Knowledge and Information Systems (2024) 66:3231–3259
https://doi.org/10.1007/s10115-023-02057-4

REGULAR PAPER

Position-category-aware attention network for next-item recommendation

Liqing Qiu[1] · Mingjian Dou[1] · Caixia Jing[1] · Yuying Liu[1]

Received: 25 February 2023 / Revised: 23 August 2023 / Accepted: 26 December 2023 /
Published online: 31 January 2024

Abstract

The next-item recommendation can extract critical information from the historical sequence and predict the next actions of users. To better extract users' interests, some sequential recommender methods propose position-aware attention networks to obtain users' general intentions. Nonetheless, although these methods have achieved superior performances, they cannot effectively extract core information from historical behavior sequences such as position weights, the dynamic categories of users, and the dynamic preferences of users. The position information in the historical sequence can assist in the modeling of user interest, and the dynamic category of users can help us ensure the major intention of users. Moreover, capturing the dynamic preference of users can help the model learn the evolution tendency of user interest and make better recommendations. Therefore, this paper proposes a Position-category-aware Attention Network (PCAN) to consider the above three factors. First, this model obtains the dynamic category of the user in the data preprocessing stage. Then, a long-term attention module is constructed to get the interaction between users and items in the long-term sequential behavior, to better capture the users' long-term preference representation. Meanwhile, the model utilizes the self-attention method to extract users' short-term interest features. Finally, two kinds of preference representation are adaptively fused through an attention-based method. On five kinds of Amazon public datasets, the experimental results indicate that our proposed model PCAN achieves better performances on AUC, $Precision$, and $Recall$, which demonstrates the superior performance of the method.

Keywords Next-item recommendation · Self-attention method · Position-category-aware attention network · Dynamic category · Preference representation

✉ Liqing Qiu
qiuliqing2019@163.com

[1] Shandong Province Key Laboratory of Wisdom Mine Information Technology, College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

1 Introduction

With the rapid development of the Internet, recommendation systems have become an effective way to help users find available information [1]. As an important branch of recommendation systems, sequential recommendation has recently gained popularity. It learns the evolution of user interests by modeling the historical sequence of users to predict the user's next behavior [2]. In the early stage of sequential recommendation, Markov chain and matrix factorization methods have been widely applied. For instance, Rendle et al. [3] proposed factorizing personalized Markov chains model FPMC that introduces a personalized transfer matrix based on the Markov chain to capture both time information and users' general preferences. In addition, to solve the sparse transfer matrix problem, it uses the matrix factorization method, which reduces the number of parameters while improving model performance. The study of [4] proposed a Fossil method, which uses matrix factorization to model long-term preferences and utilizes the Markov chain to capture short-term sequential dynamics. To sum up, although these models have made good performances, the Markov chain recommender models can only capture short-term dependencies. When the historical sequence is long, the method cannot effectively capture user interest. Later, some scholars proposed applying deep learning techniques to the sequential recommendation [5]. For instance, Hidasi et al. [6] proposed GRU4REC, they utilize multi-layer RNN to extract sequential information from historical behaviors and train the model through a ranking loss function. However, this method only captures the short-term preference and ignores the context information of users and items.

In the last few years, an increasing number of sequential recommenders have attempted to utilize attention-based methods, including the location-aware context attention network (PACA) [7]. However, these models cannot take full use of the category information and position information hidden in the historical sequence. To be specific, first, most attention-based models regard item ID embedding as the historical session representation and neglect to consider the category information. Second, capturing the dynamic category of users can help models understand user intention and make recommendations to them. Third, one item has different levels of importance when it appears in different positions of historical sequence, and those vital items in the sequence should be assigned higher weight when predicting the next item. Finally, integrating the position and category information effectively is very important to improve the recommendation performance.

To solve the above problems, this paper proposes a novel recommendation method named position-category-aware attention network (PCAN). First, The PCAN method extracts the user dynamic category in the data preprocessing stage. Second, considering the context information of items and users, the long-term attention module is designed to obtain users' general preferences. Third, the short-term self-attention network is constructed to generate users' current interests. Finally, this paper utilizes an attention-based method to adjust the weight of these two user interests under specific circumstances.

A summary of the contributions is given below:

1. A new position-category-aware attention network called PCAN is proposed. This method uses the user dynamic category and attention-based network to capture the users' interest representation from the historical behavior.
2. A long-term attention module is proposed to integrate the position and context features of items and users, which builds the deep interaction between items and users. Moreover, to generate the deep preference representation of short-term historical behavior, a self-attention module is proposed.

3. The experiment results on five kinds of Amazon review datasets demonstrate that our proposed algorithm outperforms all competitors under the metric of Recall, Precision, and Area Under Curve.

The remainder of this article is organized as follows. Section 2 makes a brief review of the sequential recommendation, attention mechanism, as well as positional embedding. Section 3 first introduces some symbols utilized in the paper, and then introduces our proposed PCAN model. The experimental result and the performance of all models are provided in Sect. 4. Section 5 concludes the research and future directions.

2 Related work

The paper will review three related tasks in this section, which include sequential recommendation, attention mechanism, as well as positional embedding.

2.1 Sequential recommendation

Different from traditional recommendation systems, sequential recommender tries to model the evolution of user interests. It learns the changes in users' interests and predicts the next item based on user historical records. Rendle et al. [3] proposed to fuse the Markov chain and matrix factorization for the sparse sequential recommendation. He et al. [4] proposed to use the k-order interaction to predict the next item. Hidasi et al. [6] used GRU to model the dependencies between sequential behaviors, which first introduced recurrent neural networks (RNN) to the sequential recommendation.

Moreover, considering the dynamics of user intent, Jannach et al. [8] proposed to use LSTM to build a self-regression model RRN, which adaptively learns the dynamic embedding of items and users. The study of [9] proposed to employ hierarchical RNN to make a sequential recommendation, which characterizes the evolution of users' interests in a session sequence and makes personalizing session-based recommendations. Ludewig et al. [10] proposed to combine RNN with the K-Nearest Neighbor algorithm to improve the recommendation performance. What's more, some works utilized convolution neural networks to model users' interests in historical behavior sequences. For example, the study of [11] proposed a CNN-based model Caser. This model uses the convolutional neural network and the Latent Factor Method to capture the sequential features and user features.

2.2 Attention mechanism

In the last few years, a lot of attention-based sequential recommenders have been proposed. For instance, the study of [12] utilized the self-attention method to capture users' preferences from historical behaviors. At every time step, this method seeks those actions related to the target item to predict the next item. AFM [13] introduced the attention mechanism to traditional factorization machines, to learn the importance and extent of the interaction feature (learn a weight for each interaction item). Moreover, DMAN [14] employed the attention mechanism and dynamic memory network to generate user preference. Cen et al. [15] proposed a multi-interest sequential recommender (ComiRec), which utilizes a multi-head self-attention method and capsule network to capture multiple interests from users' historical behaviors, but it fails to obtain the temporal interaction of items. The study of [16] proposed the SASRec model, which employs the self-attention method to extract core information from user sequential behaviors, it considers items' relative position and shows superior performance. Nonetheless, this method ignores the short-term interest of users. Zhang et al. [17] proposed an AttRec model, which models the short-term user preference through the self-attention method and uses the metric learning method to model the long-term interaction of items and users. The study of [18] proposed a deep attention network model (DIN) and applied it to advertising recommendations, which uses a local activation unit to capture user interest from historical behaviors.

Generally, users' interests follow a hierarchical pattern, and users often show their interests in different granularities. Considering different granularities of users' interests, the study of [19] proposed to use a hierarchical network to obtain interests of different levels. Zhuang et al. [20] proposed a SHAN model, which utilizes two layers attention network to model sequential behaviors. Specifically, this model uses the first layer to generate users' general preferences, and it utilizes the second layer to couple the general preference and user sequential representation. However, this method does not strengthen short-term interests and ignores the category information of items. Liu et al. [21] proposed a STAMP model, which utilizes a novel neural attention network to capture users' long-term interests and short-term preferences in the historical sequence. Nonetheless, this model only treats the last term in the historical behavior as the short-term sequence, which affects the modeling of user preferences. The study of [22] used BERT to model users' preferences from historical behaviors. SSE-PT [23] applied a personalized transformer to the sequential recommendation, which utilizes new regularization technology of randomly shared embedding to regularize the embedding layer. Compared with SASRec [16], SSE-PT tends to recommend items closer to the target item. Zhang et al. [24] proposed a TLSAN model, which utilizes the attention module and personalized position embedding to generate user interest, but it cannot effectively obtain users' dynamic interest from long sequential behaviors. Niu et al. [25] proposed a CLSR method, they employ the BIGRU and self-attention mechanism to generate the general preference and current interest, but they limited the length of the recently interacted sequence to obtain short-term preference representation. Therefore, the captured user preference representation cannot adapt to all kinds of users, especially some users whose historical behaviors are sparse.

2.3 Positional embedding (PE)

When capturing the sequential information from user historical behaviors, the position information of each item in a session can assist in the modeling of dependencies between sequential behaviors.

The transformer [26] first introduced the concept of position embedding into the deep learning field, which uses sine and cosine functions to generate the position feature representation. The study of [16] added a novel position embedding that can be learned to the input embedding and showed superior performance. Atrank [27] extracted temporal information from the source dataset and concatenated the embedding of time with input embedding to obtain the final item representation. Huang et al. [28] proposed a CSAN model, which generates the position matrix for users' behavior sequences and adds the positional weights to the input embedding directly. Nonetheless, the position matrix cannot be trained and fails to capture the personalized preferences of users. The PACA proposed by Cao et al. [7] utilized positional vectors to learn the weight of each item in a session and multiplied item embedding by position embedding to generate the position feature representation of each session. However, this method considers all features of users' preferences the same and cannot capture their personalized interests.

To sum up, the above-mentioned position-aware attention methods cannot effectively extract core information from historical behavior sequences such as position weights, the user dynamic category, and the dynamic preferences of users. In contrast, the PCAN model improves the recommendation performance by capturing the deep interaction of user and item context information, and it adaptively fuses the captured general preference and current interest.

3 Position-category-aware attention network

This paper will define the problem of the next-item recommendation, describe the details of the PCAN method, and then introduce the model training and loss function in this section.

3.1 Problem formulation

Before introducing the details of the PCAN method, this paper first defines some basic symbols and the problem. Let $U = \{u_1, u_2, \ldots, u_m\}$ denotes the set of users and $I = \{i_1, i_2, \ldots, i_n\}$ denotes the set of items. $C = \{c_1, c_2, \ldots, c_c\}$ represents the items' category set. For each user $u \in U$, her/his sequential sessions are denoted as $L_t^u = \{S_1^u, S_2^u, \ldots, S_t^u\}$, where t represents the current time step, S_i^u represents the user's historical behaviors on the $i - th$ day. Specifically, this paper divides the historical behaviors into sessions according to their occurring time, each session represents a series of actions within a day. On the one hand, at time step t, the item set $S_t^u \in R^S$ contains the user's recently interacted items that can reflect their short-term preferences, S is the length of the short-term sequence. On the other hand, the historical item set before time step t, denoted by $L_{t-1}^u = \{S_1^u, S_2^u, \ldots, S_{t-1}^u\}$, can reflect the general interests of users, where $L_{t-1}^u \in R^L$, L represents the length of the long-term sequence. Moreover, this paper utilizes the user historical session to get sequential category information from the category set C. For example, user Zhangsan bought a series of products in the dataset, and we represent his historical sequence as {banana, iphone7, earphone, computer, adidas}. Then, this paper searches the corresponding category information of each item in the historical sequence from category set C and generates the sequential category representation {fruit, electronics, electronics, electronics, clothes}. Specifically, for each historical sequence of user u, this paper uses $C_L^u = \{c_1^u, c_2^u, \ldots, c_{t-1}^u\}$ to denote the categorical sequence corresponding to the long-term actions, uses $C_S^u = \{c_t^u\}$ to represent the short-term item categorical sequence, where $C_L^u \in R^L$ and $C_S^u \in R^S$.

3.2 The network architecture

This paper proposes a new method base on the attention mechanism and shows it in Fig. 1. To effectively obtain the long- and short-term interest from user behavioral sequence, this paper first divides the whole user historical behaviors into the long- and short-term sequence. The behaviors that happened on the latest day are regarded as short-term items and other behaviors as long-term items. To utilize the user context information more comprehensively, this paper extracts the user dynamic category in the data preprocessing stage.

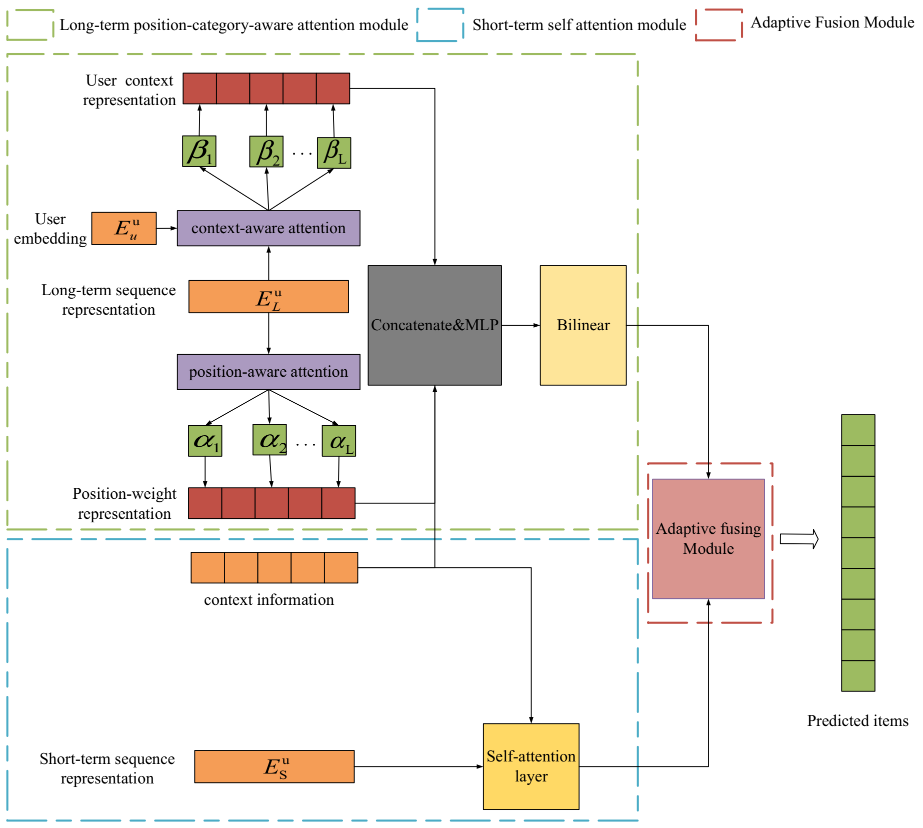
Long-term position-category-aware attention module
Short-term self attention module
Adaptive Fusion Module
User context representation
β_1 β_2 ... β_L
User embedding
E_u^u
context-aware attention
Long-term sequence representation
E_L^u
position-aware attention
α_1 α_2 ... α_L
Position-weight representation
Concatenate&MLP
Bilinear
Adaptive fusing Module
Predicted items
context information
Short-term sequence representation
E_S^u
Self-attention layer

Fig. 1 Architecture of PCAN. PCAN mainly consists of three modules: (1) the long-term position-category-aware attention module incorporates category and position information to generate user preference in the long-term historical sequence. The input of this module is the user embedding $E_u^u \in R^{2D}$, long-term sequence representation $E_L^u \in R^{L\times 2D}$ and context information. $\alpha_i \in R^1$ and $\beta_i \in R^1$ represent the weight representation generated by the corresponding methods; (2) the short-term self-attention module explicitly filters out the noisy items and learns an accurate preference representation for the short-term session. The input of this module is a short-term sequence representation $E_S^u \in R^{S\times 2D}$ and context information; (3) Adaptive fusion module dynamically adjusts long-term and short-term interest weights in specific circumstances and generates the final user preference representation

Considering that the self-attention mechanism is not sensitive to the position information, this paper utilizes this method to capture the important short-term items from the relatively short sequence. At the same time, the position-category-aware module is sensitive to position information and context information, which helps extract the vital items from the long sequence. Therefore, we use this method to deal with long-term historical behavior. To fuse these two modules effectively, this paper uses the attention-based method to adjust the weight of long- and short-term preferences dynamically. However, this design way can increase the number of parameters and training time. We plan to propose a better way to deal with the long- and short-term interest and leave it for future work.

The core idea of our method is to generate the user interest representation through adaptively fusing long- and short-term interests. First, the method embeds the item, user, and context information to get their dense embedding representation. After that, the model inputs the embedding representation of all features into the attention network to capture the general preference and current interest. Specifically, to get the long-term interests before the time step t, the model uses a long-term attention network to generate the position weight representation and user context representation. Then, MLP (multi-layer perception) is used to learn the nonlinear interaction between them and context information. The core idea of long-term position-category-aware attention module came from [7, 19, 20, 24], which employs the attention mechanism to utilize the position information and context information. Furthermore, inspired by [27], which utilized the self-attention method to project heterogeneous user behaviors. The model uses the self-attention method to obtain the deep interaction of short-term item sets and context information, generating users' current intentions. Finally, an attention-based method adaptively fuses the captured interest and generates the final user preference representation. This idea came from [29], which utilizes the attention-based method to fuse general interest and short-term preference under specific circumstances. Next, we will describe each module of the PCAN model.

3.2.1 Embedding module

The embedding module aims to convert the high-dimensional sparse vector into a low-dimensional dense vector representation. The embedding structure diagram of this module is shown in Fig. 2. Specifically, for each historical sequence of user u, uses ct_u^t to represent the dynamic user category at time step t, and uses tg_t^u to denote the target item at time step t.

When processing long-term historical behaviors, this paper limits the length of the historical sequence to r. If the length is less than r, the right side of the historical sequence will be added a padding item 0. Else if the length is greater than r, the latest r items will be selected as our input. The embedding module of our model is based on the specific sequence of a given user, which is defined as follows:

$$E_c^u = CL(ct_u^t)$$
$$E_L^c = CL(C_L^u)$$

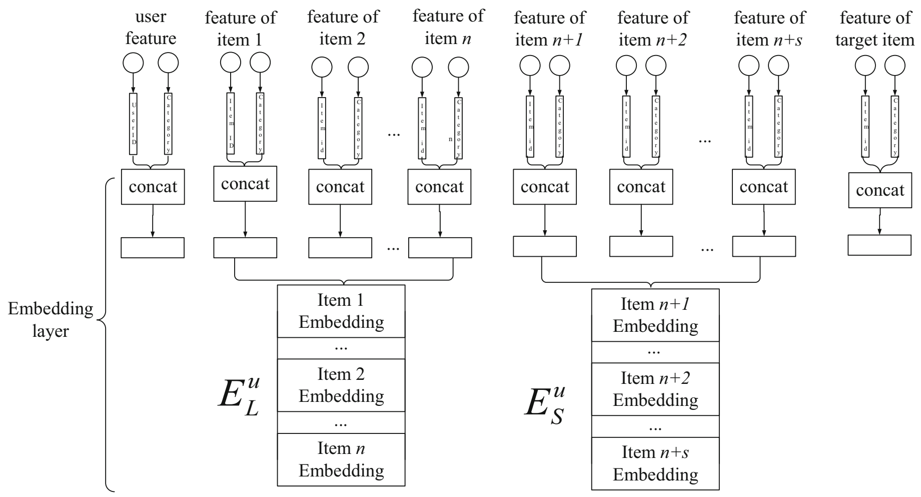

Fig. 2 Structure diagram of the Embedding layer

Table 1 Notation

Notation	Description
I, U, C	Item set, user set, and category set
$\mathbf{I}, \mathbf{U}, \mathbf{C}$	Item embedding set, user embedding set, and category embedding set
L_{t-1}^u, S_t^u	Long-term and short-term item sequence at time step t
ct_u^t	Dynamic user category ID
C_L^u, C_S^u	User u's long-term and short-term item category sequence
D	Embedding dimension
n, m, c	Number of items, users, and categories
L, S	Length of long-term and short-term sequence

$$\begin{aligned}
E_S^c &= CL(C_S^u) \\
E_T^u &= IL\left(tg_t^u\right) \\
E_u^u &= Conc(E_c^u,\ UL(u)) \\
E_L^u &= Conc(E_L^c,\ IL(L_{t-1}^u)) \\
E_S^u &= Conc(E_S^c,\ IL(S_t^u))
\end{aligned} \tag{1}$$

where $Conc(.)$ is the concatenate function, $IL(.)$ represents the embedding lookups of item ID, $CL(.)$ is the embedding lookups of category, $UL(.)$ represents the user ID's embedding lookups. u represents the user ID, $E_c^u \in R^D$ denotes the embedding of the dynamic user category at time step t, $E_L^c \in R^{L\times D}$ denotes the embedding of the long-term item category, L is the length of the long-term sequence, $E_u^u \in R^{2D}$ denotes the user embedding, $E_L^u \in R^{L\times 2D}$ denotes the embedding of the long-term items, $E_S^u \in R^{S\times 2D}$ denotes the short-term item embedding, S is the length of the short-term sequence, and $E_T^u \in R^{2D}$ represents the target item embedding. Table 1 shows the notations used and their descriptions.

In summary, given users and their historical behavior sequence HS, the model aims to predict the next items and recommend them to users.

3.2.2 Dynamic user category extraction module

The user category can be described as he/she is interested in some categories of items at a solid time. To get the user dynamic category at each time step t, this paper regards the most frequent item category in the historical sequence as the user category. For example, Tom bought a bag, a phone, and a headset three days ago, we represent the historical sequence as {bag, phone, headset}. Phone and headset belong to electronics, bag belongs to grocery, and the user category would be considered to be electronics. Moreover, Tom bought three bags yesterday, and the historical sequence of him is {bag, phone, headset, bag, bag, bag}. In this sequence, the numbers of grocery and electronics are 4 and 2, so this method regards grocery as the user category. The user category can change dynamically over time, and capturing the user dynamic category is an important direction to improve recommendation performance. Some researchers have tried to extract the user category in the training stage, but the time consumption is relatively high [30, 31]. After that, this paper uses the captured user category to generate their embedding representation and concatenates it to the user ID embedding.

3.2.3 Long-term position-category-aware attention module

The method proposed by [7] discards the personalized interest of users and context information, which assumes the position weight in each historical sequence is the same for all users. Moreover, it only considers the user's general preferences. Ying et al. [20] proposed to utilize a hierarchical network to generate long-term interest and short-term preference, which regards user embedding as context information and utilizes the original attention method to model the user interest representation. However, it does not strengthen the short-term interests and discards the category information. This paper builds the long-term attention network through the modeling of the position and category of items and users. First, this module extracts the weights of each position p_t in the historical sequence. And then the context-aware attention method is used to obtain user context representation u_t. At last, this module utilizes MLP to fuse $p_{t.}$, u_t and context information.

To utilize the context information of each item in the historical sequence, this module utilizes the mean pooling method to get each item's mean pooling representation and denotes the pooling sequence as $mp = \{mp_1, mp_2, ..., mp_L\}$. The mean pooling method is defined as Eq. (2), where v_i represents the i-th item embedding, the index L represents the length of the long-term sequence, $mp_i \in R^{2D}$ represents the mean value of all items in the historical sequence, $mp \in R^{L\times 2D}$. Furthermore, to utilize the category information of each item, this paper concatenates the item ID embedding and category embedding for each item, so the dimension of item embedding is $2D$, that is, $v_i \in R^{2D}$.

And then MLP is used to get the session-specific feature tmp_emb_i. The computing process is defined as Eq. (3), where the symbolic $\varnothing(.)$ represents the sigmoid activation function, which can capture the nonlinear interaction between v_i and mp_i. $W_1 \in R^{2D\times 2D}$ and $W_2 \in R^{2D\times 2D}$ are transformation matrices of long-term item sets and their mean pooling representation, $tmp\_emb_i \in R^{2D}$ is the session-specific feature.

$$mp_i = \frac{v_1 + v_2 + \cdots + v_L}{L} \tag{2}$$

$$tmp\_emb_i = \varnothing(W_1 v_i + W_2 mp_i) \tag{3}$$

$$\alpha_i = \frac{\exp(pos_i^T tmp\_emb_i)}{\sum_{k=1}^{L}\exp(pos_k^T tmp\_emb_k)} \tag{4}$$

$$h = \sum_{i=1}^{L} \alpha_i v_i \tag{5}$$

After that, a series of position vectors are used to capture the position weight information, which is denoted as $pos = \{pos_1, pos_2, ..., pos_L\}$. Then, the module uses the $softmax$ function to measure the attention weight α_i of each position in the long-term item list. In this way, the important item in a session will be assigned a higher attention score. The formula is defined as Eq. (4), where $pos \in R^{L\times 2D}$, $pos_i^T \in R^{2D}$, $\alpha_i \in R^1$.

After capturing the position weight α_i, this paper multiplies it by the item embedding to get the item feature representation. And then adds them together to get the output state h. The output state represents the positional weight representation of items in the historical sequence. The formula is defined as Eq. (5), where $h \in R^{2D}$.

On the other hand, a context-aware attention method is utilized to generate the user context representation. First, this method calculates each item's weight in the long-term sequential behaviors of a given user. Second, it aggregates the item embedding and the generated weights to get the user context representation. Specifically, the weight computing process is defined as Eq. (6), where $W_3 \in R^{2D\times 2D}$ and $b_1 \in R^{2D\times 1}$ are model parameters, v_i represents the i-th item embedding vector in the historical sequence, the network first gets the hidden representation $E_i \in R^{2D}$ by feeding the embedding of each item $i \in L^u_{t-1}$ into the MLP. Function $\varnothing(.)$ is the sigmoid activation function which enhances the nonlinear capability of this network.

$$E_i = \varnothing(W_3 v_i + b_1) \tag{6}$$

$$\beta_i = \frac{\exp({E^u_u}^T E_i)}{\sum_{p\epsilon L^u_{t-1}} \exp({E^u_u}^T E_p)} \tag{7}$$

$$u^{long}_{t-1} = \sum_{i\varepsilon L^u_{t-1}} \beta_i v_i \tag{8}$$

To utilize the dynamic user category information, this paper concatenates the embedding of the user ID and the corresponding category embedding to get the hybrid embedding representation E^u_u. At the same time, this paper regards E^u_u as the context embedding matrix and computes the normalized similarity between E^u_u and E_i. And then $softmax$ function is used to get the attention score $\beta_i \in R^1$. The computing process is defined as Eq. (7), where $E^u_u \in R^{2D}$ represents the user embedding.

Then, this paper computes the user context representation u^{long}_{t-1} as a sum of the item embedding representation weighted by the attention scores. The formula is defined as Eq. (8), where $u^{long}_{t-1} \in R^{2D}$.

At last, the module uses MLP to obtain the nonlinear interaction of user context representation, position weight representation, and context information. And then this module utilizes a bilinear operation to generate long-term user interest representation. The component is defined as follows:

$$L^{final}_{t-1} = MLP(Conc(u^{long}_{t-1},\ h,\ E^u_T)) \tag{9}$$

$$u^{final}_{t-1} = W_b L^{final}_{t-1} \tag{10}$$

where $Conc(u^{long}_{t-1}, h, E^u_T)$ represents the concatenation of user context representation, position weight representation, and target item embedding, MLP is the multi-layer perception that captures their nonlinear interaction, $L^{final}_{t-1} \in R^{2D}$ represents the long-term user hybrid representation, $W_b \in R^{2D\times 2D}$ is the bilinear weight matrix, $E^u_T \in R^{2D}$ represents target item embedding, u^{final}_{t-1} denotes the generated long-term preference representation.

3.2.4 Short-term self-attention module

Users' short-term preferences can reveal their behavioral tendencies and is important to predict the next items. The model utilizes the multi-head self-attention method and multi-head vanilla attention to obtain short-term user interest, the number of heads will be discussed in the experiment part. And the structure of the short-term self-attention module is shown in Fig. 3.

First, the users' short-term item set is obtained from the items which they have interacted with on the latest day. Furthermore, this paper uses the scaled dot-product attention [26] to

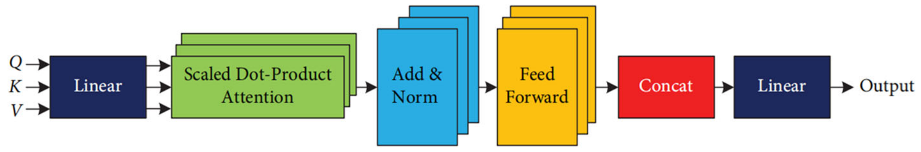

Fig. 3 Structure of the short-term self-attention module

compute the relationship of different features and defines it as follows:

$$Attention = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{11}$$

where Q, K, and V denote query, keyword, and value, $\sqrt{d}$ represents the scaling factor. And in our case, Q, K, and V are the same objects, the self-attention operation regards the short-term item embedding E_S^u as input. This paper utilizes linear operations to convert it into three matrices, and then inputs them into an attention layer. And the multi-head mechanism is utilized in the attention layer to capture the deep interaction.

$$SA = Attention(E_S^u W^Q, E_S^u W^K, E_S^u W^V) \tag{12}$$

where $W^Q \in R^{2D\times 2D}$, $W^K \in R^{2D\times 2D}$ and $W^V \in R^{2D\times 2D}$ represent the projection matrices, $E_S^u \in R^{S\times 2D}$ denotes the short-term item embedding, $SA \in R^{S\times 2D}$ represents the similarity of the S short-term items.

Second, considering that the self-attention mechanism can only capture the linear interaction of different features, this paper applies a point-wise feed-forward network to endow it with nonlinear capability. To solve the model's overfitting problem, this paper uses the dropout technique [32].

$$F = FFN(SA) = Dropout(ReLU\left(SAW^{(1)} + b^{(1)}\right)W^{(2)} + b^{(2)}) \tag{13}$$

where $W^{(1)}$ and $W^{(2)}$ represent $2D * 2D$ dimension matrix, $b^{(1)}$ and $b^{(2)}$ represent $2D$-dimensional bias vectors, $F \in R^{S\times 2D}$ is the output of the network. And then this paper successively applies layer normalization and residual connection [33] to the output to get the self-attention representation S_u.

$$S_u = LayerNorm(SA + F) \tag{14}$$

$$LayerNorm(x) = \alpha \odot \frac{x - \mu}{\sqrt{\delta^2 + \varepsilon}} + \beta \tag{15}$$

where $LayerNorm(x)$ denotes the layer normalization function used in our paper, $\odot$ represents the element-wise function, δ and μ denote the variance and mean of x, α represents the scale factors, β is the bias terms.

After capturing the self-attention representation S_u, this paper uses vanilla attention to generate the current preference representation u_t^{final}.

The procedure of multi-head vanilla attention is similar to the multi-head self-attention method. The differences between them are that vanilla attention considers the context information (target item embedding) as the query matrix of attention, and S_u is regarded as the key and value matrix. This method is defined as follows:

$$SA' = Attention(E_T^u W^{Q'}, S_u W^{K'}, S_u W^{V'}) \tag{16}$$

$$F' = FFN(SA') = Dropout(ReLU\left(SA\prime W^{(3)} + b^{(3)}\right)W^{(4)} + b^{(4)}) \tag{17}$$

$$u_t^{final} = LayerNorm\ \left(SA' + F'\right) \tag{18}$$

where $E_T^u \in R^{2D}$ denotes the target item embedding, $W^{Q'} \in R^{2D\times 2D}$, $W^{K'} \in R^{2D\times 2D}$ and $W^{V'} \in R^{2D\times 2D}$ are the projection matrices, $W^{(3)}$ and $W^{(4)}$ represent $2D * 2D$ dimension matrices, $b^{(3)}$ and $b^{(4)}$ represent $2D$-dimensional bias vectors, u_t^{final} denotes the short-term preferences generated by this module.

3.2.5 Adaptive fusion module

Both the long-term attention network and short-term self-attention module have strengths and weaknesses. Therefore, accommodating these two modules is very necessary. Instead of simply using add function to combine them, e.g., $u^{final} = u_{t-1}^{final} + u_t^{final}$, this paper designs an adaptive method to fuse them. To be specific, this paper uses an attention-based method to adjust long-term and short-term interest weights dynamically. By this means, the model can find the optimal weights between two kinds of preference representation under specific circumstances. And the proposed attention-based adaptive fusion method is defined as follows:

$$\gamma = \delta(W_m Conc(u_{t-1}^{final}, u_t^{final}, E_T^u) + b_m) \tag{19}$$

$$u^{final} = \gamma * u_{t-1}^{final} + (1 - \gamma) * u_t^{final} \tag{20}$$

where $W_m \in R^{2D\times 2D}$ and $b_m \in R^{2D\times 1}$ are model parameters, $\delta(.)$ represents the sigmoid activation function, $\gamma \in R^1$ is the weight of long-term preference, and $(1-\gamma) \in R^1$ represents the weight of recent intent. This paper uses $Conc(u_{t-1}^{final}, u_t^{final}, x_{context})$ to represent the concatenation of short-term preferences, long-term preferences, and contextual information. In fact, for contextual information, the model can include all kinds of important features, such as time stamps, target item categories, rating information, and location information. In this paper, this model only uses the target item embedding E_T^u as the contextual feature.

3.3 Model training and loss function

This paper trains PCAN with the whole historical behaviors in the train set and then predicts the items' labels in the test set. To be specific, the model will be more effective when the predicted label is closer to the truth. Moreover, this paper uses the unified sigmoid cross entropy loss to optimize the method and to seek the optimal model parameters. And this paper defines the loss function as Eq. (21):

$$Loss = -\sum_{j,u} y_j \log\left(\delta\left(f\left(u_t^{final}, M_j^I\right)\right)\right) + (1 - y_j)\log\left(1 - \delta\left(f\left(u_t^{final}, M_j^I\right)\right)\right) + \lambda(||\theta||) \tag{21}$$

where $f(.)$ represents a dot-product function, M_j^I is the $j - th$ item embedding, $\theta = \{\mathbf{U}, \mathbf{I}, W_*, b_*\}$ represents the regularized parameters in the training process, and λ represents the l2 regulation rate. This paper uses $y \in \{0, 1\}$ to represent the labels, and utilizes $\delta(.)$ to represent the sigmoid activation function.

Table 2 Five Amazon dataset statistics (After preprocessing)

Dataset	Users	Items	Categories	Instances	Train sessions	Test sessions	Avg Length
Electronics	39,991	22,048	673	561,100	568,506	39,991	10.21
Video	5436	4295	58	83,748	95,668	5436	12.25
Office	1720	901	170	29,387	40,828	1720	10.28
Home	11,567	7722	683	143,088	152,310	11,567	9.48
Phones	2097	1553	82	21,659	22,876	2097	8.83

4 Experiments

4.1 Experiment setting

4.1.1 Datasets

This paper evaluates our proposed model and the baselines over the popular Amazon review dataset [12].

Amazon Dataset The dataset includes product data and metadata from Amazon. This paper chooses five categories of Amazon Datasets including Electronics, Video, Office, Home, and Phones to conduct the experiments. In the following experiment, the paper only utilizes users, items, categories, and timestamp information to make a recommendation. Then the paper preprocesses the original dataset according to the following subsection and shows the statistics in Table 2.

4.1.2 Dataset preprocessing

Firstly, this paper selects those items with interactions of no less than 8 and those users whose interactions are not less than 10 from each dataset to make sure that each item and user in the dataset is effective for our experiment, and those infrequent users and items are removed by us.

Secondly, this paper regards the interactions within a day as a session and divides all users' historical behaviors into ordered sessions. To ensure the existence of both short-term and long-term historical sequences, this paper chooses those users who own the number of sessions between 4 and 90, and then utilize their historical actions to generate the corresponding historical sequences.

Finally, this paper lets $L_t^u = \{S_1^u, S_2^u, \ldots, S_t^u\}$ represent the sequential behaviors for user u at each time step t, and this paper chooses the $1 \sim (t-1)$-th sessions as the historical sequences. For the methods that take the user's long- and short-term preference (TLSAN, SHAN, PCAN) into consideration, this paper regards the newest session without the target items as the short-term item set, and the $1 \sim (t-2)$-th sessions as the long-term item set to obtain the training set. For other methods, this paper considers all sessions before time t as the historical sequence to generate the training set. Moreover, this paper uses the latest session to generate the test set. Specifically, if the number of items in the latest session is 1, this paper will regard it as the target item. And the paper chooses the $1 \sim (t-2)$-th and $(t-1)$-th sessions as the long- and short-term historical sequence. Else this paper randomly selects one item in the latest session as the target item. The other items in the session are considered as the short-term historical behaviors, and the $1 \sim (t-1)$-th sessions are regarded as the long-term sequence.

4.1.3 Evaluation standard

This paper uses three frequently used metrics, the area under the curve (AUC), $Precision$, and $Recall$ to evaluate the efficiency of our proposed method PCAN.

Generally speaking, AUC is equal to the probability that the predictive score of the model for a positive sample is greater than that for a negative sample when a positive sample and a negative sample are randomly selected. Therefore, it can reflect the sorting ability of the classifier for the samples. However, it cannot reflect the accuracy of prediction, so this paper introduces recall and precision to evaluate the experimental result more comprehensively. The following formula shows the computing process of AUC.

$$AUC = \frac{1}{|U^{Test}|} \sum_{u \in U^{Test}} \frac{1}{|I_u^+||I_u^-|} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \sigma(p_{u,i} > p_{u,j}) \tag{22}$$

where $|U^{Test}|$ denotes the size of the test set, U^{Test} denotes the test set, I_u^+ and I_u^- represent the positive and negative sample sets respectively, $\sigma(.)$ represents the indicator function, $p_{u,j}$ is the predicted score of user u may choose negative sample j, and $p_{u,i}$ represents the predicted score of user u may choose positive sample i.

$Precision@K$ and $Recall@K$: Precision rate and recall rate are two measures which are often used to evaluate the quality of results. The precision rate is specific to the predicted results, it represents the ratio of the correctly predicted positive samples to all predicted positive samples. The recall rate is specific to the original samples, it represents the ratio of correctly predicted positive samples to all original samples. K represents only the top-K items that are considered. And these two evaluation metrics are defined as follows:

$$Precision@K = \frac{1}{|U^{Test}|} \sum_{u \in U^{Test}} \frac{\sum_{s=1}^{K} fp(s, pos(u))}{K} \tag{23}$$

$$Recall@K = \frac{1}{|U^{Test}|} \sum_{u \in U^{Test}} \frac{\sum_{s=1}^{K} fp(s, pos(u))}{NK(u)} \tag{24}$$

where $|U^{Test}|$ represents the size of the test set, U^{Test} denotes the test set, $pos(u)$ represents the ground-truth item set related to user u, $NK(u)$ denotes the number of positive items in the top-K predicted items of user u, $pos(u)$ denotes the set of items which has interacted with user u, $fp(s, pos(u))$ represents an indicator function, it returns 1 when item s is in $pos(u)$ and returns 0 if item s is not in $pos(u)$.

In our experiment, this paper utilizes AUC to evaluate the classification ability and uses $Precision@K$ and $Recall@K$ to test the prediction accuracy of the proposed method.

4.1.4 Hyperparameters

The PCAN model and all baselines use different hyperparameters, the adjusting of embedding size ES, learning rate λ, decay rate φ, L2 regulation rate ω, the number of heads h in short-term self-attention module and the long-term sequence length L_S will be discussed in Sect. 4.3. To ensure the fairness and comparability of the experiments, this paper sets the batch sizes to 32.

As for BPR-MF, SHAN, PACA, and TLSAN, this paper sets the parameter values to be the same as [24]. The parameters for other baselines, such as learning rate, decay rate, and L2 regulation rate are tuned according to the corresponding papers [6, 15, 18]. And the lengths of recent sessions are set to 90 for them, which keeps the same as PCAN. Due to space limitation, this paper does not show the details.

Moreover, for our proposed model and all baselines, the paper considers the dot-production function as the ranking function, and the Stochastic Gradient Descent method is utilized as the optimizer for all baselines and PCAN.

4.1.5 Baselines

This paper compares PCAN with the baseline methods as follows:

BPR-MF [34]: A nonsequential recommender that uses the Bayesian personalized ranking to optimize the order between items, which can effectively rank the items that interacted with users according to their preference. This method introduces BRP into matrix decomposition, it learns the model parameters by maximizing a posteriori probability and aims to make visited items better than unvisited items.

GRU4REC [6]: A classical sequential recommender that applies RNN to the sequential recommendation area and regards the historical behaviors as hidden states to calculate the dependencies of adjacent behaviors. GRU4REC then regards the final hidden states of GRU to generate user preference representation.

DIN [18]: It introduces a local activation mechanism to utilize different user behavior features. At the same time, DIN proposes mini-batch aware regularization and adaptive activation functions to assist model training. This model considers the historical behavior sequentially and captures user interest effectively.

SHAN [19]: User general taste and recent demand can make different contributions to the prediction of the next item, so SHAN proposes the hierarchical attention network to learn the dynamic long-term preference and short-term interest. And then it combines these two features to generate the comprehensive user representation.

PACA [7]: PACA users a trainable position vector to represent the positional weight of each item in the historical sequence. And then it utilizes an attention-based method to integrate the context information and position information in the historical sequence, to generate the user preference representation.

ComiRec-SA [15]: It was proposed to capture multiple interests of users from click sequence, and two methods were utilized by this algorithm. One method applied to it is the capsule network, and the other method is the self-attentive method. The self-attention method utilizes the self-attention mechanism to compute user interest capsule, and this paper utilizes this method as the competitor.

TLSAN [24]: A state-of-the-art method that was proposed recently, it mainly utilizes personalized time aggregation and attention methods to model user preference representation. Long-term preference and recent demand are considered by this model, it proposes two kinds of feature-wise attention layers to capture these two preferences effectively. And the final user preference representation is generated by the second feature-wise attention layer.

4.2 Results analysis

Table 3 shows the AUC results of all methods on the five datasets, Tables 4, 5, 6 and 7 show the $Precision$ and $Recall$ performance of all methods on the Electronics and Homes_kitchen

Table 3 Results (AUC) on all datasets

Model	Electronics	Video	Office	Home	Phones
BPR-MF	0.7457	0.6609	0.5576	0.6352	0.6084
GRU4REC	0.7916	0.7549	0.8110	0.6970	0.7348
DIN	0.7491	0.7658	0.7256	0.6602	0.7000
SHAN	0.7234	0.7930	0.7645	0.6573	0.6581
PACA	0.7573	0.7927	0.7907	0.6902	0.6981
ComiRec-SA	0.7279	0.7666	0.6994	0.6622	0.7020
TLSAN	0.7966	0.7908	0.8319	0.6985	0.6762
PCAN	**0.8164**	**0.8267**	**0.8430**	**0.7163**	**0.7663**

Bold indicates the optimal experimental results

Table 4 Result ($Recall$) on electronics dataset

Model	Recall@1	Recall@10	Recall@20	Recall@30	Recall@40	Recall@50
BPR-MF	0.0052	0.0290	0.0431	0.0508	0.0594	0.0670
GRU4REC	0.0035	0.0236	0.0379	0.0488	0.0579	0.0662
DIN	0.0025	0.0133	0.0186	0.0225	0.0256	0.0283
SHAN	0.0026	0.0219	0.0344	0.0455	0.0544	0.0624
PACA	0.0024	0.0155	0.0261	0.0348	0.0429	0.0506
ComiRec-SA	0.0029	0.0164	0.0276	0.0385	0.0485	0.0569
TLSAN	0.0039	0.0239	0.0380	0.0493	0.0589	0.0674
PCAN	**0.0052**	**0.0300**	**0.0465**	**0.0595**	**0.0705**	**0.0803**

Bold indicates the optimal experimental results

Table 5 Result ($Precision$) on electronics dataset

Model	Prec@1	Prec@10	Prec@20	Prec@30	Prec@40	Prec@50
BPR-MF	0.0052	0.0029	0.0022	0.0017	0.0015	0.0013
GRU4REC	0.0035	0.0024	0.0019	0.0016	0.0014	0.0013
DIN	0.0025	0.0013	0.0009	0.0007	0.0006	0.0006
SHAN	0.0026	0.0022	0.0017	0.0015	0.0014	0.0012
PACA	0.0024	0.0016	0.0013	0.0012	0.0011	0.0010
ComiRec-SA	0.0029	0.0016	0.0014	0.0013	0.0012	0.0011
TLSAN	0.0039	0.0024	0.0019	0.0016	0.0015	0.0013
PCAN	**0.0052**	**0.0030**	**0.0023**	**0.0020**	**0.0018**	**0.0016**

Bold indicates the optimal experimental results

Table 6 Result (Recall) on Home_Kitchen dataset

Model	Recall@1	Recall@10	Recall@20	Recall@30	Recall@40	Recall@50
BPR-MF	0.0009	0.0085	0.0165	0.0230	0.0292	0.0343
GRU4REC	0.0012	0.0111	0.0200	0.0280	0.0351	0.0419
DIN	0.0006	0.0046	0.0087	0.0128	0.0165	0.0199
SHAN	0.0006	0.0078	0.0146	0.0208	0.0266	0.0318
PACA	0.0016	0.0091	0.0161	0.0221	0.0277	0.0328
ComiRec-SA	0.0010	0.0063	0.0120	0.0174	0.0231	0.0281
TLSAN	0.0015	0.0126	0.0224	0.0311	0.0391	0.0464
PCAN	**0.0025**	**0.0171**	**0.0287**	**0.0386**	**0.0475**	**0.0556**

Bold indicates the optimal experimental results

Table 7 Result (Precision) on Home_Kitchen dataset

Model	Prec@1	Prec@10	Prec@20	Prec@30	Prec@40	Prec@50
BPR-MF	0.0009	0.0008	0.0008	0.0008	0.0007	0.0007
GRU4REC	0.0012	0.0011	0.0010	0.0009	0.0009	0.0008
DIN	0.0006	0.0005	0.0004	0.0004	0.0004	0.0004
SHAN	0.0006	0.0008	0.0007	0.0007	0.0007	0.0006
PACA	0.0016	0.0009	0.0008	0.0007	0.0007	0.0007
ComiRec-SA	0.0010	0.0006	0.0006	0.0006	0.0006	0.0006
TLSAN	0.0015	0.0013	0.0011	0.0010	0.0010	0.0009
PCAN	**0.0025**	**0.0017**	**0.0014**	**0.0013**	**0.0012**	**0.0011**

Bold indicates the optimal experimental results

dataset. It can be found that our proposed method PCAN outperforms other models on five datasets.

PCAN outperforms TLSAN, it is because that TLSAN cannot effectively integrate the current interest and general preferences of users, and it does not take full use of the context information of items and users when generating the corresponding preference representation.

PCAN achieves better performances than DIN and ComiRec-SA, it may be because that DIN utilizes the attention mechanism to learn the relationship between historical behavior and candidate items without considering the dependency of different items. And ComiRec-SA only uses the self-attentive method to capture the multi-interest of users. Both of them cannot model the temporal interaction of items and users.

The main reason why PCAN outperforms PACA is that PACA only obtains the general preferences of users and ignores users' context information.

PCAN achieves better performances than SHAN, it may be because that SHAN does not emphasize the current preferences of users and ignores the category representation of items and users.

The main reason why PCAN outperforms BPR-MF is that BPR-MF only captures users' general preferences and does not consider the short-term interaction of items and users.

Table 8 Metrics on embedding size (ES)

Parameter	ES= 16	ES= 32	ES= 48
AUC	0.8110	**0.8164**	0.8118
$Precision$@50	0.0014	**0.0016**	0.0014
$Recall$@50	0.0698	**0.0803**	0.0706

Bold indicates the optimal experimental results

PCAN outperforms GRU4REC, probably because GRU4REC uses the dependency of subsequent items to model the evolving of user interests and only considers the current interest of users, ignoring the general preferences and the context feature of items and users. In this paper, the method utilizes the self-attention method to capture the current preferences and uses the long-term module to generate the general interest representation, while adaptively fusing the current preference and general interest through an attention-based method.

4.3 Parameter analysis

There are six hyperparameters in the PCAN model, including the size of embedding ES, the length of long-term historical sequence L_S, learning rate λ, decay rate φ, L2 regulation rate ω, and the number of heads h in the short-term self-attention module. This paper utilizes batch learning to learn the parameters and sets the batch size to 32. This subsection describes the experiments to determine the above six hyperparameters in detail.

4.3.1 Embedding size

In this subsection, we introduce in detail the experiment that determines the embedding size ES. This experiment is conducted on the Electronics dataset, and ES is searched from {16, 32, 48}. In the method, the dropout rate $\varphi = 0.0$, the long-term sequence length $L = 90$, the L2 regulation rate ω =5e−5, the learning rate $\lambda = 1.0$, and the number of heads $h = 8$. As ES increased by 16 every time, different embedding sizes are compared simultaneously. On one hand, as shown in Table 8, all metrics rise and plateau when ES is adjusted from 16 to 32. On the other hand, as we adjust ES from 32 to 48, the AUC, $precision$, and $recall$ decrease quickly. The reason is that when ES is too small, the method cannot model the potential features of users and items precisely. On the other hand, if ES is set too large, it will cause an overfitting problem. For the PCAN model, when ES is 32, the recommendation result reaches the maximum.

4.3.2 Learning rate

In this subsection, this paper introduces the adjusting process of learning rate λ. This experiment is conducted on the Office dataset, and λ is searched from {0.001, 0.01, 0.1, 0.8, 0.9, 1.0, 1.1, 1.2}. In the method, the dropout rate $\varphi = 0.0$, the long-term sequence length L = 5, the L2 regulation rate ω =5e–5, the embedding size $ES = 32$, and the number of heads h = 8. Figure 4 shows how AUC changes at different learning rates.

It can be observed from the figure that AUC increases in the range [0.001, 1.0] and decreases in the range [1.0, 1.2] with the increase in λ. Therefore, this paper set the initial learning rate $\lambda = 1.0$.

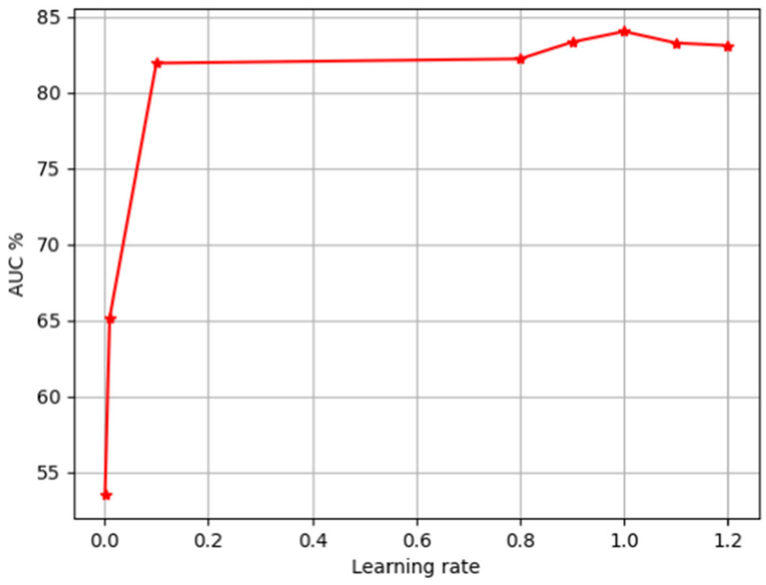

Fig. 4 AUC changes with different learning rate values

4.3.3 Dropout rate

In this subsection, this paper introduces the adjusting process of the dropout rate φ. The experiment is conducted on the Office dataset, and φ is selected from $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. In the method, the learning rate $\lambda = 1.0$, the long-term sequence length $L = 5$, the L2 regulation rate ω =5e-5, the embedding size $ES = 32$, and the number of heads $h = 8$.

It can be observed from Figs. 5 and 6 that $Precision@50$ and $Recall@50$ decreases in the range of [0.0, 0.5] with the increase in , and reach the maximum when $\varphi = 0.0$. Therefore, this paper sets the dropout rate $\varphi = 0.0$.

4.3.4 L2 regulation rate

In this subsection, this paper introduces the adjusting process of the L2 regulation rate ω. The experiment is conducted on the Office dataset, and ω is searched from {0.0, 1e−5, 2e−5, 5e−5, 6e−5, 7e−5}. In the method, the learning rate $\lambda = 1.0$, the long-term sequence length $L = 5$, the dropout rate $\varphi = 0.0$, the embedding size $ES = 32$, and the number of heads $h =$ 8. Figure 7 shows how AUC changes at different L2 regulation rates.

It can be observed from the figure that AUC increases in the range [0.0, 1e−5] with the increase in ω and decreases in the range [1e−5, 7e−5] with the increase in ω. Therefore, this paper set the L2 regulation rate $\omega =$ 1e−5.

4.3.5 The number of heads in the short-term self-attention module

In this subsection, this paper introduces the adjusting process of the number of heads h. The experiment is conducted on the Office dataset, and h is searched from {2, 4, 8, 16}. In the method, the learning rate $\lambda = 1.0$, the long-term sequence length $L = 40$, the dropout rate φ

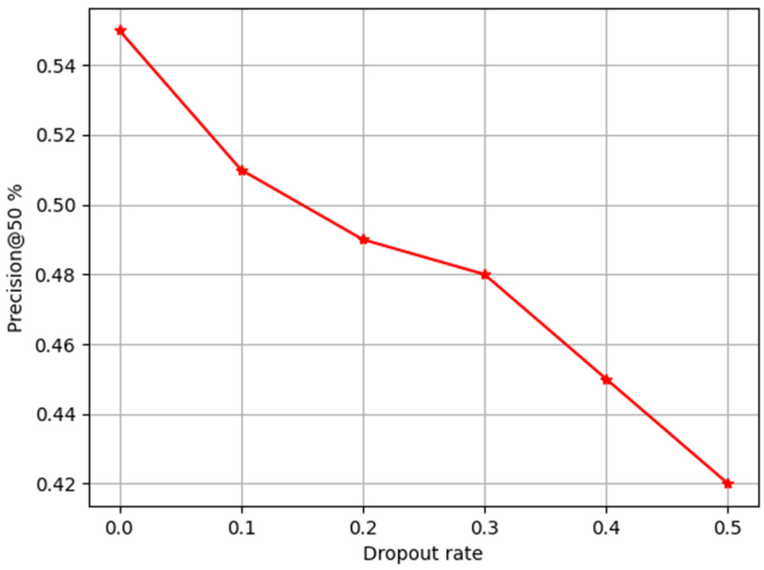

Fig. 5 $Precision$@50 changes with different dropout rate values

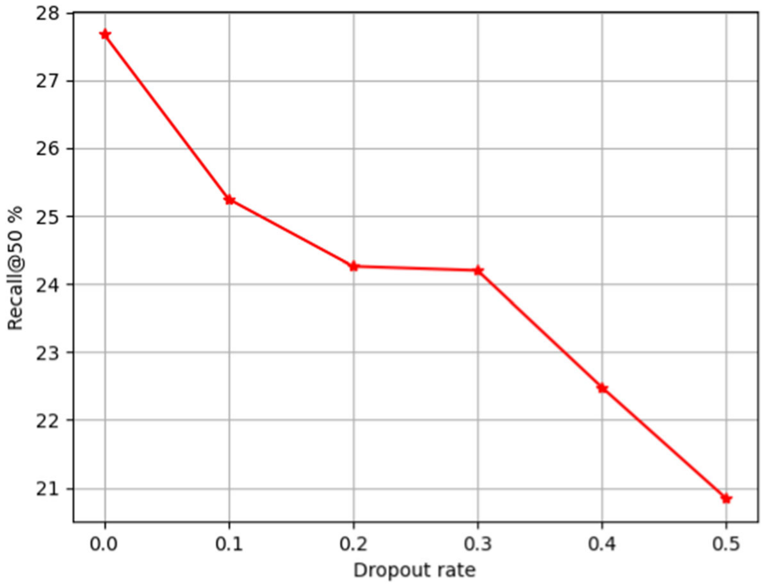

Fig. 6 $Recall$@50 changes with different dropout rate values

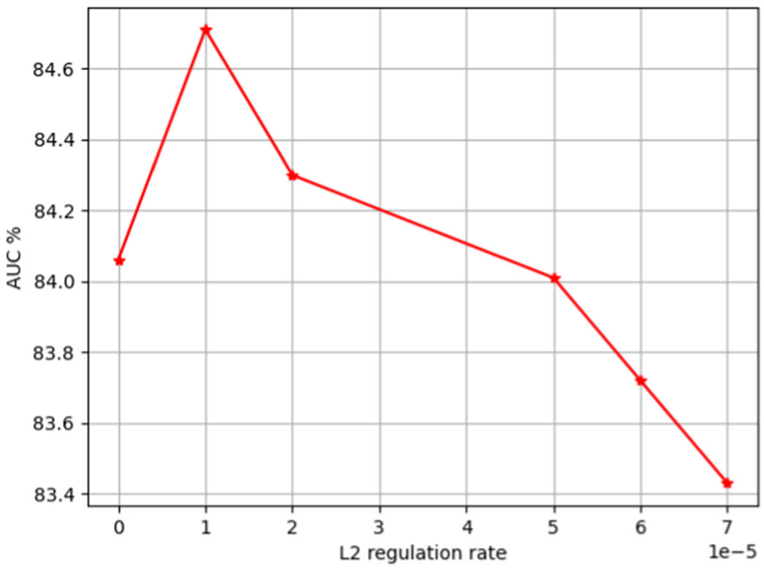

Fig. 7 AUC changes with different L2 regulation rate values

$= 0.0$, the L2 regulation rate ω =1e−5, and the embedding size $ES = 32$. Figure 8 shows how AUC changes at different numbers of heads.

It can be observed from the figure that AUC increases in the range [2, 8] with the increase in h and decreases in the range [8, 16] with the increase in h. Therefore, this paper sets the number of heads in the short-term self-attention module $h = 8$.

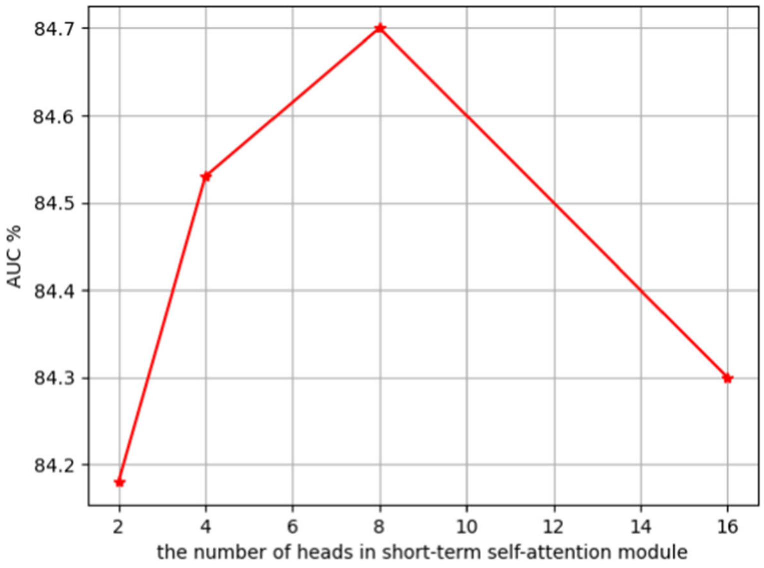

Fig. 8 AUC changes with the different numbers of heads

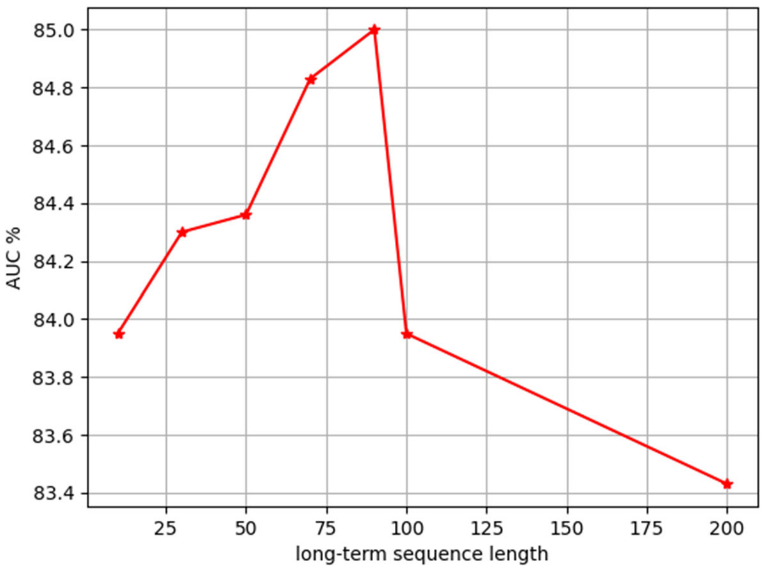

Fig. 9 AUC changes on long-term sequence length

4.3.6 Long-term sequence length

The long-term historical behaviors are also important to model user preferences. Therefore, to select the optimal long-term sequence length L from $\{10, 30, 50, 70, 90, 100, 200\}$, we conduct the comparable experiment on the Office dataset. In the method, the learning rate $\lambda = 1.0$, the dropout rate $\varphi = 0.0$, the L2 regulation rate $\omega =$1e-5, the embedding size $ES =$ 32., and the number of heads $h = 8$ As shown in Fig. 9, the AUC increases with the growth of L in the range of $\{10, 30, 50, 70, 90\}$, but it decreases quickly in the range of $\{90, 200\}$ and reaches the maximum when L is set to be 90. In Figs. 10 and 11, the $Precision$@50 and $Recall$@50 increase with the growth of L in the range of $\{10, 30, 50, 70, 90\}$, but it decreases quickly in the range of $\{90, 200\}$ and reaches the maximum when L is set to be 90. When L is set to 90, the above three metrics reach the maximum. Therefore, this paper sets the long-term sequence length L to 90.

4.4 Complexity analysis

To demonstrate the computational feasibility of our proposed model PCAN, this subsection carries out the complexity analysis and evaluates the training efficiency of PCAN and all baselines. Due to limited space, this paper only experiments on the Office dataset.

First, this paper studies the training efficiency of our PCAN model. It can be seen from Fig. 12 that the training loss decreases rapidly with the growth of the epoch. And it converges quickly after 20 epochs. These results validate that PCAN is easy to train.

After analyzing the training efficiency of our PCAN model, we conduct the complexity analysis of PCAN. Firstly, we show the number of parameters of all models in Table 9.

As shown in Table 9, TLSAN has the most parameters among all the methods, whereas our proposed method PCAN has relatively fewer parameters. TLSAN employs an attention-based

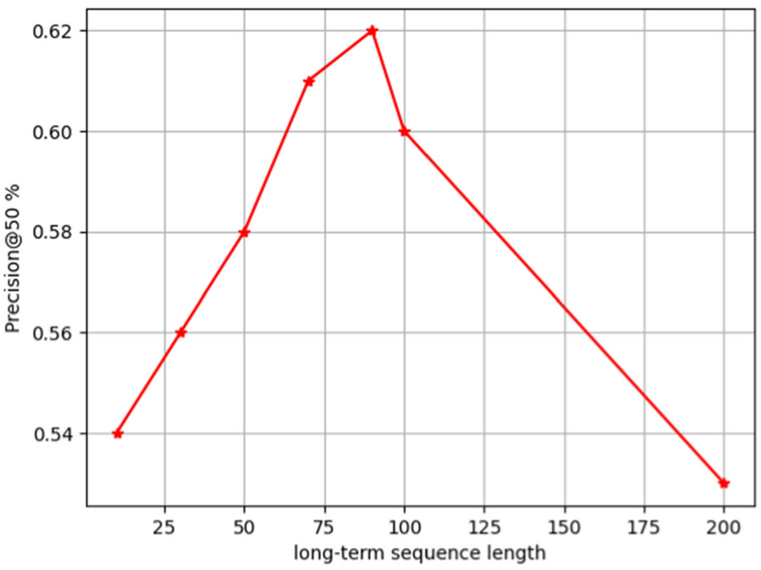

Fig. 10 $Precision@50$ changes on long-term sequence length

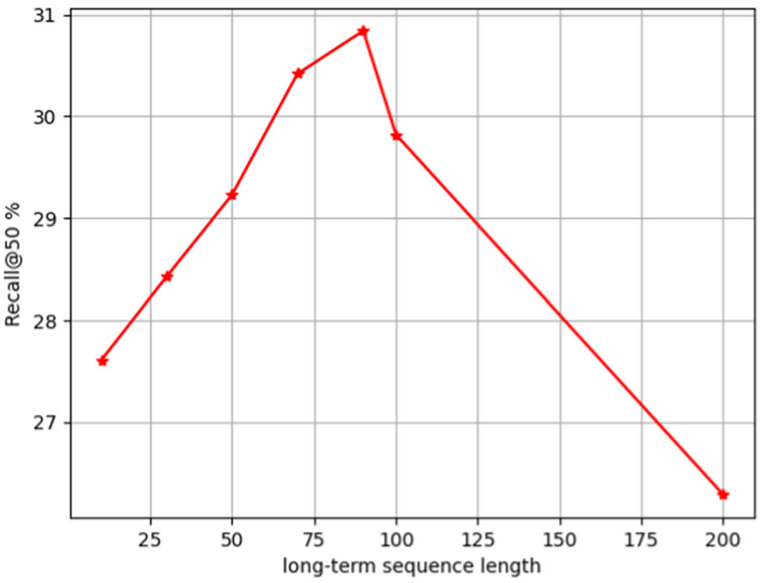

Fig. 11 $Recall@50$ changes on long-term sequence length

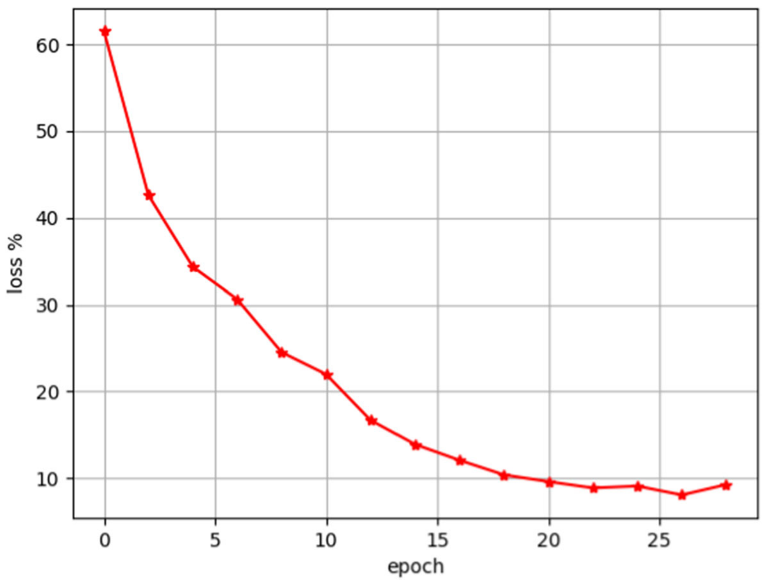

Fig. 12 Loss changes on different epoch values

Table 9 Number of parameters of all models

Models	Number of parameters
BPR-MF	145,253
GRU4REC	217,280
DIN	168,193
SHAN	86,885
PACA	34,784
ComiRec-SA	81,092
TLSAN	**376,352**
PCAN	221,398

Bold indicates the maximum number of parameters

method to separately capture the long- and short-term interest and effectively integrates the time information, category information, and item information. This method requires more parameters than other methods. Conversely, PCAN uses two different modules to capture user interest: the self-attention-based method and the position-category-aware attention method. Moreover, other methods use a single structure to capture user preference. For example, GRU4REC uses GRU to generate user interest representation, and PACA uses the traditional attention mechanism to integrate position information and item information. Both of these methods have fewer parameters than PCAN. Increasing the number of model parameters makes the model more complex but also enhances its expressive ability, which is reflected in the recommendation performance of all methods. The more comprehensive user interest vector can better express users' intent and help the model recommend appropriate items for them. For instance, TLSAN uses an attention-based method to separately consider the long-

Table 10 Computational cost comparison on Office dataset

Models	Runtime/epoch	Training epoch	Total training time
BPR-MF	1.1 s	20	22 s
GRU4REC	101.74 s	20	2034.8 s
DIN	33.59 s	20	671.8 s
SHAN	11.17 s	40	446.8 s
PACA	13.09 s	70	916.3 s
ComiRec-SA	39.85 s	20	797 s
TLSAN	17.65 s	20	353 s
PCAN	72.37 s	20	1447.4 s

and short-term interest, which needs more parameters but generates a better vector representation than those methods such as SHAN, PACA, etc. Our proposed method PCAN can generate user interest representation with fewer parameters and achieve better recommendation performance.

This paper then compares the computational cost of PCAN with all baselines. The results are shown in Table 10.

The number of TLSAN's parameters is maximum, while PACA has the minimum number of parameters. And the number of PCAN's parameters is 1.02 times that of GRU4REC. That is to say, PCAN is more complex than GRU4REC. Even then, our proposed method has a faster training speed than GRU4REC regarding training per epoch. The reason may be that the calculation of each time step depends on the calculation and output of the previous time step while training GRU4REC. And it is difficult to scale due to its inherent nature of state computation, which is difficult to parallelize. However, thanks to the computational parallelism of the attention mechanism, PCAN can be quickly trained and achieves the best performance. As for other attention-based models, just like SHAN, DIN, and TLSAN et al., the number of their parameters is almost equal to GRU4REC, but their training speed is faster than GRU4REC. Finally, although the traditional method BPR-MF can be trained fastest, this method cannot capture the deep interaction of users and items, and it has the worst recommended performance.

4.5 Component analysis

To demonstrate the effectiveness of each component in the PCAN, this subsection conducts the ablation experiment on the core modules of the proposed method. To be specific, PCAN mainly consists of the long-term attention module, dynamic user category extraction module, short-term self-attention module, and adaptive fusion module. Therefore, this paper removes the corresponding components and describes the ablation experiment. Specifically, PCAN-NL is a model that removes the long-term attention module of PCAN; PCAN-NS is a model that removes the short-term self-attention module of PCAN; PCAN-NU is a model that removes the dynamic category extraction module of PCAN; PCAN-NA is a model that removes the adaptive fusion module of PCAN, and we set a fixed weight for the captured current interests and general preferences to generate the final preference representation. The ablation experiment is conducted on the Electronics dataset. Due to space limitation, this

Table 11 Metrics on PCAN, NL (without long-term module), NU (without dynamic user category extraction module), NA (without adaptive fusion module), and NS (without short-term module)

Components	PCAN	PCAN-NL	PCAN-NU	PCAN-NA	PCAN-NS
AUC	**0.8164**	0.7906	0.8100	0.8141	0.7764
$Precision@50$	0.0016	0.0014	0.0015	0.0014	**0.0017**
$Recall@50$	0.0803	0.0686	0.0726	0.0689	**0.0836**

Bold indicates the optimal experimental results

paper just shows the experimental performances under the metric of AUC, $Precision@50$, and $Recall@50$.

Table 11 shows that those incomplete models' recommendation performance is not as good as PCAN, and each component of the model can improve the recommendation performance effectively. To be specific, this paper will explain the following four aspects.

First, according to the experiment results of the PCAN and PCAN-NL methods, considering the long-term users' interests can effectively improve the recommendation performance. The PCAN method improves the AUC, $Precision@50$, and $Recall@50$ on the Electronics dataset by 3.26%, 14.29%, and 17.06%, respectively. Second, according to the experiment results of the PCAN and PCAN-NS methods, PCAN-NS performs a little better than PCAN at precision@50 and recall@50, but the result of AUC is the worst compared to other models. This indicates that the recent intent of users may make noise to the final recommendation, but adding the short-term self-attention module to PCAN-NS can improve the overall performance. Third, according to the experimental performance of the PCAN and PCAN-NU methods, it can be seen that considering the dynamic user category also improves the recommendation performance. The PCAN method improved the AUC, $Precision@50$, and $Recall@50$ by 0.79%, 6.67%, and 10.61%, respectively. Finally, according to the experimental performance of the PCAN and PCAN-NA models, it can be concluded that utilizing the adaptive fusion module can improve the recommendation performance effectively. And the PCAN model increases the AUC, $Precision@50$, and $Recall@50$ by 0.28%, 14.29%, and 16.55%, respectively, on the Electronics dataset.

5 Conclusions

To sum up, a position-category-aware attention network called PCAN is proposed by us. Specifically, this method first utilizes the embedding module to generate the high-dimensional dense vector representation of all features. Then, the method utilizes the long-term and short-term components to capture the corresponding preference representation. Finally, an adaptive fusion method is constructed to dynamically adjust the weight of two kinds of preferences representation under specific circumstances. The method considers the dynamic user preferences while modeling the high-level interaction between users and items. From the experiments, the PCAN model has achieved better recommendation results than other baselines on five real datasets according to AUC, $Precision$, and $Recall$.

In the future, we plan to continue to study the PCAN method, hoping to integrate more auxiliary information such as time interval, rating information, and comments, to improve the recommendation performance. In addition, the future method will further exploit the information of users and items for better recommendation.

Acknowledgements We thank the handling editors and reviewers for their effort and constructive expert comments. This work has been supported by the Shandong Nature Science Foundation of China (No. ZR2020MF044).

Author contributions Dear Editor, The personal contributions of all authors are as follows: Mingjian Dou: Conceptualization, Methodology, Software, Writing- Original draft preparation Liqing Qiu: Supervision, Data curation Caixia Jing: Investigation, Writing- Reviewing and Editing, Software Yuying Liu: Visualization, Validation

Declarations

Conflict of interest The authors declare no competing interests.

References

1. He P, Wu H, Zeng C, Ma Y (2019) Truser: an approach to service recommendation based on trusted users. Chin J Comput 42(4):851–863
2. Dong J, Sun F, Wu T, Wu X, Zhang W, Wang S (2022) A hierarchical network with user memory matrix for long sequence recommendation. Wirel Commun Mob Comput. https://doi.org/10.1155/2022/5457044
3. Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on world wide web, pp 811–820
4. He R, McAuley J (2016) Fusing similarity models with markov chains for sparse sequential recommendation. In: 2016 IEEE 16th international conference on data mining (ICDM). IEEE, pp 191–200
5. Zha Y, Zhang Y, Liu Z, Dong Y (2022) Self-attention based time-rating-aware context recommender system. Comput Intell Neurosci. https://doi.org/10.1155/2022/9288902
6. Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2015) Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939
7. Cao Y, Zhang W, Song B, Pan W, Xu C (2019) Position-aware context attention for session-based recommendation. Neurocomputing 376:65–72
8. Wu, C.Y., Ahmed, A., Beutel, A., Smola, A.J., Jing, H.: Recurrent recommender networks. In: Tenth ACM international conference on web search data mining (2017)
9. Quadrana M, Karatzoglou A, Hidasi B, Cremonesi P (2017) Personalizing session-based recommendations with hierarchical recurrent neural networks. In: ACM
10. Jannach D, Ludewig M (2017) When recurrent neural networks meet the neighborhood for session-based recommendation. In: Eleventh ACM conference on recommender systems
11. Tang J, Ke W (2018) Personalized top-n sequential recommendation via convolutional sequence embedding. In: ACM
12. Li J, Wang Y, Mcauley J (2020) Time interval aware self-attention for sequential recommendation. In: WSDM '20: The Thirteenth ACM international conference on web search and data mining
13. Xiao, J., Ye, H., He, X., Zhang, H., Wu, F., Chua, T.S.: Attentional factorization machines: Learning the weight of feature interactions via attention networks. arXiv:1708.04617 (2017)
14. Tan Q, Zhang J, Liu N, Huang X, Hu X (2021) Dynamic memory based attention network for sequential recommendation. In: Proceedings of the AAAI conference on artificial intelligence
15. Cen Y, Zhang J, Zou X, Zhou C, Yang H, Tang J (2020) Controllable multi-interest framework for recommendation. In: Knowledge discovery and data mining
16. Kang WC, Mcauley J (2018) Self-attentive sequential recommendation. In: 2018 IEEE international conference on data mining (ICDM)
17. Zhang S, Tay Y, Yao L, Sun A, An J (2019) Next item recommendation with self-attentive metric learning. In: Thirty-Third AAAI conference on artificial intelligence, vol. 9
18. Zhou G, Song C, Zhu X, Fan Y, Zhu H, Ma X, Yan Y, Jin J, Li H, Gai K (2017) Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining
19. Xu, W., He, H., Tan, M., Li, Y., Lang, J., Guo, D.: Deep interest with hierarchical attention network for click-through rate prediction. arXiv (2020)
20. Ying H, Zhuang F, Zhang F, Liu Y (2018) Sequential recommender system based on hierarchical attention network. In IJCAI international joint conference on artificial intelligence

21. Zhang MH (2018) Stamp: Short-term attention/memory priority model for session-based recommendation. SIGKDD explorations (Udisk). In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining
22. Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P (2019) BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management
23. Wu L, Li S, Hsieh C, Sharpnack J (2020) Sse-pt: Sequential recommendation via personalized transformer. In: RecSys '20: Fourteenth ACM conference on recommender systems
24. Zhang J, Wang D, Yu D (2021) Tlsan: time-aware long- and short-term attention network for next-item recommendation. Neurocomputing 441:179–191
25. Niu L, Peng Y, Liu Y (2021) Deep recommendation model combining long- and short-term interest preferences. IEEE Access 9:166455–166464
26. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. arXiv (2017)
27. Zhou C, Bai J, Song J, Liu X, Zhao Z, Chen X, Gao J (2017) Atrank: an attention-based user behavior modeling framework for recommendation. In: Proceedings of the AAAI conference on artificial intelligence
28. Huang X, Qian S, Quan F, Sang J, Xu C (2018) Csan: contextual self-attention network for user sequential recommendation. In: 2018 ACM multimedia conference (2018)
29. Yu Z, Lian J, Mahmoody A, Liu G, Xie X (2019) Adaptive user modeling with long and short-term preferences for personalized recommendation. In: International joint conference on artificial intelligence
30. Papagelis M, Plexousakis D (2005) Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. Eng Appl Artif Intell 18(7):781–789
31. Yang X, Steck H, Yong L (2012) Circle-based recommendation in online social networks. In: ACM SIGKDD international conference on knowledge discovery and datamining
32. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
33. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition
34. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) Bpr: Bayesian personalized ranking from implicit feedback. In: UAI 2009, Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, Montreal, QC, Canada, June 18–21

Liqing Qiu received the PHD degree from Beihang University in 2008. She is currently an associate professor at Shandong University of Science and Technology. Her current research focuses on social networks and knowledge tracking.

Mingjian Dou obtained a bachelor's degree from Henan University of Engineering in 2021. He is currently a postgraduate with Shandong University of Science and Technology. His research interests include social networks and recommendation system.

Caixia Jing obtained a master's degree in Computer Science and Technology from Shandong University of Science and Technology in 2023. Her research interests include social networks, deep learning and click-through rate prediction.

Yuying Liu obtained a bachelor's degree from Shandong First Medical University in 2020 and a master's degree in Computer Science and Technology from Shandong University of Science and Technology in 2023. Her research interests include social networks and machine learning.