Received November 15, 2020, accepted November 28, 2020, date of publication December 1, 2020,
date of current version December 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041772

WGIN: A Session-Based Recommendation Model Considering the Repeated Link Effect

ZHENYU YANG[1,2], HAO WANG[2], AND MINGGE ZHANG[2]
[1]School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221116, China
[2]School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China

Corresponding author: Zhenyu Yang (yang_zhenyu@163.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB1404700.

ABSTRACT Session-based recommendation systems have high application value. Determining how to make better use of anonymous user sessions to recommend items of interest is a considerable challenge for current recommendation systems. Existing research has mainly focused on sequential session patterns; however, due to the complexity and diversity of user interests, such interests cannot be effectively modeled in this way. Therefore, in this paper, we investigate the transition patterns between items by constructing a session graph and propose a novel model called Weighted Graph Interest Networks (WGIN) that collaboratively considers hidden user preference information and the potential order of items in the session graph for a session-based recommendation system. Specifically, we propose a repetitive weighted graph neural network (RWGNN), which pays attention to the transitions between frequent items in a session to deeply explore the preferences of users. In addition, we establish a new Transformer structure to model long-term and short-term user preferences and obtain rich session embeddings. Extensive experiments on two real datasets illustrate that the proposed model outperforms other state-of-the-art session-based recommendation methods.

INDEX TERMS Graph neural network, user preferences, transformer, attention mechanism, session-based recommendation.

I. INTRODUCTION

As one of the main tools for overcoming information overload in the information age, recommendation systems [1] have come to be widely used. This is because they can help users discover and mine valuable information without needing to explicitly specify their intentions [2]. At present, the degree of personalization of recommendation systems is becoming an increasing focus of research. Existing recommendation systems rely mainly on the explicit historical interactions of users to personalize recommendations. Although some success has been achieved, in many scenarios, the user identity is unknown, meaning that explicit interactions alone are insufficient to provide accurate user information. Therefore, it is imperative to research recommendation systems that can effectively explore users' implicit preferences.

The session-based recommendation has high application value, so the increasing research interest on this problem can be observed. In recently, researchers have developed a variety of session-based recommendation methods. The methods based on Markov chains are a classic method [3], [4]. It predicts the next action based on the user's previous behavior. However, this method is easily affected by noise data and cannot model the user's long-term interest. In recent years, the methods based on deep learning have made some progress. Researchers have proposed that Recurrent Neural Network(RNN) [5]–[8] is used to model session sequences and get good results. For example, [5] proposed GRU4Rec, which applies the Gated Recurrent Unit (GRU) to model user preferences. [9] proposed NARM, which employs two RNN modules to model user behavior sequences to capture user preferences. However, these methods only model the single-way transition between consecutive items and do not consider the complex transformation between user behaviors. Recently, some researchers have utilized graph neural networks [10]–[12] to solve the problem of anonymous session recommendation. SR-GNN [13] models the session as a graph structure and only applies a single-layer gated graph neural network to extract the intricate transition pattern between items. In general, although the above methods are effective, they do not fully explore the influence of user interest on the recommendation effect.

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Wang.

By analyzing user sessions, we have found that there are many recurring links in such sessions that reflect user preferences but have not been given sufficient attention in previous research. To better model the complex interactions between users and items in a spatial manner, we choose to use a graph structure [14] to model the data because such a structure can effectively represent the link relationships of interest [15]–[17]. Accordingly, we propose a graph neural network (GNN) framework, RWGNN, that pays more attention to the weights of repeated link edges for the spatial modeling of repeated connections. In detail, we increase the weights of recurring edges to make them more reflective of the user's real consumption habits. In addition, we believe that single-layer information propagation in a graph structure limits the receptive fields of the nodes and does not allow the complete expression and transmission of node information. Therefore, we aggregate the information expressed in different layers to form the final node representation.

In a real recommendation scenario, a complete sequence of sessions may contain information on a variety of different user interests [18]. For example, a user's history may simultaneously contain information about purchases of clothes, cosmetics, and electronic devices. It is easy to imagine that the items reflected in these records may not be equally important to the user. Due to the diversity of user interests, only part of the historical data, rather than the entirety of the historical record, will be relevant to whether the user will click on the currently recommended item. Considering the different levels of importance of different items, we propose the use of the multihead attention mechanism in the Transformer [19], [20] model to adaptively extract users' interests in different spaces to assign different weights to different item representations in a session. This approach helps to obtain a more informative session embedding vector.

Based on the above motivations, we propose a novel model called Weighted Graph Interest Networks (WGIN), in which we first model a session as a directed graph, called the 'session graph'. Then, a repetitive weighted graph neural network (RWGNN) is used to process the session graph and learn the node vector of the items in the session to better reflect the structural information of the session. Based on the different values of different items to the user, we use a pair of multi-perspective attention mechanisms called long multi-interest attention (LMIA) and short single-interest attention (SSIA). Specifically, LMIA allows us to assign different weights to different items in the same session, giving higher weights to items that the user more strongly prefers. SSIA, on the other hand, considers the user's last-clicked item, as the user's most recent behavior tends to be of higher importance than behavior earlier in the session because it represents the user's short-term preferences. Since using these attention mechanisms separately and in isolation would pose difficulties for parallel training of the model, we instead incorporate both multidimensional attention mechanisms into Transformer simultaneously. This approach allows each computation to be performed independently, improving the computational efficiency within the session to obtain higher-order, more granular representations of user preferences. However, although an attention mechanism allows different weights to be adaptively assigned to different items, it lacks position information in its calculations and thus ignores the order in which items appear within a session. Therefore, we first encode a node position vector before applying the Transformer model to obtain a complete session vector representation. Finally, for each session, we predict the probability of each item being clicked next time. In summary, the proposed LSTransformer model captures not only the position order of items within a session but also the spatial structural information extracted by the RWGNN and the weights of additional repeated connections.

In brief, our contributions are as described below:

- A new Transformer structure called LSTransformer, which incorporates two attention mechanisms, is proposed. We use a pair of multidimensional attention mechanisms to efficiently extract users' long-term and short-term preferences in parallel.
- The new RWGNN structure is proposed. This GNN pays more attention to repeated connections within a session on the basis of modeling node information and assists the recommendation model in obtaining more accurate preference information that is more in line with user habits.
- Based on the two modules above, an end-to-end heterogeneous session-based recommendation model is proposed. This model considers both location information within a session and repeated connections in the session graph to extract richer user preferences. A large number of experiments on two real datasets show that our proposed method performs significantly better than existing methods.

II. RELATED WORK

A. SEQUENCE-BASED RECOMMENDATION METHODS

Concerning recommender systems based on sequence data, [3] proposed an MC-based method. Markov models are statistical models with outstanding performance in the fields of natural language processing and biological gene sequence analysis. The session-based recommendation problem can be transformed into an ordered sequence prediction problem; thus, MCs can be used to capture sequential patterns in users' click sessions. In an MC model, a state transition diagram is used to model user behavior in order to predict future behavior. Due to data sparsity, however, such MC models cannot be used directly. Reference [21] utilized a hidden Markov model to overcome the shortcomings of MC models, and [4] employed a decomposable personalized Markov chain (FPMC) method combined with a first-order MC model and matrix factorization to simulate personalized sequential behavior, achieving good results.

In previous research, RNN models have also been successfully applied to the sequence data modeling problem.

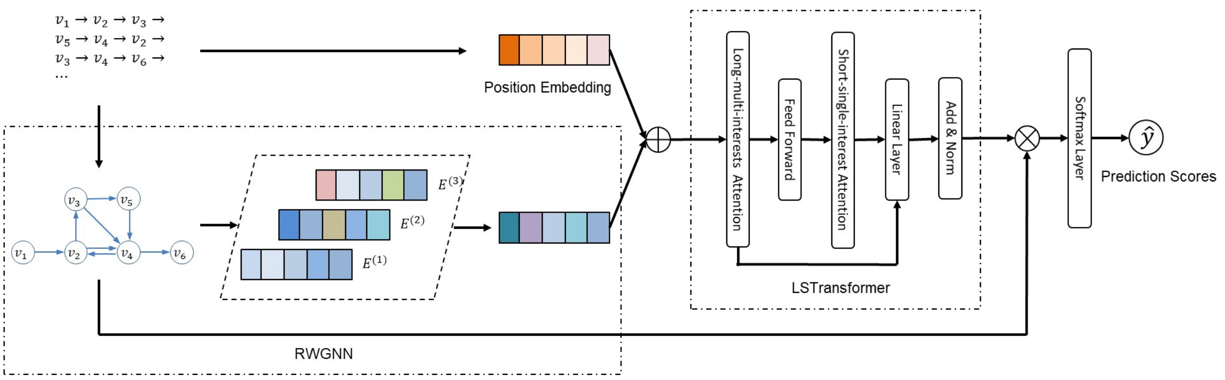

$v_1 \to v_2 \to v_3 \to$
$v_5 \to v_4 \to v_2 \to$
$v_3 \to v_4 \to v_6 \to$
Position Embedding
$E^{(3)}$
$E^{(2)}$
$E^{(1)}$
RWGNN
Long-multi-interests Attention
Feed Forward
Short-single-interest Attention
Linear Layer
Add & Norm
LSTransformer
Softmax Layer
$\hat{y}$
Prediction Scores

FIGURE 1. The structure of the proposed method. As shown in this figure, the entire model consists of two main modules, namely, an RWGNN module and an LSTransformer module. The RWGNN module is used to learn item representations in the session graph, and the LSTransformer module is used to extract long-term and short-term user preferences.

An RNN is a distributed hidden state model with nonlinear dynamic characteristics that can effectively simulate an entire interactive user session. Reference [5] applied an RNN for session recommendation for the first time, designing an RNN model (GRU4REC) with gated recurrent units (GRUs). Compared with the traditional method, the recommendation effect was significantly improved. Reference [7] further used a session-based K-nearest neighbors (KNN) algorithm to sample suitable neighbors, thereby improving the effectiveness of recommendations. Reference [6] introduced four optimization methods based on GRU4REC, namely, data augmentation, model pretraining, the use of privileged information, and output embedding. Reference [8] optimized an RNN by means of a ranking loss function. In NARM [9], a GRU module is utilized as an encoder to extract information, and an attention mechanism is then applied to capture the characteristics of user behavior sequences. CSRM [22] use a combination of sequential and attention models. STAMP [23] employs a multilayer perceptron (MLP) network and an attention mechanism to capture users' long-term and current interests.

B. GRAPH-BASED RECOMMENDATION METHODS

Since user preferences are characterized by complexity, diversity, and real-time variation, modeling user sessions only in the form of sequences is not sufficient to adequately reflect user preferences. Therefore, in recent years, researchers have analyzed user behavior sequences and modeled sessions as graphs to discover potential user preference information [16], [17]. SR-GNN [13] represents the first application of a GNN for the session-based recommendation task. This model uses a gated GNN for session modeling; it also uses an attention mechanism to obtain rich session information. AUTOMATE [24] integrates a graph convolutional layer in a time-featured autoregressive moving average (ARMA) filter into a GNN to process session sequences. FGNN [10] applies multiple graph attention networks to compute the information flows between items within a session. KGCN [25] uses a graph convolutional network to automatically mine the higher-order structural and semantic information of corresponding items in a knowledge graph and to capture potential remote interests of the user. NGCF [26] extracts rich lateral information from user-item interaction graphs via GNNs to better learn embedded representations of users and items.

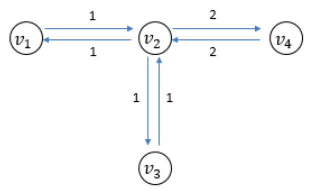
III. PROPOSED RWGNN STRUCTURE WITH INCREASED FOCUS ON REPEATED CONNECTING EDGES

GNNs are well suited for the task of session-based recommendation because they are able to learn and represent complex transition relationships between items during the process of extracting the structural features of session graphs. In this section, we propose a GNN framework called RWGNN, in which more attention is paid to the weights of repeated link edges for the spatial modeling of repeated connections.

In Section III-A, we describe how repeated links in a session graph are handled. Section III-B describes the process of updating the node information in a graph, and Section III-C shows how to perform aggregation operations on node information.

A. REPRESENTATION OF REPEATED CONNECTING EDGES

We have found that in the explicit session data generated by a user, there tend to be many recurring connections. These links can implicitly reflect user preferences and enhance the transfer of item information between sessions. The first consideration is that the weights of such recurring edges should be increased to make them more reflective of the user's true consumption habits. In detail, for each session sequence, we can obtain its corresponding adjacency matrices from the session graph. To represent the directions of information propagation between nodes, we construct an incoming adjacency matrix M_i and an outgoing adjacency matrix M_o. Because frequent subsequences in a user's session sequence often imply information about the user's preferences, we extract the frequent subsequences in each group of session sequences and sum the weights of the corresponding positions in the adjacency matrix to play the role of an attention mechanism. For example, the subsequence $[v_2, v_4]$ frequently appears in the session $S = [v_1, v_2, v_4, v_2, v_4, v_2, v_3, v_2, v_1]$. We extract this subsequence and count its number of occurrences to serve as the corresponding weight in the adjacency matrix. Through this method, we can better learn information on the interactions between nodes to enable better information transmission. The corresponding directed graph and adjacency matrices are shown in Figure 2.



	1	2	3	4	1	2	3	4
1	0	1	0	0	0	1	0	0
2	1	0	1	2	1	0	0	2
3	0	0	0	0	0	1	0	0
4	0	2	0	0	0	2	0	0

in-degree matrix out-degree matrix

FIGURE 2. **An example of a session graph structure and the corresponding adjacency matrices.**

Here, to better learn the information of neighboring nodes, we normalize the weights in each adjacency matrix by row to update the in-degree and out-degree matrices, calculated as follows:

$$M_i[i, j] = \frac{w_{i,j}}{\sum_{j=0}^{n} w_{i,j}} \tag{1}$$

$$M_o[i, j] = \frac{w_{i,j}}{\sum_{j=0}^{n} w_{i,j}} \tag{2}$$

The numerator $w_{i,j}$ represents the weight corresponding to the i-th row and the j-th column, and the denominator is the sum of the weights in the corresponding row.

Subsequently, we map each item $v \in V$ to a unified low-dimensional hidden space; the node vector $e \in R^d$ is the embedding vector of item v, where d is the dimensionality. Next, for the nodes in the session graph, we use the adjacency matrices M_i and M_o to learn the structural information of the graph, and the information propagation between different nodes can be formalized as follows:

$$o^{(l)} = \text{Concat}\left(M_i E^{(l-1)} W_i, M_o E^{(l-1)} W_o\right) + b \tag{3}$$

where $W_i, W_o \in R^{d\times d}$ are the parameter matrices, $e \in R^d$ is the bias vector, $E^{(l-1)} = \left[e_1^{(l-1)}, e_2^{(l-1)}, \cdots, e_n^{(l-1)}\right]$ is the vector of all nodes in the session graph, and $M_i, M_o \in R^{d\times d}$ represent the in-degree and out-degree matrices, respectively. We propagate information from both the in-edges and the out-edges, and $o^{(l)}$ denotes a node e that gathers information from its neighboring nodes at level l.

B. UPDATING OF NODE INFORMATION

We update the current node information by aggregating the information of neighboring nodes. In detail, we use a modified GRU-based RNN to achieve this. By means of the 'updating and memory' functions of GRUs, we are able to retain as much valid neighboring node information as possible. This approach is fundamentally different from average pooling or maximum pooling. The update process can be represented by the following equations:

$$z_i^l = \sigma\left(W_z o_i^{(l)} + U_z e_i^{(l-1)}\right) \tag{4}$$

$$r_i^l = \sigma\left(W_r o_i^{(l)} + U_r e_i^{(l-1)}\right) \tag{5}$$

$$\tilde{e}_i^{(l)} = \tanh\left(W_e o_i^{(l)} + U_e\left(r_i^l \odot e_i^{(l-1)}\right)\right) \tag{6}$$

$$e_i^{(l)} = \left(1 - z_i^l\right) \odot e_i^{(l-1)} + z_i^l \odot \tilde{e}_i^{(l)} \tag{7}$$

where $W_z, W_r, W_e \in R^{2\,d\times d}$ and $U_z, U_r, U_e \in R^{d\times d}$ are all learnable parameters; $\sigma(\cdot)$ is the sigmoid function; $\odot$ denotes the elementwise multiplication operator; z_i^l and r_i^l represent the update and reset gates, respectively, which decide what information is to be preserved and discarded; and $\tilde{e}_i^{(l)}$ is the candidate state of the node. We construct the candidate state $\tilde{e}_i^{(l)}$ based on the previous state, the current state, and the reset gate as described in Eq. (6). The final state $e_i^{(l)}$ is then calculated as a combination of the previous hidden state and the candidate state under the control of the update gate, as shown in Eq. (7). Once all nodes in the session graph have been updated until convergence, we can obtain the final vector representation of all nodes in the session graph.

C. AGGREGATION OF NODE INFORMATION

After L rounds of propagation, we obtain node vectors that aggregate neighborhood information from different layers, expressed as $\left\{e_i^{(1)}, e_i^{(2)}, \cdots, e_i^{(L)}\right\}$. Since the representations obtained in different layers emphasize messages passed over different connections, they make different contributions to the node representation. Therefore, we perform aggregation operations on nodes from different layers to obtain the final node representation. Here, we use the node aggregation method used in LightGCN [27], for which the calculation is as follows:

$$e_i = \sum_{l=0}^{L} a_l e_i^{(l)} \tag{8}$$

where $a_l \geq 0$ denotes the importance of the l-th layer embedding in constituting the final embedding. It can be treated as a hyperparameter to be tuned manually or as a model parameter to be optimized automatically.

IV. WGIN: AN END-TO-END HETEROGENEOUS SESSION RECOMMENDATION MODEL

A. PROBLEM DESCRIPTION

In a session-based recommendation system, $V = \{v_1, v_2, \cdots, v_m\}$ represents the set of item IDs clicked by users in all sessions. We arrange the items clicked by a user in chronological order to form a session $S = [v_{s1}, v_{s2}, \cdots, v_{sn}]$, where $v_{si} \in V$ represents the i-th item clicked by the user in session S. The goal of the session-based recommendation system is to predict the user's next click v_{sn+1}. Under the session-based recommendation model, for session S, we output a probability vector y for all possible items, where the value of each element of the vector y is the recommendation score for the corresponding item. The top N items with the highest scores in the vector y are considered as candidates for recommendation. We provide an overview of the model to visualize its structure in Figure 1.

B. MODEL DETAILS

In this section, we will introduce each important part of the model in detail.

1) CONSTRUCTION OF SESSION GRAPHS

First, we use each session sequence to construct a weighted directed graph $G_s = (V_h, W_e, V_r)$, where the head nodes $V_h = \{v_1, v_2, \cdots, v_n\}$ and the tail nodes $V_r = \{v_1, v_2, \cdots, v_n\}$ correspond to the set of items appearing in the session sequence. Adjacent items $\langle V_h, V_r \rangle$ in the sequence represent the user's clicking order, i.e., the user clicked item V_r after item V_h. W_e represents the weights of the edges, where each edge weight is defined as the number of times the user's clicks formed the corresponding subsequence $\langle V_h, V_r \rangle$ during the session. The more occurrences of an edge there are, the greater the correlation between the two corresponding nodes.

2) REPETITIVE WEIGHTED GRAPH NEURAL NETWORK (RWGNN)

The RWGNN module presented in Section III is an important component of our end-to-end model. The model learns the node vectors of the items appearing in the session via the RWGNN. The GNN assigns higher weights to repeated connections within a session, thus better reflecting the structural information of the session.

3) POSITION EMBEDDING

Through the RWGNN, we finally obtain an embedding vector $E = [e_1, e_2, \cdots e_n]$ of all items appearing in the session.

The different positions of the item within a session have different effects on the prediction. For example, the position of v_5 in the session $\{v_1 \rightarrow v_2 \rightarrow v_5 \rightarrow ?\}$ is close to the position of the item to be predicted, which will have a greater impact on the prediction, but in the session $\{v_4 \rightarrow v_5 \rightarrow v_2 \rightarrow v_3 \rightarrow v_2 \rightarrow ?\}$, the impact of v_5 is relatively small.

Therefore, in order to strengthen the influence of the interaction sequence within the session, we map the position information to a unified low-dimensional hidden space to obtain the position embedding matrix $P = [p_1, p_2, \cdots, p_k] \in R^{k \times d}$ where p_i represents the i-th position embedding vector, k represents the number of positions. Specifically, we set $k = 6$, and for the items after the sixth item, their positions are all 6. Here, we code from back to front, the item position of the last interaction is 1, the second to last is 2, and so on. Finally, the item embedding vector with location information is computed as follows:

$$e_j^p = e_j + p_j \tag{9}$$

where $p_j \in R^d$ is the position embedding vector at position j. Thus far, we have obtained the initial session embedding representation $S = \left[e_1^{(p)}, e_2^{(p)}, \cdots, e_n^{(p)}\right]$.

4) LSTransformer MODULE

For the task of session-based recommendation, we propose a pair of multidimensional attention mechanisms, namely, LMIA and SSIA. LMIA is used to extract a user's long-term preferences. It can extract different user preferences from complete session sequence information to obtain a richer session embedding representation. SSIA focuses on the user's recent interests and considers the impact of the user's last click on the final prediction.

a: LONG MULTI-INTEREST ATTENTION (LMIA)

Here, we propose further processing of the session embedding vector $S = \left[e_1^{(p)}, e_2^{(p)}, \cdots, e_n^{(p)}\right]$ by means of the LMIA mechanism [28]. The user's interests are adaptively extracted in different spaces through the multihead attention mechanism of the Transformer model to assign different weights to different item representations in the session.

In our scenario, we feed the learned sequence of embedding vectors into the attention layer as input and cause the nodes to learn different weights that reflect the user's liking for different items.

Self-attention [29] is defined as follows:

$$\text{Attention}\,(Q, K, V) = \text{softmax}\left(\frac{QK}{\sqrt{d}}\right) V \tag{10}$$

where Q represents a query, K represents a key, V represents a value, and d is the dimensionality of the key. The attention score is a weighted sum of values, where the weight assigned to each value is calculated by means of a similarity function between each query and the corresponding key. Due to the complexity of user preferences, we use a multihead attention mechanism to capture the correlations between different perspectives in a sequence. The formula is as follows.

$$F = Mh\left(E^p\right) = \text{Concat}\,(\text{head}_1, \text{head}_2, \cdots, \text{head}_h)\, W^H \tag{11}$$

$$\text{head}_i = \text{Attention}\left(SW^Q, S\,W^K, S\,W^V\right) \tag{12}$$

where $W^Q, W^K, W^V \in R^{d\times d}$ are projection matrices, E^p is the embedding vector of all items, and h is the number of heads. The multihead attention module feeds S into a self-attention layer after mapping through the parameter matrix, repeats this process h times, and finally concatenates all of the results together and sends them to a fully connected layer.

Next, we use a feedforward network (FFN) with the Leaky ReLU [30] activation function to further enhance the model by introducing nonlinearity. In addition, to avoid overfitting and enable better learning of features, we apply dropout during training. The final output is shown below:

$$S = \text{LeakyReLU}\left(FW_1 + b_1\right) W_2 + b_2 + F \tag{13}$$

where W_1, W_2, b_1, and b_2 are the learnable parameters.

b: SHORT SINGLE-INTEREST ATTENTION (SSIA)

After LMIA, we obtain the session representation $S_l = [i_1, i_2, \cdots, i_n]$, which represents the user's long-term preferences. However, this representation alone is not sufficient to serve as a global representation of the user session. Since the nodes in the session represent the user's interaction sequence, we believe that the user's latest interaction behavior will be close to the user's current preference; that is, the positive effect of the user's last click on the model cannot be ignored. Thus, we additionally use the SSIA mechanism to extract users' short-term preferences.

Since a user's latest click is close to the user's current preference, we use only the node vector i_n corresponding to the item last clicked by the user in the session sequence to obtain the current interest representation $S_s = i_n$.

Finally, we combine the long-term interest representation for the session with the current interest representation to obtain the final session embedding. The formula is as follows:

$$S_f = W_3 \text{ Concat } (S_l, S_s) \tag{14}$$

where the matrix $W_3 \in R^{2\,d\times d}$ compresses the two combined embedding vectors into the latent space R^d.

C. MODEL PREDICTION AND TRAINING

Since the model will have a popularity bias problem in the actual online configuration, we use standardized representations in the training phase to alleviate this problem [31]. We perform L2-norm regularization on the final session embedding S_f and the embedding vectors of the candidate items:

$$\widetilde{S_f} = \frac{S_f}{\|S_f\|_2} \tag{15}$$

$$\widetilde{e_i} = \frac{e_i}{\|e_i\|_2} \tag{16}$$

Then, each candidate item is scored through the dot product operation:

$$\widehat{g}_i = \gamma \widetilde{S}_f^{\mathrm{T}} \widetilde{e}_i \tag{17}$$

where γ is a hyperparameter used to further widen the gap between high-intention items and low-intention items.

Then, we use the softmax function to process the score of each candidate item:

$$\hat{y} = \text{softmax}(\hat{g}) \tag{18}$$

where $\hat{y}$ denotes the recommendation score of item v, corresponding to the probability that this item will be the next click in session S. For each session, we use the cross-entropy loss function. The formula is given as follows:

$$\mathrm{L} = -\sum_{i=1}^{n} y_i \log\left(\widehat{y_i}\right) + (1 - y_i) \log\left(1 - \widehat{y_i}\right) \tag{19}$$

where y denotes a one-hot vector exclusively activated by $v_i \in V$ (the ground truth). For example, if v_i is the next click in session S, then $y_i = 1$; if not, $y_i = 0$. An iterative stochastic gradient descent (SGD) optimizer is then applied to optimize the cross-entropy loss.

In our proposed method, a weighted gated GNN is used to process the session graph and learn the node vectors of the items appearing in the session. This weighted composition method can better reflect the structural information of the session. In addition, we use the LMIA and SSIA mechanisms in the LSTransformer module to extract long-term and short-term user preferences, respectively, to obtain a session embedding vector with richer information.

V. EXPERIMENTAL DESIGN

In this section, we introduce the datasets used in the experiments, the baseline methods, the evaluation metrics, and the parameter settings.

A. DATASETS

We evaluated our model on two representative real-world datasets, YOOCHOOSE and DIGINETICA.

YOOCHOOSE[1] comes from the RecSys Challenge 2015. It contains click streams from an e-commerce website collected over a period of six months.

DIGINETICA[2] comes from CIKM Cup 2016. It contains transaction data, which are a suitable basis for session-based recommendation.

For data preprocessing, we used the processing method presented in [6]. We filtered out items that appeared fewer than 5 times in a session and deleted sessions with a sequence length of less than 2. Additionally, in chronological order, we selected the first 80% of the generated sessions as the training set and used the remaining sessions as the test set. From the training set, we randomly selected 10% of the interactions as the validation set to adjust the hyperparameters. In addition, we used a data augmentation method to process the datasets. For example, for an input session $s = \left[v_{s1}, v_{s2}, \cdots, v_{sn}\right]$, we generated a series of sequences and corresponding labels

[1]http://2015.recsyschallenge.com/challenge.html
[2]http://cikm2016.cs.iupui.edu/cikm-cup

TABLE 1. Statistic details of datasets.

Datasets	Clicks	Train	Test	Items	Average Length
YOOCHOOSE1/4	8326407	5917746	55898	29618	5.71
YOOCHOOSE1/64	557248	369859	55898	16766	6.16
DIGINETICA	982961	719470	60858	43097	5.12

as follows: $([v_{s1}], v_{s2}), ([v_{s1}, v_{s2}], v_{s3}), \cdots, ([v_{s1}, v_{s2}, \cdots, v_{sn-1}], v_{sn})$, where $[v_{s1}, v_{s2}, \cdots, v_{sn}]$ is a generated sequence and v_{sn} denotes the next-clicked item, i.e., the label of the sequence. The statistics of the two datasets are shown in Table 1.

B. BASELINES

We use the following representative models as a baseline to evaluate the performance of our model.

- **POP**: This method selects popular items as user click predictions for the recommendation.
- **S-POP**: For each session, this method selects the item with the most occurrences as the prediction.
- **Item-KNN** [32]: This method utilizes the cosine similarity to calculate the similarity score between the user's last interactive item and the candidate item in the session, and recommends the top-N items with high similarity scores.
- **BPR-MF** [33]: This method applies the Bayesian method to optimize the ranking of user preferences, and then combines the matrix factorization method to make personalized recommendations.
- **FPMC** [4]: This method employs a first-order Markov chain combined with matrix factorization to predict the session sequence and recommend the user's next click.
- **GRU4REC** [5]: This method uses the RNN model on the task of a session-based recommendation system, and utilizes GRU to extract sequence information.
- **NARM** [9]: This method applies the RNN model to extract sequence information and adds an attention mechanism to capture the user's main purpose for the recommendation.
- **STAMP** [23]: This method captures the user's long-term preferences through sequence information, and captures the current preferences through the user's last click. The two work together to improve the recommendation effect.
- **SR-GNN** [13]: This method applies a gated neural network to capture complex transitions of items for session-based recommendation.
- **GACOforRec** [34]: This model is based on GCNs, and applies ConvLSTM and ON-LSTM to deal with users' long-term stable preferences and retain their hierarchical structure.
- **AUTOMATE** [24]: This model applies the combination of an ARMA filter with time-series features and a graph convolutional neural network for session recommendation.

C. EVALUATION METRICS

To evaluate the performance of the model, we adopted Recall@20 and MRR@20 as our evaluation metrics.

Recall@20 (recall calculated over the top 20 items): This is the primary performance metric for a recommendation system, representing the proportion of correctly recommended items among the top 20 items.

MRR@20 (mean reciprocal rank calculated over the top 20 items): This metric is the average of the reciprocal ranks of the correctly recommended items, where the reciprocal rank is set to 0 if the rank is greater than 20. The MRR considers the positions of the correctly recommended items in a ranked list. The higher the MRR@20 score is, the better the recommendation quality.

D. PARAMETER SETUP

In our model, we set the dimensionality of the latent vectors to 128. All parameters were initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. We used the minibatch Adam optimizer to optimize our model. The learning rate was initially set to 0.001 and decayed by 0.1 after every three epochs. In addition, the batch size was set to 100, and the L2 penalty was set to 10^{-5}. The number of heads for the LMIA mechanism was set to 4.

VI. EXPERIMENTAL RESULTS

In this section, we compare the proposed method with the baseline methods. Then, we present a detailed analysis of the impact of the different modules implemented in our model.

A. COMPARISON WITH BASELINE METHODS

We presented the representative existing baseline methods chosen for comparison with our method in Section V. In this section, we use Recall@20 and MRR@20 as the evaluation metrics to compare our method with these baseline methods, and the results are shown in Table 2.

This table reveals several shortcomings. The traditional methods POP and S-POP achieve relatively low scores in terms of both evaluation metrics because they only recommend products that appear frequently in the session, which is far from sufficient for the session-based recommendation task. In addition, S-POP shows better performance than the POP and BPR-MF methods, demonstrating that it is necessary to introduce contextual information when generating session-based recommendations. It is also worth noting that the MC-based method FPMC does not perform as well as Item-KNN, which utilizes only the similarity between items without considering sequence information. This proves that

TABLE 2. The performance of WGIN with other baseline methods over three datasets.

Methods	YOOCHOOSE 1/64		YOOCHOOSE 1/4		DIGINETICA	
	Recall@20	MRR@20	Recall@20	MRR@20	Recall@20	MRR@20
POP	6.71	1.65	1.33	0.3	0.89	0.2
S-POP	30.44	18.35	27.08	17.75	21.06	13.68
Item-KNN	56.6	21.81	52.31	21.7	35.75	11.57
FPMC	45.62	15.01	-	-	26.53	6.95
BPR-MF	31.31	12.08	3.4	1.57	5.24	1.98
GRU4Rec	60.64	22.89	59.53	22.6	29.45	8.33
NARM	68.32	28.63	69.73	29.23	49.7	16.17
STAMP	68.74	29.67	70.44	30	45.64	14.32
GACOforRec	68.79	29.38	67.66	28.13	47.83	14.28
SR-GNN	70.57	30.94	71.36	31.89	50.73	17.59
AUTOMATE	70.15	30.72	69.89	29.16	50.11	16.72
Ours Model	**70.97**	**31.35**	**71.98**	**32.31**	**53.04**	**18.52**

TABLE 3. Performance when K = 5 and 10 for YOOCHOOSE 1/64.

Method	YOOCHOOSE 1/64			
	Recall@5	MRR@5	Recall@10	MRR@10
NARM	44.34	26.21	57.50	27.97
STAMP	45.69	27.26	58.07	28.92
SR-GNN	47.42	28.41	60.21	30.13
WGIN	**48.23**	**29.16**	**60.97**	**30.85**

it is not sufficient to model only sequence information for session-based recommendation.

In addition, it is evident that the performance of methods based on deep learning is significantly better than that of traditional methods. Compared with traditional methods, a neural-network-based method can better capture the hidden information from session data. For example, GRU4REC and NARM use GRU, an improved variant of standard RNNs, to model the session sequences. This solves the problem of information loss in traditional neural networks and further enables the extraction of hidden information from session sequences. The STAMP model uses an MLP to model the user's recent interests, which improves the modeling effect, thus proving the importance of recent user behavior for session-based recommendation. Compared with RNN-based methods, GNN-based methods also achieve better results for the session-based recommendation task. SR-GNN represents the first application of a GNN for session-based recommendation. Specifically, it uses a single-layer gated GNN to model user behavior while considering the complex transitions between items, thus improving the recommendation effect. In addition, GACOforRec uses graph convolutional neural networks to extract spatiotemporal information from a session, and AUTOMATE uses a graph convolutional layer in an ARMA filter to process session sequences. Both methods achieve good results, proving that GNNs are more suitable for session-based recommendation than traditional methods.

Our method, WGIN, not only considers the structural information of the session graph but also retains the original timing information of the session sequence. In our model, we first use a multilayered RWGNN module to learn the node information from the session graph. Compared with a traditional GNN, our network attaches greater importance to more frequent item transitions in the session and, at the same time, aggregates different levels of information to obtain a richer item feature representation. In addition, we consider the influence of sequence information on the recommendation effect. An LSTransformer module is used to extract user preferences from different perspectives, thereby enhancing the expressiveness of the model. The experimental results show that our proposed method obtains the best results on both datasets.

In addition, we have improved upon the standard evaluation metrics by also testing the performance of WGIN in terms of Recall@5, Recall@10, MRR@5 and MRR@10. It can be seen from the experimental results that our model can still maintain excellent performance under higher recommendation standards, thus proving that our model offers high recommendation accuracy and can well reflect user preferences.

B. ABLATION EXPERIMENTS

1) THE IMPACT OF SESSION EMBEDDING

To evaluate the impacts of different embedding methods on the modeling effect, we compared our method with several variants: (1) WGIN-S, which uses only the SSIA mechanism and does not consider the influence of the user's long-term preferences; (2) WGIN-L, which uses only the LMIA mechanism and does not consider the impact of the user's short-term preferences; (3) WGIN-AVG, which uses average pooling to process user session sequences. and (4) WGIN-ATT [13], which uses each item in the sequence and the most recently clicked item to calculate the attention score when processing global preferences. The results are shown in Figure 3.

It can be seen from Figure 3 that our method is better than the method that uses only short-term preferences, indicating that short-term preferences have limitations and cannot fully reflect user preferences. Similarly, the method that uses only long-term preferences is also less effective than the proposed method. This proves that considering short-term preferences has a positive effect on the model. Thus, it is necessary to simultaneously consider a user's long-term preferences and current interests. We also find that the performance of our method is better than that of WGIN-AVG, indicating that each

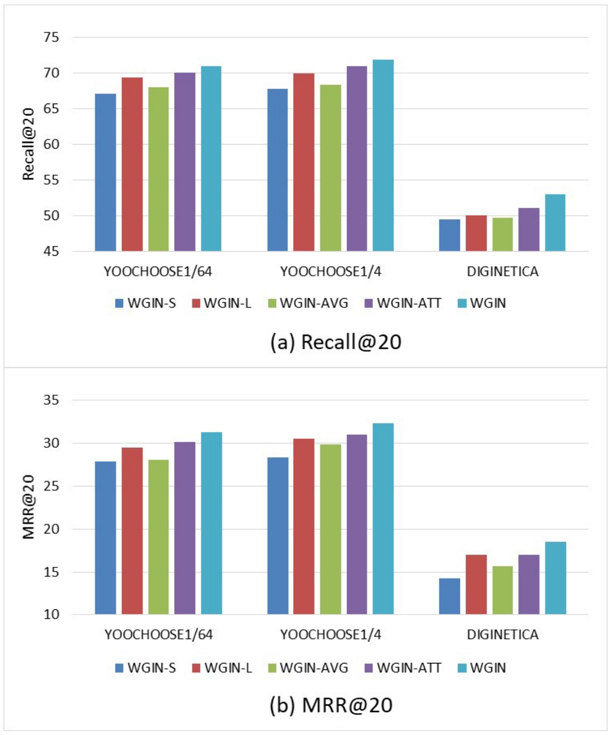


FIGURE 3. The performance of different session representation.

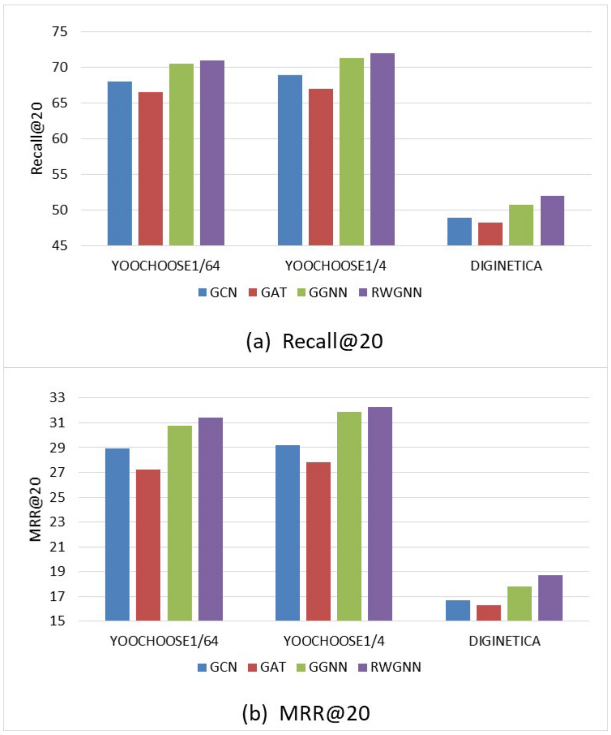


FIGURE 4. The performance of different GNN layers.

item in the session has a different impact on the prediction. In addition, the performance of our method is better than that of WGIN-ATT, which shows that compared with soft attention, multi-head attention can mine the potential user preference information in the sequence and enrich the session embedding.

2) THE IMPACT OF GRAPH NEURAL NETWORK

In addition, we studied the influence of the GNN module. First, we verified how the different information dissemination methods in different GNNs affect the model. For these comparative experiments, we used a graph convolutional network (GCN) [14], a graph attention network (GAT) [35], and a gated graph neural network (GGNN) [36] in place of the RWGNN module in the original model. The results are shown in Figure 4.

From this figure, we can see that the GAT does not perform well on our task. A GAT uses the similarities between the current node and its neighboring nodes as the weights to update the node representation. In our task, the node embedding vectors are randomly initialized to learn the structural information of the graph, and the use of a GAT is likely to cause the graph structure information to be lost in the propagation process; thus, the GAT information propagation method is not feasible for our task. In addition, we can see that the effect of the GCN is not as good as that of the GGNN, thus proving that the gating mechanism plays a vital role in the dissemination of node information. Furthermore, in our method, weight information is added to the edges based on the GGNN approach. The addition of this weight information makes the graph more expressive and allows a richer information representation to be learned.

Then, we investigated the influence of the number of layers of GNN propagation on the modeling effect. We conducted experiments on the DIGINETICA dataset, and the experimental results are shown in Figure 5. As shown in this figure, our model performs best after two layers of information dissemination. After three layers of dissemination, it can be clearly seen that the experimental effect has declined in terms of both the Recall@20 and MRR@20 indicators. These results indicate that as the number of layers increases, the node embedding vector will be oversmoothed, causing some potentially useful information to be lost and degrading the modeling effect.

Since the information contained by neighboring nodes in different layers of a GNN is different, we also evaluated the impact of the aggregation of hierarchical information in the GNN on our model. We recorded the results of each of the first 4 layers of the aggregated neural network on the DIGINETICA dataset. As shown in Figure 6, the model achieves the best effect in terms of Recall@20 when the first two layers of information are aggregated. When the first 3 layers of information are aggregated, the model achieves the best performance in terms of MRR@20. From the above results, we can see that the results obtained by applying the aggregation operation to nodes from different layers are better than those obtained by using only a single-layer network in terms of both evaluation metrics. Because different layers can

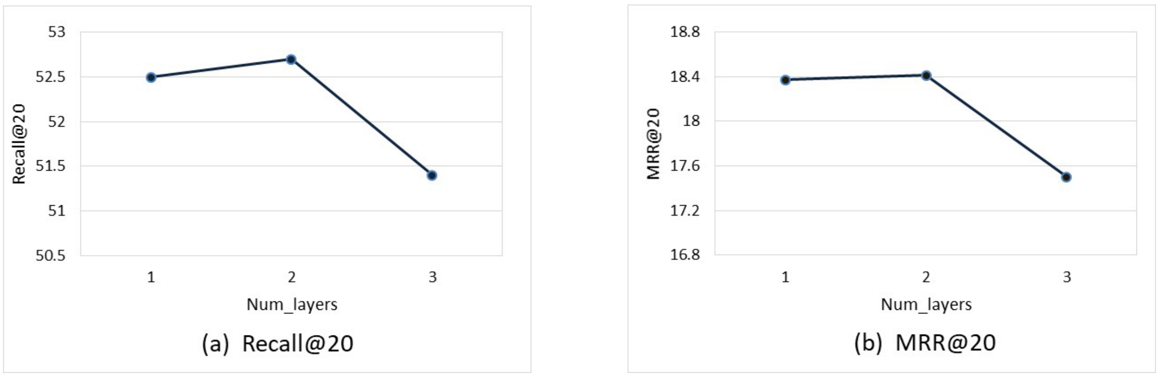


FIGURE 5. The performance of different number of layers.

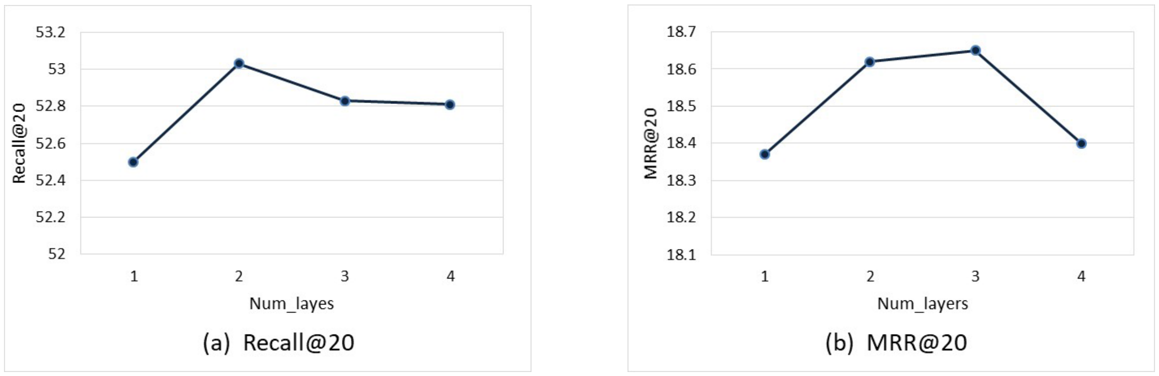


FIGURE 6. The performance of aggregation of hierarchical information.

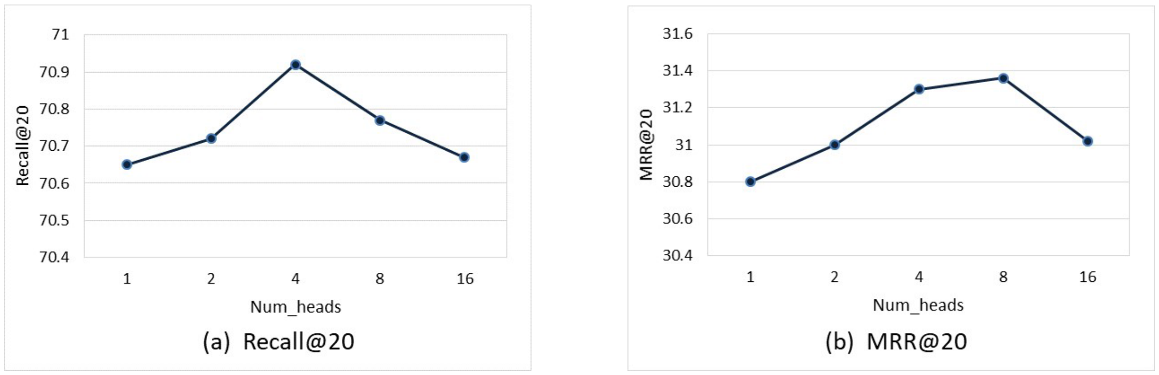


FIGURE 7. The performance of aggregation of hierarchical information.

capture different node information, aggregating the information from multiple layers can effectively supplement missing node information, thereby improving the effectiveness of the model.

3) THE IMPACT OF THE NUMBER OF HEADS IN THE MULTIHEAD ATTENTION MECHANISM

Finally, we tested the influence of the number of heads in the multihead attention mechanism in the LSTransformer module to optimize our model to the greatest possible extent. We used Recall@20 and MRR@20 as our evaluation metrics on the YOOCHOOSE1/64 dataset and recorded the experimental results achieved with 1, 2, 4, 8, and 16 heads, as shown in Figure 7. From this figure, we can see that when the modeling effect is evaluated in terms of Recall@20, the best effect is obtained when the number of heads is set to 4. In terms of MRR@20, the best effect is obtained when the number of heads is set to 8. Due to the influence of the sequence length, a larger number of heads does not necessarily result in a better modeling effect. Instead, we need to set different numbers of heads for different problems.

VII. CONCLUSION

In this paper, we propose a novel architecture called WGIN, a session-based recommendation model that considers the repeated link effect. This model captures user preferences from the perspectives of both graph and sequence representations. The proposed method not only considers the complex structure of the transitions between items appearing in session sequences by means of a repetitive weighted graph neural network but also integrates different levels of information to deeply explore user preferences. In addition, a long multi-interest attention mechanism and a short single-interest attention mechanism are both used in a Transformer-based module to extract long-term and short-term user preferences. Comprehensive experiments on two real datasets illustrate that the proposed method outperforms other state-of-the-art session-based recommendation methods.

In the future, we will further explore the potential correlations within sessions. We will also analyze users' short-term preferences in greater depth and seek better ways to express them. In addition, we hope to introduce rich external information (e.g., knowledge graphs) to more accurately reflect user preferences.

REFERENCES

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[2] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The Adaptive Web*. Berlin, Germany: Springer, 2007, pp. 291–324.

[3] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-based recommender system," *J. Mach. Learn. Res.*, vol. 6, no. 9, pp. 1265–1295, 2005.

[4] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized Markov chains for next-basket recommendation," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 811–820.

[5] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," 2015, *arXiv:1511.06939*. [Online]. Available: http://arxiv.org/abs/1511.06939

[6] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *Proc. 1st Workshop Deep Learn. Recommender Syst. (DLRS)*, 2016, pp. 17–22.

[7] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 306–310.

[8] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-K gains for session-based recommendations," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 843–852.

[9] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1419–1428.

[10] R. Qiu, J. Li, Z. Huang, and H. Yin, "Rethinking the item order in session-based recommendation with graph neural networks," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 579–588.

[11] Q. Lin, Y. Niu, Y. Zhu, H. Lu, K. Z. Mushonga, and Z. Niu, "Heterogeneous knowledge-based attentive neural networks for short-term music recommendations," *IEEE Access*, vol. 6, pp. 58990–59000, 2018.

[12] Y. Zhu, H. Lu, P. Qiu, K. Shi, J. Chambua, and Z. Niu, "Heterogeneous teaching evaluation network based offline course recommendation with graph learning and tensor factorization," *Neurocomputing*, vol. 415, pp. 84–95, Nov. 2020.

[13] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 346–353.

[14] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907

[15] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[16] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for Web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.

[17] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 555–563.

[18] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1059–1068.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[21] M. H. Aghdam, N. Hariri, B. Mobasher, and R. Burke, "Adapting recommendations to contextual changes using hierarchical hidden Markov models," in *Proc. 9th ACM Conf. Recommender Syst. (RecSys)*, 2015, pp. 241–244.

[22] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. de Rijke, "A collaborative session-based recommendation approach with parallel memory modules," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 345–354.

[23] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: Short-term attention/memory priority model for session-based recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1831–1839.

[24] H. Wang, G. Xiao, N. Han, and H. Chen, "Session-based graph convolutional ARMA filter recommendation model," *IEEE Access*, vol. 8, pp. 62053–62064, 2020.

[25] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf. (WWW)*. New York, NY, USA: Association Computing Machinery, 2019, pp. 3307–3313.

[26] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 165–174.

[27] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," 2020, *arXiv:2002.02126*. [Online]. Available: http://arxiv.org/abs/2002.02126

[28] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 197–206.

[29] Z. Lin, M. Feng, C. Nogueira dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," 2017, *arXiv:1703.03130*. [Online]. Available: http://arxiv.org/abs/1703.03130

[30] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou, "Behavior sequence transformer for e-commerce recommendation in Alibaba," in *Proc. 1st Int. Workshop Deep Learn. Pract. High-Dimensional Sparse Data*, Aug. 2019, pp. 1–4.

[31] P. Gupta, D. Garg, P. Malhotra, L. Vig, and G. Shroff, "NISER: Normalized item and session representations to handle popularity bias," 2019, *arXiv:1909.04276*. [Online]. Available: http://arxiv.org/abs/1909.04276

[32] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web (WWW)*, 2001, pp. 285–295.

[33] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*. [Online]. Available: http://arxiv.org/abs/1205.2618

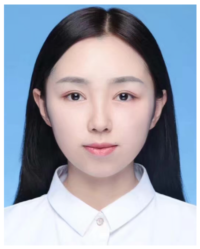
[34] M. Zhang and Z. Yang, "GACOforRec: Session-based graph convolutional neural networks recommendation model," *IEEE Access*, vol. 7, pp. 114077–114085, 2019.

[35] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903

[36] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*. [Online]. Available: http://arxiv.org/abs/1511.05493

ZHENYU YANG received the M.S. degree in computer application technology from the Qilu University of Technology (Shandong Academy of Sciences), in 2007. He is currently pursuing the Ph.D. degree in control engineering and theory with the China University of Mining and Technology. He is also an Associate Professor with the Qilu University of Technology (Shandong Academy of Sciences) and the Deputy Director of the Software Integration Institute. His research interests include artificial intelligence, knowledge management, and information integration.

HAO WANG received the B.E. degree in electronic information engineering from Qufu Normal University, China, in 2018. He is currently pursuing the M.S. degree with the College of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), China. His research interests include deep learning, personalized recommendation, and interpretable recommendation.



MINGGE ZHANG received the B.Sc. degree in administration from the Tianjin University of Science and Technology, in 2018. She is currently pursuing the M.S. degree in software engineering with the Qilu University of Technology (Shandong Academy of Sciences). Her research interests include deep learning, interpretable reasoning, and recommendation systems.