Deep recommendation model combining long- and short-term interest preferences

LUSHUAI NIU, YAN PENG, (Member, IEEE) AND YIMAO LIU
College of Automation and Information Engineering, Sichuan University of Science and Engineering, Yibin 644000, China

Corresponding author: Yan Peng(2634932795@qq.com).

This work was supported in part by the Key research and development Project of Science and Technology Department of Sichuan Province, No.19ZDYF1078; Science and Technology Plan of Zigong Science and Technology Bureau, No.2018GYCX33; Key Laboratory of Enterprise Informationization and Internet of Things Measurement and Control Technology in Sichuan Province Universities, No. 2021WYJ04.

Abstract The existing sequential recommendation algorithms cannot effectively capture and solve the problems such as the dynamic preferences of users over time. This paper proposes a deep Recommendation model CLSR (Combines Long-term and Short-term interest Recommendation) that Combines long-term and short-term interest preferences. Firstly, the model models the potential feature representation of users and items, and uses the self-attention mechanism to capture the relationship between items in the interaction of users' historical behavior, so as to better learn the short-term interest representation of users. At the same time, the BiGRU network is used to extract the features of users' long-term interests on a deep level. Finally, the features of long-term and short-term interest are fused. On four publicly available datasets, experimental results show that the proposed method has better improvement on HR@N, NDCG@N and MRR@N, which validates the effectiveness of the model.

Index terms Recommendation algorithm, Self-attention mechanism, Bidirectional gating cyclic network, Sequence recommendation, Deep learning.

I. INTRODUCTION

With the rapid development of big data technology, the problem of information overload [1] is becoming more and more serious, and the problem of how to make users efficiently obtain information of interest has become the main research content of recommendation systems [2].

In recent years, recommendation systems using deep learning have been the main research direction [3]. Compared with traditional recommendation models, deep learning recommendation models are more capable of fitting data patterns and mining feature combinations, and the deep learning model structure is very flexible and can adjust the model according to different recommendation scenarios to make it fit perfectly with the special application scenarios.

Among the currently used deep learning models, for example, the deep learning-based collaborative filtering model NeuralCF proposed by He et al. [4], which replaces the inner product operation in matrix decomposition with a neural network, giving the model a stronger feature combination and nonlinear capability. Cheng et al. [5] proposed a hybrid model Wide& Deep, which consists of a single-layer Wide part and a multi-layer Deep part, pioneering the construction of a combined model with improved generalization and memory capabilities. Xiao et al. [6] proposed AFM, a factorization model incorporating an attention mechanism, reflecting the different importance of different crossover features. zhou et al. [7] proposed a model incorporating two-headed attention and self-encoder for learning representation from user comments and implicit feedback. Chen et al. [8] proposed a ranking recommendation using neural networks that captures the latent features of users and items in complex nonlinear structures and is easily scalable. These efforts focus on mining static correlations between users and items, which can capture users' long-term preferences and static behaviors, but ignore the decay of user interests and dynamic changes in preferences implied by the sequence of user interactions formed over time.

Sequential recommendation models have powerful time series representation capabilities, making them well suited to predict users' next action after a series of behaviors. Rendle et al. [9] successfully applied Markov chains to short sequence recommendation models (FPMC), and Smirnova et al. [10] used recurrent neural networks to predict users' next action, both of which can capture users' short-term interest preferences but ignore that the features of items also change.
dynamically with interactions. Kang et al. [11] proposed SASRec model, which uses a self-attention mechanism to model users’ historical behavioral information to extract more valuable information, and finally the obtained information is inner-producted with the embedding content of all items separately to get Top-K recommendations, but it only focuses on users’ short-term preferences and ignores long-term interest preferences. Tang et al. [12] proposed a sequential recommendation model (Caser), which focuses on extracting information of short-term sequences through convolutional neural networks, but does not deeply extract users’ generic preferences. Zhang et al. [13] proposed a model (AttRec) to capture the item-to-item relationships in users’ historical behavioral interactions using a self-attention mechanism, so as to better learn the representation of users’ short-term interests and use metric learning to model users’ long-term preferences. However, their approach is relatively simple and does not represent long-term interests well. Lin et al. [14] proposed FISSA, a sequence recommendation model combining item similarity and self-attention mechanism, to calculate the weight of local representation and global representation by modeling item similarity between candidate item and recently interacting item as well as similarity between candidate item and historical behavior item. Wu et al. [15] proposed a personalized Transformer model with Random Shared Embedding to address the lack of personalized representation of SASRec, using Transformer as the subject and adding Stochastic Shared Embeddings regularization technique for sequential recommendation, they are all long-term interest preference expression is slightly inadequate. the time interval self-attention model TiSASRec proposed by Li et al. [16], the context-based temporal attention mechanism model CTA proposed by Wu et al. [17], and the MEANTIME model proposed by Cho et al. [18], are all for most self-attention models simply extract a kind of sequential relationship of users, ignoring the information of time and context and therefore consider from the perspective of timestamp to extract additional contextual information for recommendation. The focus is on the effective extraction of short-term interests. The multi-interest recommendation model ComiREC proposed by Cen et al. [19] mainly focuses on capturing multiple interests from user behavior sequences to reflect recommendation diversity, but cannot effectively learn the temporal characteristics of users and items.

To address the above problems, this paper proposes a deep recommendation model (CLSR) that fuses long- and short-term interest preferences. The model first extracts the short-term interest preferences of users through the self-attention mechanism, while effectively builds long-term interest features based on the BiGRU model, and finally fuses the long-term and short-term interest features to obtain the final recommendation results by weighted summation.

The remainder of our work is organized as follows. We make a brief review about the bi-directional gated recurrent network as well as the self-attention mechanism in Section 2. Section 3 first lists the notations we used in this paper, and then describes our proposed CLSR in detail. Comprehensive experiments will be performed in Section 4 to illustrate the performance of CLSR. Section 5 makes a conclusion of this paper.

II. RELATED WORK

A. BIDIRECTIONAL GATED RECURRENT NETWORK

GRU [20] is a highly effective variant of the long short-term memory network (LSTM) [21], which has a simpler structure and lower computational cost compared to the LSTM network, preserves long-term sequence information while reducing the gradient disappearance problem, and outperforms the standard LSTM network even more on multiple benchmark datasets.

The GRU network has only update and reset gates, discarding the forgetting gates in the LSTM network. The reset gate determines how the new input information is combined with the previous state information, the update gate is used to control how much of the previous state information is brought into the current state, and the two gating vectors determine which information can eventually be used as the output of the gated loop unit, which not only preserves the information in the long-term sequence, but also does not clear it over time or remove it because it is not relevant to the prediction. GRU network structure is shown in Figure 1.

The mathematical expressions of the states of the two gates at moment t are shown in Eqs. (1) to (4).

\[
\begin{align*}
    z_t &= \sigma \left( W^{(c)} x_t + U^{(c)} h_{t-1} \right) \\
    r_t &= \sigma \left( W^{(r)} x_t + U^{(r)} h_{t-1} \right) \\
    h_t' &= \tanh \left( W x_t + r_t \odot U h_{t-1} \right) \\
    h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot h_t'
\end{align*}
\]

where \( x_t \) is the input vector at the \( t \)th time step, i.e., the \( t \)th component of the input sequence \( x \). \( h_{t} \) holds the information from the previous time step \( t-1 \), \( z_t \) and \( r_t \) are the update gate and reset gate, respectively. \( W \) and \( U \) are the corresponding weight matrices, respectively, \( h_t' \) is the memory content of the current output, \( \odot \) represents the Hadamard product, i.e., the corresponding elements are multiplied together, and \( h_t \) is the final gated loop unit output.
Although GRU can better capture long-term sequence information, the sequence information is simply front-to-back, and Zheng et al. [22] pointed out that GRU has problems such as exponential decay of long-term information and this information will be quickly forgotten, so bidirectional sequence information capture is needed to better obtain contextual information. The bidirectional GRU network, namely BiGRU[23], is adopted in this paper, and the final output expression is shown in Equation (5).

\[
H_t = \begin{bmatrix}
\overrightarrow{h_t}; \overleftarrow{h_t}
\end{bmatrix}
\]  

(5)

where \(\overrightarrow{h_t}\) denotes the forward hidden state, \(\overleftarrow{h_t}\) denotes the reverse hidden state, and \(H_t\) contains the contextual information of the whole sequence.

### B. SELF-ATTENTION MECHANISM

The attention mechanism is derived from the most natural human habit of selective attention, which can efficiently extract the focus in the whole picture, and has been widely applied to natural language processing, speech recognition, and other fields in recent years. Attention mechanisms are also widely used in recommender systems, for example, Yakhchi et al. [24] proposed a novel deep attention-based sequential model that uses attention mechanisms to distinguish the importance of items in long-term and short-term user preferences. Chen et al. [25] proposed a co-occurrence model-based recommendation algorithm that models users and items and between items and items, and incorporates attention networks to learn users’ personalized weight preferences to effectively encode highly descriptive features. Xia et al. [26] proposed a knowledge-enhanced hierarchical graph transformation network that uses graph-structured neural networks to capture specific types of behavioral features and combines self-attention mechanisms and temporal coding to learn more multiple types of user-item and item-item relationships. Cai et al. [27] proposed a category-aware collaborative sequential recommendation model that uses a self-attentive mechanism to capture excessive patterns within categories and decides which transition patterns within categories need to be considered based on the categories of recent actions. Jiang et al. [28] proposed a knowledge representation learning model based on a self-attentive mechanism that uses knowledge graphs and a self-attentive mechanism to represent entity learning features to address the problem of under-representation of semantic features.

The self-attention mechanism [29] is an improvement of the attention mechanism, which reduces the influence of the external information it is subject to and is better at capturing the relevance within the user behavior sequence. The input of the self-attention mechanism consists of Q, K and V, where \(Q=K=V\). The purpose is for the model to better learn all direct dependencies of each item with all other items and capture the internal structure of the behavioral sequence. First, the dot product of Q and K is calculated and divided by the root K. The weight distribution on V is obtained by softmax, and finally the weighted value of V is calculated by dot product. The output of the calculated matrix is shown in equation (6).

\[
\text{Attention}(Q,K,V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_i}} \right) V
\]

(6)

where softmax is the activation function and \(d_i\) is the scaling factor.

### III. CLSR MODEL

![FIGURE 2. CLSR model structure diagram.](attachment:image.png)
A. PRELIMINARIES

In this paper, the set of users \( U = \{ u_1, u_2, ..., u_n \} \) and the set of items \( I = \{ i_1, i_2, ..., i_m \} \). \( X^u = (X^u_1, X^u_2, ..., X^u_{L}) \) denotes the sequence of items in which a user has previously interacted, \( X^u_i \in I \), and the subscript index \( i \) of \( X^u_i \) indicates the order in which item interactions appear in the sequence. The purpose of the model is to predict the next item with which a user may interact based on the user's historical behavior. Table I lists the key notations used in this paper.

| Symbols | Description |
|---------|-------------|
| \( n \) | Number of users/items |
| \( X^u \) | Historical interaction sequence of user \( u \) |
| \( X^u_t \) | Order of item interaction |
| \( E_i \) | User embedding matrix |
| \( E_x \) | Item embedding matrix |
| \( E_{x} \) | User behavior sequence embedding matrix |
| \( k \) | Embedding dimension |

The structure of the proposed deep recommendation model CLSR (Combines Long-term and Short-term interest Recommendation) that fuses long- and short-term interest preferences is shown in Figure 2. The model is divided into an embedding layer, a self-attention layer, a BiGRU layer, and a fused output layer. The self-attention layer uses a self-attention mechanism to capture the user's short-term interest (i.e., Self-A), and the BiGRU layer uses a multi-layer Bi-GRU network to capture the user's long-term interest (i.e., M-Bi-GRU). And the item embedding vectors in the input of the two modules are not shared.

B. EMBEDDED LAYER

The inputs of the input layer model are preprocessed to generate user feature columns, user-item interaction sequences and item feature columns. Since the sample feature vectors have the problem of high-dimensional high sparsity, the next step of the input layer needs to go through the embedding layer responsible for converting the high-dimensional sparse feature vectors into dense low-dimensional feature vectors, so as to better obtain the feature representations of users and items. The user embedding matrix is \( E_u \in \mathbb{R}^{n \times k} \), the item embedding matrix is \( E_i \in \mathbb{R}^{m \times k} \), and the embedding matrix of the user behavior sequence is \( E_x \in \mathbb{R}^{L \times k} \). \( e_u, e_i, e_x \) denote the embedding vector representations of the user, item, and behavior sequences, respectively.

C. SHORT-TERM INTEREST MODULE

1) SELF-ATTENTION LAYER

Users' recent behaviors will reveal their behavioral tendencies or temporary needs. In short-term interest modeling, a self-attention mechanism is used to capture sequential features from users' short-term behavioral history to fully understand and express their short-term interests.

First, the user's short-term interest is obtained from the L items they have recently interacted with, and the user's recent L interactions are derived from the embedding vector of all items, as shown in Equation (7).

\[
X_i^u = \left[ X_{i-1}^u, ..., X_i^u, X_{i+1}^u \right] \quad (7)
\]

It is then subjected to the embedding operation to obtain the embedding matrix \( E_x^u \in \mathbb{R}^{L \times k} \), where \( L \) represents the set of items.

The input of the self-attention layer contains three parts: query, key, and values, and the values of all three are composed of user history behavior features, i.e., \( E_x^u \), query and key are nonlinearly transformed to obtain a similarity matrix. The formula is shown in equation (8).

\[
f_t^u = \text{softmax} \left( \frac{QK^T}{\sqrt{d}} \right) \quad (8)
\]

where \( f_t^u \in \mathbb{R}^{L \times L} \), denotes the similarity between the L items.

The similarity matrix of historical interaction items is multiplied with values to form the weighted output of the final self-attention mechanism. In order to learn a single attention representation, the self-attention of average \( L \) users eventually forms the short-term interest preference of users. The formula is shown in equation (9).

\[
m_t^u = \frac{1}{L} \sum_{l=1}^{L} f_t^u E_x^u_l \quad (9)
\]

Due to the self-attention mechanism itself loses location information when processing user behavior sequences and cannot establish a temporal relationship, the location embedding \( P_i \in \mathbb{R}^{n \times k} \) is chosen to represent a sequential location relationship in the sequence, which is summed with the embedding matrix of historical behavior sequences as the input of the self-attention layer, and the location embedding formula is shown in equation (10).

\[
E = \{ E_i + P_i \} \quad (10)
\]

where \( E_i \) denotes the \( t \)th historical behavioral feature of the user and \( P_i \) denotes the order of the \( t \)th behavioral interaction.

Before the nonlinear transformation of query and key, the query and key with time information are provided by positional embedding, as shown in Equations (11) and (12).

\[
P_{(\text{pos}, 2i)} = \sin \left( \frac{\text{pos} \times 10000^i}{2^{i \text{dim}_{\text{model}}}} \right) \quad (11)
\]

\[
P_{(\text{pos}, 2i+1)} = \cos \left( \frac{\text{pos} \times 10000^{2i+1}}{2^{i \text{dim}_{\text{model}}}} \right) \quad (12)
\]

where \( \text{pos} \) denotes the position of the item in the sequence and \( i \) represents the dimension of the embedding matrix.

Eventually, through the calculation of the self-attention layer, the short-term behavioral history sequence of the user is converted into a deeper representation that reflects the short-term interest \( m_t^u \) of the current user.
D. LONG-TERM INTEREST MODULE

1) BIGRU LAYER

Short-term intentions alone are not enough to express the complete user interest preferences; long-term preferences are also very important for the portrayal of users, and it is often better to combine the long-term user preferences. The inner product of the user embedding vector $e_u$ and the item embedding vector $e_i$ is used as input to the BiGRU layer to model the user-item interaction. To make more use of contextual information and to achieve deep feature mining, a multi-layer bi-directional GRU network is used to capture the deep feature information of the time series.

Firstly, the feature vectors of users and items are input into the first layer of BiGRU network, and the equations are shown in equations (13–17).

$$z_t = \sigma(W^{(z)}(e_u \cdot e_i) + U^{(z)}h_{t-1})$$  \hspace{1cm} (13)

$$r_t = \sigma(W^{(r)}(e_u \cdot e_i) + U^{(r)}h_{t-1})$$  \hspace{1cm} (14)

$$h_t^{'} = tanh(W(e_u \cdot e_i) + r_t \odot Uh_{t-1})$$  \hspace{1cm} (15)

$$h_t = z_t \odot h_t^{'} + (1 - z_t) \odot h_t^{'}$$  \hspace{1cm} (16)

$$H_t = [h_t^{'}; h_t]$$  \hspace{1cm} (17)

When both the forward and reverse GRU networks have completed all the time series information extraction, then move to the next layer of bidirectional GRU and continue to repeat the above steps until the output of the last layer of bidirectional GRU is derived, which is the probability of whether the end user interacts with the target item or not. The formula is shown in equation (18).

$$M_t = Bi-GRU <e_u \cdot e_i>$$  \hspace{1cm} (18)

where $M_t$ is a representation of the user's long-term interest preferences.

E. FUSION OUTPUT LAYER

When making recommendation predictions, users' long-term and short-term interests present different degrees of dominance, and fusing the two can better represent the final user preferences. In this paper, Euclidean distance is used for the fusion between long and short interests, and the final recommendation score of candidate items is obtained by weighted sum, as shown in Formula (19).

$$y_{t+1}^u = w\|M_t\|^2 + (1-w)\|m_u^t - X_{t+1}^u\|_2$$  \hspace{1cm} (19)

The purpose is to predict the items that user $u$ is likely to interact with at time $t+1$, where $M_t$ denotes the user's long-term preference, $m_u^t$ denotes the short-term preference, $X_{t+1}^u$ denotes the next item's embedding vector, and $w$ is the weight parameter.

In this paper, the pairwise ordering method is used to learn the model parameters, and the hinge loss function is used as the objective function. The model parameters are learned by minimizing the objective function. The formula is shown in equation (20).

$$L = \sum_{(u,i) \in N^+} (y_{i}^u - y_{j}^u) + \lambda \|\theta\|^2_2$$  \hspace{1cm} (20)

where $N^+$ is the positive sample, which denotes the set of items with which users interacted, and $N^-$ is the negative sample, which denotes the set of items with which users did not interact. $\lambda$ denotes the regularization factor of $L_2$ and \(\theta\) denotes the parameters of the model.

IV. EXPERIMENT

A. DATASETS

The model proposed in this paper is experimented on MovieLens-100K, MovieLens-1M, Amazon Product, and Yelp datasets. MovieLens [30] is a movie rating dataset that is widely used to study the performance of recommendation algorithms. The MovieLens-100K dataset contains more than one hundred thousand ratings for more than one thousand movies from more than nine hundred users. Each anonymous user in MovieLens-1M makes at least 20 ratings, containing a total of more than one million ratings data. Amazon Product is a subset of selected product reviews from Amazon. It contains 6170 users and 2753 products with 195791 ratings on a scale of 1 to 5. Yelp is a well-known user review site in the US. The dataset includes 16,239 users and 14,284 movies with 198,397 ratings, ranging from 1 to 5. The dataset details are shown in Table 2.

| Dataset          | Users | Items | Comments | Density |
|------------------|-------|-------|----------|---------|
| MovieLens-100K   | 943   | 1682  | 100000   | 6.33%   |
| MovieLens-1M     | 6040  | 3883  | 1000209  | 4.26%   |
| Amazon Product   | 6170  | 2753  | 195791   | 1.15%   |
| Yelp             | 16239 | 14284 | 198397   | 0.09%   |

In the experiments of this paper, the rating data of specific values are first transformed into implicit feedback (i.e., users interact with items), and user ID and timestamps are used to rank the order of user-item interactions. For positive and negative sample classification, those with ratings greater than or equal to 1 are considered as positive samples, and those without ratings are treated as negative samples. For a positive sample in the test, 100 negative samples are randomly selected in order to reduce the computational effort of the model. The interaction probability of each sample is obtained through model evaluation and sorted, and the list of top-K items is finally obtained.

B. EVALUATION INDICATORS

In order to verify the effectiveness of the CLSR model, this paper uses Top-K evaluation metrics to measure the recommendation performance of the model, including Hits Ratio (HR), Normalized Discounted Cumulative Gain (NDCG) and Mean Reciprocal Rank (MRR). HR indicates...
whether the items in the test set will appear in the final recommendation list, NDCG indicates the position of the hit items in the recommendation list, and MRR indicates the position rank of the items that users actually clicked in the recommendation list. The higher the value of the three metrics, the better the recommendation performance.

HR, NDCG and MRR are defined as shown in equations (21), (22) and (23).

$$HR@K = \frac{\text{Number of Hits @ K}}{|GT|}$$

(21)

$$NDCG@K = Z_K = \sum_{i=1}^{K} \frac{2^r - 1}{\log_2 (i+1)}$$

(22)

$$MRR@K = \frac{1}{Q} \sum_{i=1}^{Q} \frac{1}{\text{rank}_i}$$

(23)

where K denotes the number of items recommended, GT represents all test sets, and represents the relevance of the recommendation result where the position of the item is i, and the value of item i is 1 if it is in the recommendation list, otherwise the value is 0. Zk is the regularization. Q is the number of users. It indicates the position of the first valid value in the recommendation list for the ith user, and is less than or equal to K. Otherwise the value of MRR@K is 0.

C. COMPARISON METHODS

In order to evaluate the validity of the model proposed in this paper, it is compared experimentally with the following related classical models.

1) BPRMF [31]

This method is one of the widely used matrix decomposition methods that use Bayesian personalized ranking loss to optimize the matrix decomposition model.

2) FPMC

By combining matrix decomposition and Markov chains to capture users' long- and short-term preferences and predict the next suggestion.

3) GRU4REC[32]

A session-based serialization prediction model, which uses RNN to model the interaction sequence between users and items.

4) SASREC

Self attention mechanism is used to model user history behavior information. Inner product the modeling information with all item embedding vectors separately.

5) TRANSREC[33]

Modeling the third-order relationships between user items and proposing item-to-item recommendations.

6) CASER

A sequence recommendation model that uses convolutional neural network to extract short-term preference information, and learns users’ sequence information through vertical convolution and horizontal convolution.

7) ATTREC

Using self-attention mechanism and metric learning to capture users' short-term interest and long-term interest preferences, respectively.

D. EXPERIMENTAL SETTINGS

For the experiments of all methods, the learning rate is set to 0.01, the embedding dimension is 100, the sequence length is 5, the depth of the BiGRU network is 2, the batch size is set to 1024, the $L_2$ regularization coefficients are chosen between [1e-6, 1e-5, 1e-3], and the model is optimized using the Adam function. All models are based on TensorFlow 2.0 and Python 3.6, and experiments are performed using a Tesla GPU with 36G of video memory and NVIDIA.

E. EXPERIMENTAL RESULTS AND ANALYSIS

In order to demonstrate the validity of the CLSR model, this paper will experimentally analyze it from three aspects. First, the model CLSR proposed in this paper is evaluated with five comparison methods on two datasets by HR@10, NDCG@10 and MRR@10; Secondly, the experimental comparison of the CLSR model with its two variants is conducted; Finally, the effect of sequence length on the model CLSR is analyzed for different embedding dimensions.

1) COMPARISON OF ALGORITHM RESULTS ANALYSIS

The experimental results of all methods under the four datasets are shown in Table 3, and Table 4, respectively.

| Method | MovieLens-100K | MovieLens-1M | MovieLens-1M |
|--------|---------------|--------------|--------------|
|        | HR@10         | NDCG@10      | MRR@10       | HR@10         | NDCG@10      | MRR@10       |
| BPRMF  | 0.4017        | 0.2466       | 0.0946       | 0.4515        | 0.3816       | 0.1821       |
| FPMC   | 0.5124        | 0.3271       | 0.1875       | 0.6437        | 0.5941       | 0.2746       |
| GRU4Rec| 0.5014        | 0.3369       | 0.2139       | 0.6514        | 0.6083       | 0.3198       |
| TransRec| 0.4927       | 0.3944       | 0.1748       | 0.5346        | 0.6145       | 0.2846       |
| SASRec | 0.5695        | 0.3987       | 0.1674       | 0.6561        | 0.5987       | 0.2991       |
| Caser  | 0.4971        | 0.3861       | 0.1599       | 0.6751        | 0.6432       | 0.2574       |
| AttRec | 0.5535        | 0.4252       | 0.2357       | 0.7485        | 0.6975       | 0.4139       |
| CLSR   | 0.5864        | 0.4457       | 0.2461       | 0.7514        | 0.7200       | 0.4308       |
From Table 3 and Table 4, the following observations can be drawn:

The model CLSR outperforms all other comparison methods on all four datasets, verifying the validity of its recommended performance.

The main reason why the experimental results of model CLSR outperform AttRec is that AttRec uses metric learning to model users' long-term interests, which is too simple to adequately learn users' long-term preferences and cannot explicitly consider the relationship between items.

The possible reasons for CLSR to outperform Caser and SASRec are that Caser only uses convolutional neural networks to learn group-level representations of continuous items without considering the item importance of different users, and SASRec only uses a self-attention mechanism to model users' historical behavioral information, both of which focus too much on users' short-term preferences and ignore their long-term preferences.

CLSR achieves better results than GRU4Rec, the possible reason GRU4Rec only considers the transition relationship from the previous node to the current node and only considers the current interest of the user, ignoring the long-term interest.

The main reason why CLSR outperforms FPMC and TransRec is because the FPMC model only models first-order Markovian connections, TransRec focuses on modeling third-order relationships between users and items, and CLSR captures the higher-order relationships that exist between items, capturing more complex relationships.

CLSR outperforms BPRMF, probably because BPRMF only captures users' long-term interests and does not consider the relationship between users' short-term preferences and items. In this paper, the model captures users' short-term interest preferences using the self-attention mechanism, and models users' long-term interest preferences using the BiGRU network, while considering long-term and short-term preferences, so that the model CLSR can achieve better recommendation results.

2) COMPARISON OF CLSR MODEL VARIANTS

In order to verify the influence of BiGRU network on the model in this paper, the effect of GRU and BiGRU networks were firstly compared on the basis of CLSR model, and secondly CLSR and its two variants were experimentally compared on four datasets. Namely, CLSR-SimpleRNN (replacing the BiGRU network with a simple recurrent neural network for the CLSR model) and CLSR-BiLSTM (replacing the BiGRU network with a bidirectional long and short-term memory neural network), the model depths were kept consistent. The comparison results are shown in Table 5 and 6.
The length of the recommendation list directly affects the results of the model. When the sequence length is equal to 5, but the cumulative gain is the largest when the length of the sequence keeps increasing, and the hit rate of the model as the cumulative gain gets larger. This shows that the recommendation effect of the CLSR model is higher than that of the two variants in all four datasets, which indicates that the BiGRU network, compared with SimpleRNN and BiLSTM, can effectively preserve the information in long-term sequences in the model proposed in this paper and better capture the users’ long-term interest preferences, which further improves the recommendation performance of the model.

From Tables 5 and 6, we can see that the BiGRU network is more effective than one-way GRU in all four datasets, and the recommendation effect of the CLSR model is higher than that of the two variants in all four datasets, which indicates that the BiGRU network, compared with SimpleRNN and BiLSTM, can effectively preserve the information in long-term sequences in the model proposed in this paper and better capture the users’ long-term interest preferences, which further improves the recommendation performance of the model.

### Table 6

**Experiments with Variants of the CLSR Model on Four Different Datasets**

| Dataset          | Method       | HR@10 | NDCG@10 | MRR@10 |
|------------------|--------------|-------|---------|--------|
| MovieLens-100K   | CLSR-SimpleRNN | 0.5609 | 0.4279  | 0.2317 |
|                  | CLSR-BiLSTM  | 0.5641 | 0.4332  | 0.2594 |
|                  | CLSR         | **0.5864** | **0.4457** | **0.2461** |
| MovieLens-1M     | CLSR-SimpleRNN | 0.7392 | 0.7061  | 0.4145 |
|                  | CLSR-BiLSTM  | 0.7456 | 0.7113  | 0.4250 |
|                  | CLSR         | **0.7514** | **0.7200** | **0.4308** |
| Amazon Product   | CLSR-SimpleRNN | 0.3301 | 0.2638  | 0.1687 |
|                  | CLSR-BiLSTM  | 0.3452 | 0.2724  | 0.1703 |
|                  | CLSR         | **0.3620** | **0.2943** | **0.1788** |
| Yelp             | CLSR-SimpleRNN | 0.4102 | 0.3215  | 0.1789 |
|                  | CLSR-BiLSTM  | 0.4324 | 0.3402  | 0.1874 |
|                  | CLSR         | **0.4463** | **0.3596** | **0.2071** |

### Table 7

**Influence of Different Sequence Lengths L on the Model**

| Settings | MovieLens-100K | MovieLens-1M | Amazon Product | Yelp  |
|----------|----------------|--------------|----------------|-------|
|          | HR@10 | NDCG@10 | HR@10 | NDCG@10 | HR@10 | NDCG@10 | HR@10 | NDCG@10 | HR@10 | NDCG@10 |
| l=1      |      |        |       |        |       |        |       |        |       |        |
| l=2      | 0.5097 | 0.4023 | 0.7103 | 0.6843 | 0.3098 | 0.2386 | 0.3096 | 0.3091 |
| l=3      | 0.5667 | 0.4171 | 0.7265 | 0.6939 | 0.3184 | 0.2598 | 0.4217 | 0.3348 |
| l=4      | 0.5583 | 0.4231 | 0.7416 | 0.7155 | 0.3305 | 0.2745 | 0.4181 | 0.3390 |
| l=5      | 0.5614 | 0.4367 | 0.7478 | 0.7142 | 0.3395 | 0.2706 | 0.4383 | 0.3451 |
| l=6      | **0.5864** | **0.4457** | **0.7514** | **0.7200** | **0.3620** | **0.2943** | **0.4463** | **0.3596** |
| l=7      | 0.5773 | 0.4381 | 0.7433 | 0.7034 | 0.3512 | 0.2808 | 0.4316 | 0.3480 |
| l=8      | 0.5615 | 0.4229 | 0.7458 | 0.7031 | 0.3487 | 0.2701 | 0.4352 | 0.3414 |

3) **Effect of Parameters on the Model**

The model in this paper presents the effect of two parameters on the model on four datasets: sequence length $l$ and embedding dimension $d$. The evaluation is performed using HR@10 and NDCG@10. Firstly, the results of the experiments on the effect of sequence length $l$ on the model are shown in Table 7.

From Table 7, it can be seen that the effectiveness of the recommendation grows year-on-year as the length of the sequence keeps increasing, and the hit rate of the model as well as the cumulative gain is the largest when the length of $l$ is equal to 5, but the recommendation effect gradually decreases when $l$ is larger than 5. One possible reason is that when predicting future items, if the length of $l$ is too long, it may bring a lot of noise to the model and cannot get a better expression, and the $l$ is too small in length, it cannot effectively express the short-term interest preferences of users.

Different item embedding dimensions $d$ and different sequence lengths $l$ are compared simultaneously on the four datasets, firstly, the experimental results under MovieLens-100K and MovieLens-1M datasets are shown in Figs. (3) to (6).
Secondly the model CLSR is experimentally compared for different item embedding dimensions \( d \) and sequence length \( l \) on Amazon Product and Yelp datasets as well, and the results are shown in Figs. (7) to (10).
From Figs. (3) to (10), it is clear that the recommendation performance is not good enough when the embedding dimension of the model is too small or too large, because the embedding dimension is too small to model the potential features of the items at a deep level, and too large to cause the model overfitting problem. For the model in this paper, when the sequence length is 5 and the embedding dimension is 100, the model’s recommendation performance reaches the maximum.

V. CONCLUSION
In this paper, a deep recommendation model (CLSR) fusing long-term and short-term interest preferences is proposed. By modeling users’ recent behaviors, the self-attention mechanism is used to extract short-term behavior sequence features, and mining potential semantic information in interaction sequences, while taking into account users’ long-term preferences, and the BiGRU network is adopted for deep learning, the combination of both can better solve the sequence recommendation problem. Experiments on four public datasets show that CLSR outperforms other methods and verifies its effectiveness. The data of the recommendation system includes not only the interaction between users and items, but also much additional auxiliary information not considered, such as item tags, social information, etc. In the future, it is planned to integrate this information into the model to further improve the recommendation effectiveness of the model.

ACKNOWLEDGMENT
The authors would like to thank the editors and anonymous reviewers for their constructive comments and suggestions, which greatly helped improve the overall quality of the manuscript.

REFERENCES
[1] J. Gantz and D. Reinsel, “The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east,” IDC iView, vol. 2007, no. 2012, pp. 1-16, December. 2012.
[2] B. Alhijawi and Y. Kilani, “The recommender system: A survey,” International Journal of Advanced Intelligence Paradigms, vol. 15, no. 3, pp. 1-1, March. 2020.
[3] A. Karatzoglou and B. Hidasi, “Deep learning for recommender systems,” in Proc. Eleventh ACM Conference on Recommender Systems, Como, Italy, ACM, August. 2017, pp. 396-397.
[4] X. He, L. Liao, H. Zhang, et al., “Neural collaborative filtering,” in Proc. 26th International Conference on World Wide Web, Perth, Australia, April. 2017, pp. 173-182.
[5] H. T. Cheng, L. Koc, J. Harmsen, et al., “Wide & Deep learning for recommender systems,” in Proc. 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, September. 2016, pp. 7-10.
[6] J. Xiao, H. Ye, X. G. He, et al., “Attentional factorization machines: Learning the weight of feature interactions via attention networks,” in Proc. Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), Melbourne, Australia, Aug. 2017, pp. 3119-3125.
[7] J. P. Zhou, Z. Cheng, F. Perez and M. Volkovs, “TAFA: Two-headed attention fused autoencoder for context-Aware recommendations,” in Proc. 14th ACM Conference on Recommender Systems, Virtual Event, Brazil, September. 2020, pp. 338-347.
[8] M. Chen and X. Zhou, “DeepRank: Learning to rank with neural networks for recommendation,” Knowledge-Based Systems, vol. 209, pp. 106478, 2020.
[9] S. Rendle, C. Freudenthaler and L. Schmidt-Thieme, “Factorizing personalized markov chains for next-basket recommendation,” in Proc. 19th International Conference on World Wide Web, Raleigh, North Carolina, USA, April. 2010, pp. 811-820.
[10] E. Smirnova and F. Vasile, “Contextual sequence modeling for recommendation with recurrent neural networks,” in Proc. 2nd Workshop on Deep Learning for Recommender Systems, Como, Italy, August. 2017, pp. 2-9.
[11] W. C. Kang and J. McAuley, “Self-attentive sequential recommendation,” in Proc. IEEE International Conference on Data Mining(ICDM), Singapore, December. 2018, pp. 197-206.
[12] J. Tang and K. Wang, “Personalized Top-N sequential recommendation via convolutional sequence embedding,” in Proc. Eleventh ACM International Conference on Web Search and Data Mining, Marina Del Rey, CA, USA, February. 2018, pp. 565-573.
[13] S. Zhang, Y. Tay and L. Yao, A. Sun and J. An, “Next item recommendation with self-attentive metric learning,” in Proc. Thirty-Third AAAI Conference on Artificial Intelligence, vol. 9, November. 2019, pp. 9-19.
[14] J. Lin, W. Pan and Z. Ming, “FISSA: fusing item similarity models with self-attention networks for sequential recommendation,” in Proc. Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September. 2020, pp. 130-139.
[15] L. Wu, S. Li, C. J. Hsieh and J. Sharpack, “SSE-PT: Sequential recommendation via personalized transformer,” in Proc. Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September. 2020, pp. 328-337.
[16] J. Li, Y. Wang and J. McAuley, “Time interval aware self-attention for sequential recommendation,” in Proc. 13th International
[17] J. Wu, R. Cai and H. Wang, “Déjà vu: A contextualized temporal attention mechanism for sequential recommendation,” in Proc. Web Conference 2020, Taipei, Taiwan, April. 2020, pp. 2199-2209.

[18] S. M. Cho, E. Park, and S. Yoo, “MEANTIME: Mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation,” in Proc. Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September. 2020, pp. 515-520.

[19] Y. K. Cen, J. W. Zhang, X. Zou, C. Zhou, H. X. Yang and J. Tang, “Controllable multi-interest framework for recommendation,” in Proc. The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event CA USA, Jun. 2020, pp. 2942-2951.

[20] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, Oct. 2014, pp. 1724-1734.

[21] C. Y. Wu, A. Ahmed, A. Beutel, A. J. Smola and H. Jing, “Recurrent recommender networks,” in Proc. of the Tenth ACM International Conference on Web Search and Data Mining, New York: ACM press, Feb. 2017, pp. 495-503.

[22] W. D. Zheng and G. Chen, “An Accurate GRU-Based Power Time-Series Prediction Approach With Selective State Updating and Stochastic Optimization,” in IEEE Transactions on Cybernetics, Nov. 2021, pp. 1-13.

[23] Z. Yang, D. Yang, C. Dyer, X. He and E. Hovy, “Hierarchical attention networks for document classification,” in Proc. Hierarchical Attention Networks for Document Classification, San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 1480-1489.

[24] S. Yakhchali, A. Beheshti, S. M. Ghafari, M. A. Orgun, and G. Liu, “Towards a deep attention-based sequential recommender system,” IEEE Access, vol. 8, pp. 178073-178084, 2020.

[25] M. Chen, Y. Li, and X. Zhou, “CoNet: Co-occurrence neural networks for recommendation,” Future Generation Computer Systems, vol. 124, pp. 308-314, 2021.

[26] L. H. Xia, C. Huang, Y. Xu, et al., “Knowledge-Enhanced Hierarchical Graph Transformer Network for Multi-Behavior Recommendation,” in Proc. of the AAAI Conference on Artificial Intelligence, 2021, vol. 35, no. 5, pp. 4486-4493.

[27] R. Cai, J. Wu, A. San, C. Wang, and H. Wang, “Category-aware collaborative sequential recommendation,” in Proc. of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 388-397.

[28] Z. Jiang, C. Chi, and Y. Zhan, “Let knowledge make recommendations for you,” IEEE Access, vol. 9, pp. 118194-118204, 2021.

[29] A. Vaswani, N. Shazeer, N. Parmar, et al., “Attention is all you need,” in Proc. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, Sep. 2017, pp. 5998-6008.

[30] F. M. Harper and J. A. Konstan, “The movie lens datasets: History and context,” in Proc. ACM Transactions on Interactive Intelligent Systems, Jan. 2015, vol. 5 no. 4, pp. 1-19.

[31] S. Rendle, C. Freudenthaler, Z. Gantner and L. Schmidt-Thieme, “BPR: Bayesian personalized ranking from Implicit feedback,” in Proc. of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI2009), Los Angeles: AUAI Press, May 2009, pp. 452-461.

[32] B. Hidasi, A. Karatzoglou, L. Baltrunas and D. Tikk, “Session-based recommendations with recurrent neural networks,” Computer, Aug. 2015, pp. 1511-6939.

[33] R. He, W. C. Kang and J. J. McAuley, “Translation-based recommendation: A scalable method for modeling sequential behavior,” in Proc. of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18), UC San Diego: International Joint Conferences on Artificial Intelligence, Jul. 2018, pp. 5264-5268.
LUSHUAI NIU received the B.E. degrees in computer science and technology from the Zhengzhou Shengda Economic and Trade Management College, Zhengzhou, China, in 2019. He is currently pursuing the M.S. degree with the Sichuan University of Science and Engineering, Yibin, China. In 2019. His research interests include machine learning, recommendation algorithms, and big data processing.

YAN PENG (Member, IEEE) received the B.S. degree in applied mathematics from the Sichuan University, Chengdu, China, in 1991, and the M.S. and PH.D. degrees in metallurgical machinery and mechanical manufacturing and automation, Chongqi, China, in 1997 and 2001, respectively. He is currently the director of the Engineering Practice Center of Sichuan University of Science and Engineering and the director of the Key Laboratory of Enterprise Informatization and Internet of Things Measurement and Control in Sichuan Province Universities. His research interests include Internet of Things technology and applications, machine learning and natural language processing, expert systems, and intelligent information processing.

YIMAO LIU received the B.E. degrees in measurement and control technology and Instrumentation from the Chengdu Technological University, Chengdu, China, in 2018. He is currently pursuing the M.S. degree with the Sichuan University of Science and Engineering, Yibin, China. In 2019. His research interests include machine learning, recommendation algorithms.