scale data. To solve these challenges, researchers have proposed different improvement methods. For example, Li et al. [19] proposed the AutoMLP method to further improve the recommendation performance, but it still cannot make full use of the inherent advantages of $RNNs$ in capturing temporal information. DIEN[60]and CLSR[59] adopted two RNN structures to extract time interval features for optimization, but this method is less efficient. To address the above challenges, we need to optimize the parallelized RNN architecture to improve computational efficiency while retaining the advantages of $RNNs$ for processing sequential data.

Our differences from other works are threefold, corresponding to three key challenges in existing works: (1) For the first challenge that interest drift has not been fully studied, we transform interest drift into a sequence stability problem to better study drift learning. (2) For the second challenge that the implicit feature level modeling hasn't been well explored, we consider both item-level and feature-level stability to better explore fine-grained information and design a framework that incorporates stability-aware preference to mitigate interest drift. (3) For the third challenge that existing methods either lack expression ability or are quite complex, we design an efficient parallel $RNNs$ structure with Multi-head GRU to address the shortcomings of the $RNNs$ model architecture.

6 CONCLUSION

In this paper, we identify three challenges of interest drift (i.e., unstable interaction sequences), fine-grained implicit features, and the expression ability of the modeling technique in the sequential recommendation. To address those challenges, we propose the stability-aware preference modeling framework SAPM. It can capture both long and short-term interests with an efficient multi-head GRU (MHGRU) structure, and obtain multi-dimensional feature-level information with an improved memory network; Then it measures interest drift through stability awareness and adaptively fuses both item-level and feature-level information into user interest. We conduct extensive experiments in two real-world datasets and the results demonstrate the effectiveness of our work as well as the importance of each component. Furthermore, we analyze the relationship between stability and interaction behaviors and obtain several useful insights. In future work, we will further explore the associations between stability, categories, behaviors, and time, as well as serendipity, to enhance user interest modeling and personalized recommendations.

ACKNOWLEDGEMENT

This research was supported by the National Natural Science Foundation of China (Grant No. 62172149).

REFERENCES

[1] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural News Recommendation with Long- and Short-term User Representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 336–345. https://doi.org/10.18653/v1/P19-1033

[2] Renqin Cai, Jibang Wu, Aidan San, Chong Wang, and Hongning Wang. 2021. Category-Aware Collaborative Sequential Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Virtual Event Canada, 388–397. https://doi.org/10.1145/3404835.3462832

[3] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential Recommendation with Graph Neural Networks. arXiv:arXiv:2106.14226

[4] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, Marina Del Rey CA USA, 108–116. https://doi.org/10.1145/3159652.3159668

[5] Yongjun Chen, Zhiwei Liu, Jia Li, Julian McAuley, and Caiming Xiong. 2022. Intent Contrastive Learning for Sequential Recommendation. In *Proceedings of the ACM Web Conference 2022*. 2172–2182. https://doi.org/10.1145/3485447.3512090 arXiv:2202.02519 [cs]

[6] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based Neural Recommender. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, Torino Italy, 147–156. https://doi.org/10.1145/3269206.3271810

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv:1412.3555 [cs]

[8] Ziwei Fan, Zhiwei Liu, Alice Wang, Zahra Nazari, Lei Zheng, Hao Peng, and Philip S. Yu. 2022. Sequential Recommendation via Stochastic Self-Attention. arXiv:arXiv:2201.06035

[9] Ziwei Fan, Zhiwei Liu, Lei Zheng, Shen Wang, and Philip S. Yu. 2021. Modeling Sequences as Distributions with Uncertainty for Sequential Recommendation. arXiv:arXiv:2106.06165

[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, Perth Australia, 173–182. https://doi.org/10.1145/3038912.3052569

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-Based Recommendations with Recurrent Neural Networks. arXiv:arXiv:1511.06939

[12] Zepeng Huai, Yuji Yang, Mengdi Zhang, Zhongyi Zhang, Yichun Li, and Wei Wu. 2023. M2GNN: Metapath and Multi-interest Aggregated Graph Neural Network for Tag-based Cross-domain Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1468–1477. https://doi.org/10.1145/3539618.3591720 arXiv:2304.07911 [cs]

[13] Dongmin Hyun, Junsu Cho, Chanyoung Park, and Hwanjo Yu. 2020. Interest Sustainability-Aware Recommender System. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, Sorrento, Italy, 192–201. https://doi.org/10.1109/ICDM50108.2020.00028

[14] Dongmin Hyun, Chanyoung Park, Junsu Cho, and Hwanjo Yu. 2022. Beyond Learning from Next Item: Sequential Recommendation via Personalized Interest Sustainability. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 812–821. https://doi.org/10.1145/3511808.3557415

[15] Di Jin, Luzhi Wang, Yizhen Zheng, Guojie Song, Fei Jiang, Xiang Li, Wei Lin, and Shirui Pan. 2023. Dual Intent Enhanced Graph Neural Network for Session-based New Item Recommendation. arXiv:2305.05848 [cs]

[16] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. arXiv:1808.09781 [cs]

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263

[18] Jiayu Li, Peijie Sun, Zhefan Wang, Weizhi Ma, Yangkun Li, Min Zhang, Zhoutian Feng, and Daiyue Xue. 2023. Intent-Aware Ranking Ensemble for Personalized Recommendation. arXiv:2304.07450 [cs]

[19] Muyang Li, Zijian Zhang, Xiangyu Zhao, Wanyu Wang, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2023. AutoMLP: Automated MLP for Sequential Recommendations. In *Proceedings of the ACM Web Conference 2023*. ACM, Austin TX USA, 1190–1198. https://doi.org/10.1145/3543507.3583440

[20] Nian Li, Xin Ban, Cheng Ling, Chen Gao, Lantao Hu, Peng Jiang, Kun Gai, Yong Li, and Qingmin Liao. 2024. Modeling User Fatigue for Sequential Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Washington DC USA, 996–1005. https://doi.org/10.1145/3626772.3657802

[21] Pan Li, Maofei Que, Zhichao Jiang, Yao Hu, and Alexander Tuzhilin. 2020. PURS: Personalized Unexpected Recommender System for Improving User Satisfaction. In *Fourteenth ACM Conference on Recommender Systems*. ACM, Virtual Event Brazil, 279–288. https://doi.org/10.1145/3383313.3412238

[22] Ran Li, Liang Zhang, Guannan Liu, and Junjie Wu. 2023. Next Basket Recommendation with Intent-aware Hypergraph Adversarial Network. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Taipei Taiwan, 1303–1312. https://doi.org/10.1145/3539618.3591742

[23] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, and Guojun Wang. 2019. HAES: A New Hybrid Approach for Movie Recommendation with Elastic Serendipity. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, Beijing China, 1503–1512. https://doi.org/10.1145/3357384.3357868

[24] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, Guojun Wang, and Kenli Li. 2020. Directional and Explainable Serendipity Recommendation. In *Proceedings of The Web Conference 2020*. ACM, Taipei Taiwan, 122–132. https://doi.org/10.1145/3366423.3380100

[25] Jianxun Lian, Iyad Batal, Zheng Liu, Akshay Soni, Eun Yong Kang, Yajun Wang, and Xing Xie. 2021. Multi-Interest-Aware User Modeling for Large-Scale Sequential Recommendations. arXiv:arXiv:2102.09211

[26] Eduardo Lima, Weishi Shi, Xumin Liu, and Qi Yu. 2019. Integrating Multi-level Tag Recommendation with External Knowledge Bases for Automatic Question Answering. *ACM Transactions on Internet Technology* 19, 3 (2019), 1–22. https://doi.org/10.1145/3319528

[27] Guanyu Lin, Chen Gao, Yinfeng Li, Yu Zheng, Zhiheng Li, Depeng Jin, and Yong Li. 2022. Dual Contrastive Network for Sequential Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Madrid Spain, 2686–2691. https://doi.org/10.1145/3477495.3531918

[28] Guanyu Lin, Chen Gao, Yu Zheng, Jianxin Chang, Yanan Niu, Yang Song, Kun Gai, Zhiheng Li, Depeng Jin, Yong Li, and Meng Wang. 2024. Mixed Attention Network for Cross-domain Sequential Recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. ACM, Merida Mexico, 405–413. https://doi.org/10.1145/3616855.3635801

[29] Guanyu Lin, Chen Gao, Yu Zheng, Jianxin Chang, Yanan Niu, Yang Song, Zhiheng Li, Depeng Jin, and Yong Li. 2023. Dual-Interest Factorization-heads Attention for Sequential Recommendation. In *Proceedings of the ACM Web Conference 2023*. ACM, Austin TX USA, 917–927. https://doi.org/10.1145/3543507.3583278

[30] Jing Lin, Weike Pan, and Zhong Ming. 2020. FISSA: Fusing Item Similarity Models with Self-Attention Networks for Sequential Recommendation. In *Fourteenth ACM Conference on Recommender Systems*. ACM, Virtual Event Brazil, 130–139. https://doi.org/10.1145/3383313.3412247

[31] Kun Lin, Zhenlei Wang, Shiqi Shen, Zhipeng Wang, Bo Chen, and Xu Chen. 2022. Sequential Recommendation with Decomposed Item Feature Routing. In *Proceedings of the ACM Web Conference 2022*. ACM, Virtual Event, Lyon France, 2288–2297. https://doi.org/10.1145/3485447.3512101

[32] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, London United Kingdom, 1831–1839. https://doi.org/10.1145/3219819.3219950

[33] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S. Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive Self-supervised Sequential Recommendation with Robust Augmentation. arXiv:arXiv:2108.06479

[34] Tianze Luo, Yong Liu, and Sinno Jialin Pan. 2024. Collaborative Sequential Recommendations via Multi-view GNN-transformers. *ACM Transactions on Information Systems* 42, 6 (2024), 1–27. https://doi.org/10.1145/3649436

[35] Zheqi Lv, Wenqiao Zhang, Zhengyu Chen, Shengyu Zhang, and Kun Kuang. 2024. Intelligent Model Update Strategy for Sequential Recommendation. In *Proceedings of the ACM on Web Conference 2024*. ACM, Singapore Singapore, 3117–3128. https://doi.org/10.1145/3589334.3645316

[36] Rui Ma, Ning Liu, Jingsong Yuan, Huafeng Yang, and Jiandong Zhang. 2022. CAEN: A Hierarchically Attentive Evolution Network for Item-Attribute-Change-Aware Recommendation in the Growing E-commerce Environment. In *Sixteenth ACM Conference on Recommender Systems*. 278–287. https://doi.org/10.1145/3523227.3546773 arXiv:2208.13480 [cs]

[37] Yuqi Qin, Pengfei Wang, and Chenliang Li. 2021. The World Is Binary: Contrastive Learning for Denoising Next Basket Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Virtual Event Canada, 859–868. https://doi.org/10.1145/3404835.3462836

[38] Yuqi Qin, Pengfei Wang, Biyu Ma, and Zhe Zhang. 2022. A Multi-Interest Evolution Story: Applying Psychology in Query-based Recommendation for Inferring Customer Intention. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 1655–1665. https://doi.org/10.1145/3511808.3557221

[39] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, and Kun Gai. 2019. Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Paris France, 565–574. https://doi.org/10.1145/3331184.3331230

[40] Qijie Shen, Hong Wen, Wanjie Tao, Jing Zhang, Fuyu Lv, Zulong Chen, and Zhao Li. 2022. Deep Interest Highlight Network for Click-Through Rate Prediction in Trigger-Induced Recommendation. In *Proceedings of the ACM Web Conference 2022*. 422–430. https://doi.org/10.1145/3485447.3511970 arXiv:2202.08959 [cs]

[41] Qijie Shen, Hong Wen, Jing Zhang, and Qi Rao. 2022. Hierarchically Fusing Long and Short-Term User Interests for Click-Through Rate Prediction in Product Search. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 1767–1776. https://doi.org/10.1145/3511808.3557351

[42] Ke Sun, Chenliang Li, and Tieyun Qian. 2024. City Matters! A Dual-Target Cross-City Sequential POI Recommendation Model. *ACM Transactions on Information Systems* 42, 6 (2024), 1–27. https://doi.org/10.1145/3664284

[43] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to Go Next: Modeling Long- and Short-Term User Preferences for Point-of-Interest Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (2020), 214–221. https://doi.org/10.1609/aaai.v34i01.5353

[44] Yatong Sun, Bin Wang, Zhu Sun, and Xiaochun Yang. 2021. Does Every Data Instance Matter? Enhancing Sequential Recommendation by Eliminating Unreliable Data. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Montreal, Canada, 1579–1585. https://doi.org/10.24963/ijcai.2021/218

[45] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, Marina Del Rey CA USA, 565–573. https://doi.org/10.1145/3159652.3159656

[46] Md Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian McAuley. 2020. Attentive Sequential Models of Latent Intent for Next Item Recommendation. In *Proceedings of The Web Conference 2020*. ACM, Taipei Taiwan, 2528–2534. https://doi.org/10.1145/3366423.3380002

[47] Thanh Tran, Di You, and Kyumin Lee. 2020. Quaternion-Based Self-Attentive Long Short-term User Preference Encoding for Recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. ACM, Virtual Event Ireland, 1455–1464. https://doi.org/10.1145/3340531.3411926

[48] Yu Wang, Hengrui Zhang, Zhiwei Liu, Liangwei Yang, and Philip S. Yu. 2022. ContrastVAE: Contrastive Variational AutoEncoder for Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 2056–2066. https://doi.org/10.1145/3511808.3557268

[49] Ziyang Wang, Huoyu Liu, Wei Wei, Yue Hu, Xian-Ling Mao, Shaojian He, Rui Fang, and Dangyang Chen. 2022. Multi-Level Contrastive Learning Framework for Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 2098–2107. https://doi.org/10.1145/3511808.3557404

[50] Hongyi Wen, Xinyang Yi, Tiansheng Yao, Jiaxi Tang, Lichan Hong, and Ed H. Chi. 2022. Distributionally-Robust Recommendations for Improving Worst-case User Experience. In *Proceedings of the ACM Web Conference 2022*. ACM, Virtual Event, Lyon France, 3606–3610. https://doi.org/10.1145/3485447.3512255

[51] Jiahao Wu, Wenqi Fan, Jingfan Chen, Shengcai Liu, Qing Li, and Ke Tang. 2022. Disentangled Contrastive Learning for Social Recommendation. arXiv:arXiv:2208.08723

[52] Zhe Xie, Chengxuan Liu, Yichi Zhang, Hongtao Lu, Dong Wang, and Yue Ding. 2021. Adversarial and Contrastive Variational Autoencoder for Sequential Recommendation. In *Proceedings of the Web Conference 2021*. ACM, Ljubljana Slovenia, 449–459. https://doi.org/10.1145/3442381.3449873

[53] Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A Generic Learning Framework for Sequential Recommendation with Distribution Shifts. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Taipei Taiwan, 331–340. https://doi.org/10.1145/3539618.3591624

[54] Lina Yao, Quan Z. Sheng, Xianzhi Wang, Wei Emma Zhang, and Yongrui Qin. 2018. Collaborative Location Recommendation by Integrating Multi-dimensional Contextual Information. *ACM Transactions on Internet Technology* 18, 3 (2018), 1–24. https://doi.org/10.1145/3134438

[55] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Macao, China, 4213–4219. https://doi.org/10.24963/ijcai.2019/585

[56] Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual Sparse Attention Network For Session-based Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4635–4643. https://doi.org/10.1609/aaai.v35i5.16593

[57] Chi Zhang, Yantong Du, Xiangyu Zhao, Qilong Han, Rui Chen, and Li Li. 2022. Hierarchical Item Inconsistency Signal Learning for Sequence Denoising in Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, Atlanta GA USA, 2508–2518. https://doi.org/10.1145/3511808.3557348

[58] Wei Zhao, Benyou Wang, Jianbo Ye, Yongqiang Gao, Min Yang, and Xiaojun Chen. 2018. PLASTIC: Prioritize Long and Short-term Information in Top-n Recommendation Using Adversarial Training. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Stockholm, Sweden, 3676–3682. https://doi.org/10.24963/ijcai.2018/511

[59] Yu Zheng, Chen Gao, Jianxin Chang, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2022. Disentangling Long and Short-Term Interests for Recommendation. In *Proceedings of the ACM Web Conference 2022*. 2256–2267. https://doi.org/10.1145/3485447.3512098 arXiv:2202.13090 [cs]

[60] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep Interest Evolution Network for Click-Through Rate Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (2019), 5941–5948. https://doi.org/10.1609/aaai.v33i01.33015941

[61] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, London United Kingdom, 1059–1068. https://doi.org/10.1145/3219819.3219823