Enhancing Session-Based Recommendations by Fusing Candidate Items

Yingjuan Sun
College of Computer Science and Technology, Changchun Normal University, China
syj_pyf@sohu.com

Wanhua Li
College of Computer Science and Technology, Changchun Normal University China
wauhua@163.com

Jingqi Xing
College of Computer Science and Technology, Changchun Normal University, China
x819617292@163.com

Bangzuo Zhang
School of Computer Science and Information Technology, Northeast Normal University, China
zhangbz@nenu.edu.cn

Dongbing Pu
School of Computer Science and Information Technology, Northeast Normal University China
pudb@nenu.edu.cn

Qian Liu
College of Computer Science and Technology, Changchun Normal University, China
18952236795@163.com

Yinghui Sun
College of Computer
Jilin Normal University, China
sunyh178@163.com

Abstract: *Session-based recommendations are used to convert complex items by using the graph neural network, where this also involves combining session-level and global-level information to discover user preferences. However, this ap-proach en-counters certain problems. A user with extensive interests should be offered more than one recommendation of candidate items. We propose a neural network-based model to fuse candidate items based on this premise. We first use a graph neural network to acquire session-level and global-level information, and then use an attention mechanism to obtain a representation of candidate items recommended to the user. Finally, we integrate the candidate-level, glob-al-level, and session-level information to acquire rich information on the items in the given session. Extensive tests on three empirically ac-quired datasets showed that our model is superior to baseline models in most cases*.

Keywords: *Recommendation systems, session-based recommendation, graph neural network, attention mechanism*.

Received November 20, 2023; accepted September 22, 2024
https://doi.org/10.34028/iajit/21/6/7

1. Introduction

Recommendation systems are essential in various network platforms because they help customers avoid redundant messages by recommending suitable items to them. Traditional recommendation algorithms often rely on personal user information and a long history of their access-related data. Such traditional algorithms as collaborative filtering [17] segment users based on their interests and recommend to them products that have been chosen by users with similar interests, whereas content-based recommendation algorithms [1, 13] generate recommendations to users based on information on the relevant items. However, the traditional method performs poorly when users log in anonymously because no information on the user's configuration and long-term historical information on their inter-actions is available in this case. Session-based recommendation has been developed to solve such problems [2, 4, 8, 15, 21].

Session-based recommendation has recently attracted wide attention as it can be used to forecast the next item of interest in a chronological order based on a series of anonymous activities. At the outset of research on models for session-based recommendation, many scholars employed Markov chains [7, 18] as the core algorithm to model the sessions. Wu *et al*. [26] introduced a Markov chain-based approach to embedding that can be used to transfer items and users into the Euclidean space, with the distance between them representing the probability of their transference. The results of embedding are then used to rank the candidate items. Rendle *et al*. [16] pro-posed a hybrid Factorizing Personalized Markov Chains (FPMC) model that can mix methods of matrix decomposition with the Markov chain to obtain both the general interest and the expected interest of the user, and achieved promising results. However, the Markov chain model assumes that the user’s next action can be predicted based on their preceding actions, and this is easily influenced by noisy data that limit its effectiveness in conversation-based recommendation scenarios.

Many deep neural network models have been proposed for the above problems and have achieved impressive performance. session-based recommendations with Recurrent Neural Networks (RNN) (GRU4Rec) [6] were the first session-based model of recommendation founded on a Recurrent Neural Network (RNN) model. It uses a Gated Recurrent Unit (GRU) layer to form a one-way sequence from session-related data in strict chronological order, given that the user's data are often created during a short period and the relevant information may be temporally related. A Neural Attentive session-based Recommendation Model (NARM), reported by Li *et al*. [10], is based on the RNN, and incorpo-rates an attention mechanism into the GRU encoder to gather information on short-term sessions. Liu *et al*. [11] introduced a Short-Term Attention/Memory Priority model (STAMP), which employs a Multi-Layer Perceptron (MLP) model and an attention mechanism to extract the potential interests of users. However, many deep neural network models [19, 20, 22] simply extract information on users based on a time series to model the session. Transitioning an item is an intricate task, and the user's behavior may not develop in strictly chronological order. Changing the order of the items in a session has no effect on the user's preference for them; rather, modeling the items in a session in strict chronological order may result in overfitting.

Graph Neural Networks (GNN) have been introduced to imitate the intricacy of item relationships in session-related data to better portray the links between them. The first graph neural network developed for session-based recommendations Spatial Relation-aware Graph Neural Network (SR-GNN) was by Wu *et al*. [25]. It models sessions as graphs to capture item associations and then uses an attention mechanism to record user preferences. Following its success, many graph neural network models have emerged, such as the Graph-Contextualized Self-Attention Network (GCSAN) [28], the personalized GNN with an attention mechanism Alarm Propagation Graph Neural Network (A-PGNN) [31], and the Lossless Edge Order-Preserving Aggregation and Shortcut Graph Attention (LESSR) [3]. Qiu *et al*. [14] proposed a Full Graph Neural Network (FGNN) that uses a multi-headed attention mechanism to collect information about an item's neighbors to derive item representations. Fang *et al*. [5] proposed a session-based recommendation with Spatial Relation Self-Attention Networks (SR-SAN) to mine long-term dependencies between items by introducing a self-attention network layer. However, these methods consider only item transitions for the given session and do not consider information on transitions for all sessions. Several models that consider the global fusion of information have emerged in response to this problem. Wang *et al*. [23] introduced a Global Context Enhanced Graph Neural Network (GCE-GNN), which can examine transitions between items from a session as well as the global perspective. Xia *et al*. [27] proposed a Dual Channel Hypergraph Convolutional Network model (DHCN) to capture higher-order information between items through a hypergraph-based neural network. They used self-supervised learning to help improve hypergraph modeling. Pang *et al*. [12] proposed a Heterogeneous Global Graph Neural Networks model (HG-GNN) that constructs a heterogeneous graph to obtain the user's preference-related representation.

Although fusing information from other sessions into the present one can enrich the available content, it may fix the user's attention and degrade performance. Furthermore, adding information on the candidate items can pique the user's interest and improve the representation of information on the session.

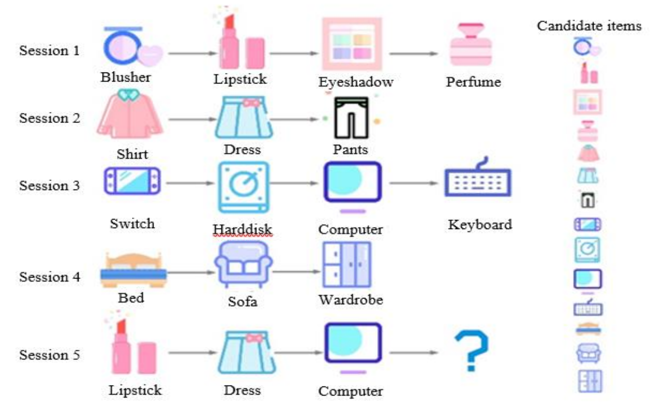


Figure 1. An example of considering information from sessions other than the current one as well as information on the candidate item.

Figure 1 illustrates this point for presentation purposes. Assuming that the current session is "session 5," recommending the next item to the user according to the previous three items is purpose of the session-based recommendation. Figure 1 shows the following:

1) Using information from other sessions facilitates the discovery of user preferences for the current session. For instance, in the other sessions, items related to the item "computer" are present in "session 3," items related to "dress" are present in "session 2," and items related to "lipstick" are present in "session 1." The representational information in "session 5" can be enriched by using information from the other sessions.
2) Using materials relevant to the current session and directly recommending them to the user may lower the user's interest if they have a wide variety of interests and want to see content other than their historical items. The user in "session 5" wants to look at furniture but, as in the GCE-GNN [23], the representation information obtained may contain only a few types of makeup, clothes, and digital items. Subsequent downstream recommendations then become limited to these types, and this may reduce the user's interest. According to the strategy proposed by Zhou *et al*. [32], given a candidate item, a user with a diverse set of interests is activated. We choose all items of session as candidate items. The

click rate of users can be improved if information on the candidate items is added to “session 5” in Figure 1. A large amount of information on the candidate items is a key aspect of session-related data, and can be used to tap into potential user preferences.

To better learn the preferences of users in their current session, this study introduces the fusion of the candidate items with the graph neural network model Furnish Charging Information Graph Neural Network (FCI-GNN), a graph neural network model based on session-based rec-ommendations, that includes information on the candidate items as well as global-level and session-level information. We construct global and session diagrams to obtain global-level and session-level item-embedding layers, respectively:

1. The global-level item-embedding layer linearly com-putes each item in a node's neighborhood set by using a session-aware attention mechanism in the global graph. The resulting representation of the neighborhood of the item is then aggregated with the representation of the node to generate a global-level embedding. To combine efficient messages into the representation of the current session, we use multiple aggregations to explore higher-order infor-mation.
2. The session-level item-embedding layer uses an at-tention mechanism to obtain the weights of distinct nodes to finally obtain the session-level embedding.
3. In addition, after merging global- and session-level embeddings, we provide a candidate-level representation layer that computes the soft attention score of the candi-date item with respect to the item in the current session by applying a target attention mechanism. The ultimate repre-sentation of the session is created by merging the candi-date-level embedding with the global and session levels, and linearly transforming them.

The main contributions of this work are as follows:

- To accommodate changes in the user’s intent during a session, we use attention mechanisms to calculate the relevance of their historical actions on items in the present session to the candidate items. This improves recommendation-related performance.
- The assumption of sequence independence is violated through the creation of messages from the global per-spective of cross-session, which is the process of ob-taining information from different sessions.
- We build a session graph and use it to obtain intricate relationships among the items in the given session. This enhances the representation of the session.

2. Preliminaries

We now define the problem that we consider here before describing the session graph and the global graph. Table 1 lists the notation used in this study.

Table 1. Notation used in this study.

Notation	Explanation
Input	
$\mathcal{T}$	Set of all session items, $\mathcal{T} = \{t_1, t_2, \dots, t_n\}$; the total number of items is n.
s	An anonymous session, $s = \{t_{s,1}, t_{s,2}, \dots, t_{s,m}\}$; m is the length of the session.
S	Set of all sessions.
Output	
$\hat{y}$	Output probability of item.
Graph	
$N_r(v)$, N_v^g	All represent node v’s r-order neighbor set on the global graph.
G_g	Global graph (undirected graph).
N_v^s	Node v’s neighbor set in the session graph.
G_s	Session graph (directed graph).
e_i,e_o,e_{i-o}, e_s	Four types of edges in a session graph, namely, in-edge, out-edge, two-way edge, and self-circulating edge.
Variable	
$\mathrm{x}_v^{g,(u)}$	Item v representation generated by the u-th layer information propagation in the global graph.
$\mathrm{x}_{N_{v_i}^g}$	Neighbor representation of item v_i in the global graph.
s	Vector representation of the current session s(letter s bolded).
x_v	Vector representation of (item) node v.
v_t	Candidate item.

2.1. Problem Setting

Session-based recommendation is designed to forecast the products that users will select according to information on their past sessions, rather than a long-term profile of their preference. Let *T*={*t*_1,*t*_2,…,*t*_*n* } be the set of items that the user has involved during all their sessions, where the total number of items in all sessions is n. The sequence of anonymous sessions comprises a series of interactions that can be described as a list “*s* = {“ “*t*” _ “*s*,1” ”, “ “*t*” _“*s*,2” “,…,”“ “*t*” _“*s*,*m*” ”} ” , where *t*_(*s*,*i*)∈*T* denotes the item t_i chosen during session s, and the length of s is m. According to the session-based recommendation, the main purpose of the model is to output the next chosen item *t*_(*s*,*i*+1) in the current session. The probability value of the candidate items y ̂is output by processing, the items are sorted, and the *K* highest-ranking items are selected to generate a sequence of suggested candidate items for the user.

Each item *t*_*i*∈*T* is encoded into a unified embedding space according to time sequence l. Let *x*_(l,*t*)∈*R*^*d* be the item vector, where d indicates the number of dimensions of the embedded item. Accordingly, each item is initialized to *x*_(0,*t*)∈*R*^(|*T*|). The embedding is performed based on one-hot embedding, and the items are converted into a d-dimensional potential vector space via the matrix *W*_0∈*R*^(*d*×|*T*|). A vector s represents each session embedding. In the following, we use v to denote the node and the item unless otherwise specified.

2.2. Global Graph Module

During the learning of graph-based models, the information contained in the graph structure defines the upper limit of the performance of the model. It is thus important for the graph model to contain as much information as possible, especially across sessions, to obtain good performance. To capture information on the transfer of global-level items, we use the method in [24] to generate a global graph and learn information between sessions.

The set of r-order neighbors of the item in the current session is defined as *N_r*（*v*）. The neighboring set of the set of items in the current session consists of items that have been interacted with in a certain step length r. Specifically, let the neighbor set of node v of the current session S_a be represented as:

N_r（*v_i^a*)={*v_j^b* |*v_i^a*=*v_(i^’)^b*∈*S_a*∩*S_b*;*v_j^a*∈*S_b*;*j*∈[*i*^'-*r*,*i*^'+*r*];*S_a*≠*S_b*}, where *v_i^a* is node i of session *S_a*, *i*^' is the sequential position of item *v_i^a* from session *S_a* in session *S_b* and *r* is a super-parameter that controls the transformation modeling between items. If r is out of range, it may capture noise from global-level information on item conversion. Note that in terms of gains in efficiency, our proposed model does not discriminate between the directions of global-level item conversion.

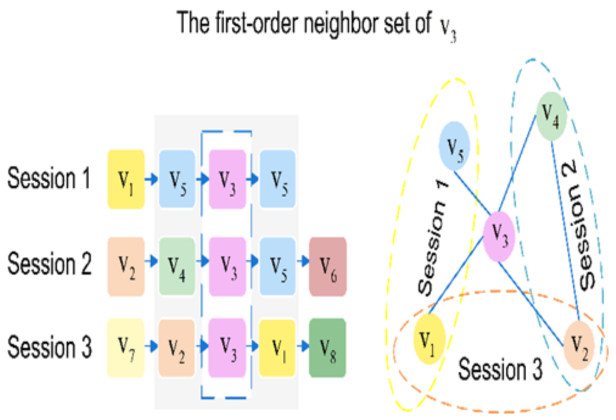


Figure 2. Construction of the global graph.

Moreover, we use the set of r-order neighboring items in all sessions to structure a global graph. Let the global graph be represented by *G_g*=(*V_g*,*E_g*), where *V_g*∈*T* is the set of clicked items in all sessions, *E_g*={*e_ij^g*} indicates all edge sets, and *e_ij^g* is the edge between nodes *v_i^s* and *v_j^s*. Each edge 〖(*v_i^g*,*v_j^g*)〗_(*v_j*∈*N_v^g*) corresponds to two paired terms in all sessions. To identify the significance of node *v_i*’s neighbors, we compute the weights of its neighboring edges. In other words, the frequency of the adjacent edges in all sessions is used to determine the weight of each edge 〖(*v_i^g*,*v_j^g*)□*v_j*∈*N_*(*v_i*)*^g*〗 because the set of r-order neighbors is undirected, the global graph *G_g* is an undirected weighted graph, and its topology is not dynamically updated during the testing phase. To maximize efficiency, node *v_i* preserves only K edges with the largest weight. Figure 2 shows the construction of the global graph.

2.3. Session Graph Module

It is important to mine the current session for data on the relationship of item conversion, which refers to such relationships between items as outgoing edges, incoming edges, bidirectional edges, and self-circulation. To learn session-level item representation, we use the method in [23] to model a session graph that can capture the complex graphical patterns of a session. For each session s={*t_*(*s*,1), *t_*(*s*,2),…, *t_*(*s*,*m*)}, we set the session graph to *G_s*=(*V_s*, *E_s*), where *V_s*∈*T* indicates the collection of selected items during session *s*, *E_s* presents the collection of edges, and *E_s*=(*v_i*,*v_j*,*e_ij^s*) is the edge between adjacent items in s. According to the relationship between items in the session graph, there are four types of edges (*v_i^s*,*v_j^s*,*e_ij^s*): *e_i*, *e_o*, *e_*(*i-o*), and *e_s*, where *e_i* indicates the incoming edge into an item, *e_o* denotes the outgoing edge, *e_*(*i-o*) defines a two-way edge between items, and *e_s* represents the item's own loop. Figure 3 shows the structure of the session graph.

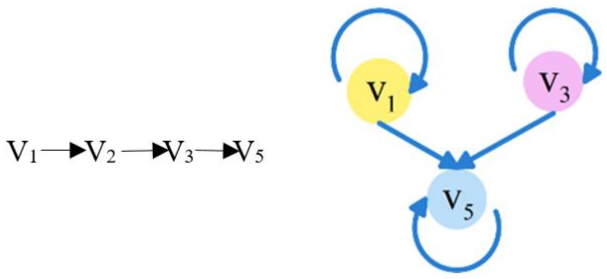


Figure 3. Structure of the session graph.

3. The Proposed Model

We now present the FCI-GNN model, which is based on graph neural network for session-based recommendation based on the fusion of candidate items. Figure 4 shows FCI-GNN model flow. The organizational framework of our model is made up of five main components, as shown in Figure 5:

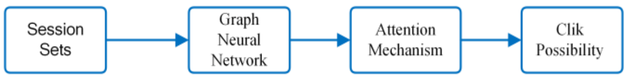


Figure 4. FCI-GNN model flow chart.

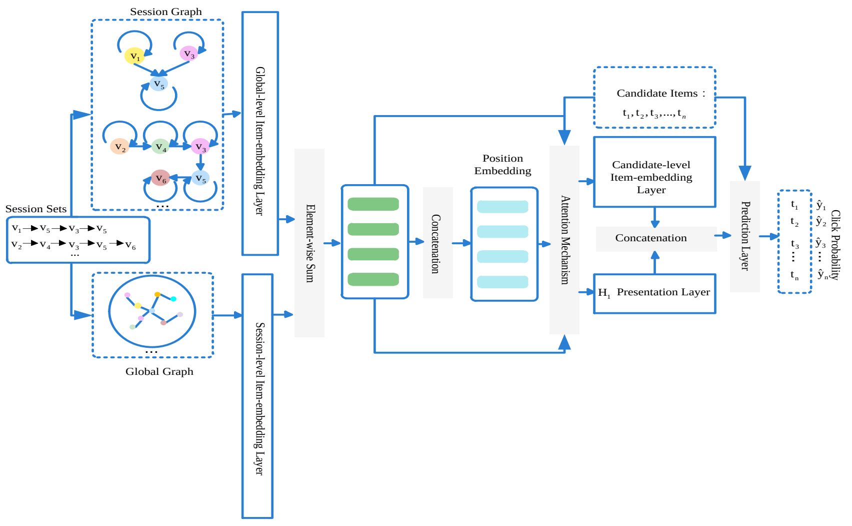

Session Graph
Session Sets
$v_1 \to v_5 \to v_3 \to v_5$
$v_2 \to v_4 \to v_3 \to v_5 \to v_6$
Global Graph
Global-level Item-embedding Layer
Session-level Item-embedding Layer
Element-wise Sum
Concatenation
Position Embedding
Attention Mechanism
Candidate Items : $t_1, t_2, t_3, \dots, t_n$
Candidate-level Item-embedding Layer
Concatenation
H_1 Presentation Layer
Prediction Layer
Click Probability

Figure 5. Organizational framework of the FCI-GNN model, where two sessions are used as an example in the session set. In practice, some items may not be clicked in one only session. We have thus listed v_3 and v_5 in two sessions to simulate this process.

- Global-level item-embedding layer: we use content on the global graph to compute nodes of the current session. This breaks the assumption of sequence independence and makes more potential information available to the current node. Because information on the node of the current session is affected by its r-order set of neighboring nodes, the attention mechanism is used to compute the affinity between the neighbor set and the current session of the current node.
- Session-level item-embedding layer: we compute nodes of the current session by using information on the session graph. We consider the relationships between items in the session sequence.
- Candidate-level item-embedding layer: to better grasp the user's intention, we analyze the influence of candidate items on nodes in the current session by using the attention mechanism.
- Position embedding: we use reverse location coding to transform the item-embedding vector to identify the location of each item to obtain valid information between the current and the predicted items.
- Prediction layer: we combine session-level, global-level, and candidate-level item-embedding representations, and linearly convert their results by using the click vector. The output probabilities are then sorted by using the top-K sorting method, and the final result is thus obtained.

3.1. Global-Level Item-Embedding Layer

Inspired by Li *et al*. [9] and Wang *et al*. [23], we update the characteristics of the global graph to obtain item conversion messages across sessions. We also use the graph attention network theory to generate the weights of the linked items. Each layer is composed of information dissemination and aggregation. The following illustrates how information dissemination and aggregation take place in each layer, and then extend to multiple layers.

Information dissemination: an item may involve numerous sessions, through which we can gain its available conversion relation to improve the forecast.

A simple technique for generating the first-order characteristics of the neighborhood of item v is the mean pooling approach. However, not all items of the r-order neighborhood set are related to the interests of the user in the current session. We think it is important to use session-aware attention to differentiate among items of the neighborhood set $N_r(v)$. We use session-aware attention to linearly compute each item in the set of neighbors to obtain a neighborhood representation of the given item, namely, $x_(N_v^g)$, illustrated in Equation (1).

$$x_(N_(v_i)^g) = \sum_(v_j \in N_(v_i)^g) ▒ 〖\xi(v_I,v_j)x_(v_j)〗 \quad (1)$$

where $\xi(v_i,v_j)$ represents the important weights of distinct neighbors. In particular, if such a neighboring item is close to the interest of the user in the current session, it is considered critical. $\Xi($〖v_i〗$_i,v_j)$ can be expressed as Equation (2).

$$\xi(v_i,v_j) = P_1^\intercal\ LeakyRelu(W_1\ [s\odot x_(v_j); \lambda_ij]) \quad (2)$$

where T denotes the transpose, Leaky Relu is the activation function used, $\odot$ denotes the elemental product, [;] denotes the splicing operation, $\lambda_ij \in R^m$ denotes the weights of the edges (v_i,v_j) of the global graph, $P_1 \in R^((d+1)\times 1)$ is a trainable parameter, and so is $W_1 \in R^((d+1) \times d)$. s is a feature of the current session that is obtained from the average value of its item representation. Illustrated in Equation (3).

$$s=(\sum_(v_i \in s) ▒ x_ (v_i))/(|s|) \quad (3)$$

Then, the coefficients of all neighbors of item v_i are normalized by applying the softmax activation function in Equation (4).

$$\xi(v_i, v_j) = (exp(\xi(v_i, v_j)))/(\sum_{v_k \in N_{v_i}^g} exp\ (\xi(v_i, v_k))) \quad (4)$$

Information aggregation: We combine the item representation x_v with the representation $x_{N_v^g}$ of its neighbor by using the aggregation function in Equation (5).

$$x_v^g = relu(W_2[x_v; x_{N_v^g}]) \tag{5}$$

Where the transformation weight is $W_2 \in R^{d \times 2d}$ and the activation function is ReLU.

The representation of an item in the aggregator layer is determined by it and its immediate surroundings. We use the aggregator to augment the item representation of the current session by mining higher-order interactions through numerous aggregations. Equation (6) is a representation of an item in layer u.

$$x_v^{g,(u)} = agg(x_v^{(u-1)}, x_{N_v^g}^{(u-1)}) \tag{6}$$

where $x_v^{(u-1)}$ denotes the representation of item v generated from the previous layer u-1 of information propagation. Let x_v^0 be the initial iteration of the propagation of x_v. Finally, the representation of layer u of the item is composed of its original representation and those of its neighbors that cycle to u. This allows for the integration of efficient information into the representation of the current session.

3.2. Session-Level Item-Embedding Layer

In the session graph module, we defined four types of edges for the relationships between items of the session graph: the incoming edge e_i, outgoing edge e_o, bidirectional edge e_{i-o}, and self-looping edge e_s. The aim is to learn the transformation of information on pairs of items in the current session. We now detail the learning of session-level representation.

We use an attention mechanism in the session graph to obtain the weights of distinct nodes because the neighbors of items have varying importance to them. The element-wise product and non-linear conversion are applied to compute the attention factor μ_{ij} in Equation (7).

$$\mu_{ij} = LeakyReLU(E_{e_{ij}}^{\intercal}(x_{v_i} \odot x_{v_j})) \tag{7}$$

where $E_* \in R^d$ represents the weight vector, μ_{ij} represents the importance weight between nodes v_i and v_j, and e_{ij} denotes the relationship between v_j and v_i.

There are four relations in the session graph that are trained for four weight vectors: E_{i}, E_{o}, E_{i-o}, and E_s. As not all pairs of nodes in the network are connected, only the attention factor μ_{ij} for node $j \in N_{v_i}^s$ is determined, where $N_{v_i}^s$ is the first-order neighbor of node v_i. We then apply the session graph to the model. We use the softmax function to normalize the attention weight and measure the coefficients of distinct nodes:

Because neighbors of the node v_i are different, the attention factor β in Equation (8) is asymmetric, meaning that the neighbors in this case do not contribute equally to each other's representations.

$$\beta_{ij} = \frac{\exp(LeakyReLU(E_{r_{ij}}^{\intercal}(x_{v_i} \odot x_{v_j})))}{\sum_{v_k \in N_{v_i}^s} \exp\left(LeakyReLU\left(E_{r_{ik}}^{\intercal}(x_{v_i} \odot x_{v_k})\right)\right)} \tag{8}$$

The result obtained by using Equation (8) can be linearly combined with features of the node to obtain its output, illustrated in Equation (9).

$$x_{v_i}^s = \sum_{v_j \in N_{v_i}^s} \beta_{ij} x_{v_j} \tag{9}$$

In summary, the representation of items in a session graph is composed of the features of items and their neighbors in the current session. Information on the conversation is further enhanced by using the attention mechanism.

3.3. Position Embedding

To prevent over-fitting, we use dropout in Equation (10) in the global-level representation. We estimate the embedding of the items by merging global-level and session-level representations, and we use sum pooling to get the result of fusion:

$$x_v^{g,(u)} = dropout(x_v^{g,(u)}) \tag{10}$$

Then, the vector of items for the current session is acquired by Equation (11), namely, $X=[x_{v_1^s}', x_{v_2^s}', \ldots, x_{v_m^s}']$, where m indicates the length of the series of the current session.

$$X_v' = x_v^{g,(u)} + x_{v_i}^s \tag{11}$$

As each item's location is unique, we transform the item-embedding vector by using position embedding to distinguish the locations of the items. We also use the matrix of trainable location embedding $Q=[q_1, q_2, \ldots, q_l]$, where $q_i \in R^d$ denotes the vector of location i.

Without loss of generality, we use reverse position embedding because the distance between the current item and the predicted item provides more efficient data than the data on the forward location. For instance, in "session "{1, 2, 3, ?}"," "3" is the third item in the sequence, and has a significant effect on the predicted items. However, its influence on the predicted items is minimal in session ""{3, 2, 1, 4, ?}"," where "3" is the first in the sequence. Information on the reverse location thus provides a more realistic representation of each item's importance. The location information is obtained by splicing as well as non-linear transformation by Equation (12).

$$\theta_i = tanh(W_3[x_{v_i^s}'; q_{m-i+1}] + b_3) \tag{12}$$

Where $b_3 \in R^d$ and $W_3 \in R^{d \times 2d}$ are trainable parameters.

We calculate the average of item representations of the current session to obtain session-related information, illustrated in Equation (13).

$$s' = 1/m \sum_{i=1}^{m} x_{v_i^s}' \tag{13}$$

We then use soft attention to learn the corresponding weights, illustrated in Equation (14).

$$\alpha_i = P_2^{\intercal}\, \sigma(W_4\theta_i + W_5 s' + b_4) \tag{14}$$

Where $W_4 \in R^{(d\times d)}$, $W_5 \in R^{(d\times d)}$, $P_2^{\intercal} \in R^d$, and $b_4 \in R^d$ are trainable parameters.

Following this, we use terms of the linear combination to generate the representation of the new session, illustrated in Equation (15).

$$H_1 = \sum_{i=1}^{m} \alpha_i\, x_{(v_i^s)}^{\prime} \tag{15}$$

This new representation H_1 contains the items involved in the current session.

3.4. Candidate-Level Item-Embedding Layer

According to the idea proposed by Yu *et al.* [29], we create an embedding of candidate items to extract more meaningful information on the current session and model different user intentions in it. It considers the correlation between the user's historical behavior and the items in their current session. We define all items to be predicted $\{t_1, t_2, \ldots, t_n\}$ as the candidate items v_t. In general, the items recommended to the user represent only a small portion of their preferences. To establish the model for this process, the soft attention scores of all items v_i in session s relative to each candidate item $v_t \in T$ are calculated by the attention mechanism, illustrated in Equation (16).

$$\rho_{(i,t)} = softmax(\varphi_{(i,t)}) = \frac{exp(v_t^{\intercal} W_6\, v_i)}{\sum_{j=1}^{n} exp(v_t^{\intercal} W_6 v_j)} \tag{16}$$

where $v_t \in T$ is the candidate item. A non-linear transformation is used to apply the weight matrix $W_6 \in R^{(d\times d)}$ to each node–candidate pair, and the softmax function normalizes the attention score.

For each session, we define $H_2 \in R^d$ as the interest of the user in candidate item v_t, namely, the candidate representation. It is calculated as Equation (17).

$$H_2 = \sum_{i=1}^{s_n} \rho_{(i,t)}\, v_i \tag{17}$$

Lastly, to construct the final representation of the session, we splice and linearly transform the session and the candidate representation generated by Equation (18).

$$H = W_7\,[H_1; H_2] \tag{18}$$

3.5. Prediction Layer

The probability of each candidate item of being chosen as the final recommendation is determined by its initial embedding and the final representation derived by Equation (18). The final output $\hat{y}$ can be generated through a dot product and the softmax function in Equation (19).

$$\hat{y} = softmax(\hat{z}) \qquad \hat{z} = H^{\intercal}\, v_t \tag{19}$$

where $\hat{z}$ represents the predicted score of the candidate item and $\hat{y}$ is the probability of the next clicked item. The recommended items are chosen from the top K items with the highest probability value in $\hat{y}$.

We use cross-entropy as the loss function in Equation (20).

$$L(\hat{y}) = -\sum_{i=1}^{n} \left[(1 - y_i)\, log(1 - \hat{y}_i) + y_i\, log(\hat{y}_i)\right] \tag{20}$$

Where y_i represents the one-hot encoding vector of the ground truth items.

4. Experiments

We performed detailed trials on three publicly available datasets to assess the efficacy of the FCI-GNN model and respond to the following research questions:

- RQ1: is FCI-GNN better than other cutting-edge SBRS methods?
- RQ2: does changing the value of "K" affect the performance of our model?
- RQ3: are all components of our model valid?
- RQ4: how does the number of neighbors influence the model?
- RQ5: how much influence do the number of dimensions of the embedding have on the model, and what is an appropriate value appropriate for this?

To answer the above questions in sequence, we now present basic information on the experiment, followed by a report and analysis of the results.

4.1. Datasets and Preprocessing

We used the Diginetica, Tmall, and Nowplaying datasets to assess the performance of the proposed model. Diginetica contains data on user interactions, and can be downloaded from the website of the Conference on Information and Knowledge Management (CIKM) Cup 2016. The tmall dataset is composed of anonymous data on purchases made by consumers on the tmall e-commerce site, and is available on the website of the IJCA15 competition. Zangerle *et al.* [30] published the Nowplaying dataset, which is a collection of the listening habits of Twitter users.

Following the preprocessing strategy provided by Rendle *et al.* [16], to have the same comparability in the follow-up research, when we preprocessed the three data sets, we removed the sessions with one length and less than five items. The Diginetica dataset contained 43,097 items, Tmall contained 40,728 items, and the Nowplaying dataset had 60, 417 items after preprocessing.

Moreover, each session was divided again to obtain more session-related data for training. For each session $s = \{t_{(s,1)}, t_{(s,2)}, \ldots, t_{(s,m)}\}$, we constructed a sequence of clicked items and the matching labels, i.e., $([t_{(s,1)}], t_{(s,2)})$, $([t_{(s,1)}, t_{(s,2)}], t_{(s,2)})$, …, ([*t*_(*s*, 1), *t*_(*s*, 2), …, *t*_(*s*, *m*-1)],*t*_(*s*, *m*)). They were used to train and test the three datasets. Table 2 lists statistical data from the datasets.

Table 2. Statistics of the datasets used.

Data	No. of items	No. of train	No. of test	No. of click	Avg. length
Tmall	40,728	351,268	25,898	818,479	6.69
Diginetica	43,097	719,470	60,858	982,961	5.12
Nowplaying	60,417	825,304	89,824	1,367,963	7.42

4.2. Evaluation Metrics

According to a previous study [11, 25], the validity of the model was tested by using two metrics: The Mean Reciprocal Rank (MRR) and precision. They are often used in recommender systems. P@K (precision) is a measure of a recommendation system's predictive accuracy, and MRR@K (mean reciprocal rank) indicates the average of reciprocal places in the correctly recommended items to rank the results of the recommendation. If the MRR is zero, this means that the correctly recommended item does not occur in the recommendation sequence. The higher its value is, the better is the predictive ability of the algorithm. K was set to either 10 or 20 in this experiment.

Baseline methods. Our method was compared with SBR representational methods. The 10 baseline models listed below were examined.

- POP suggests items based on items that appear the most frequently in the training dataset for the session. Items with a large number of clicks are more likely to be recommended.
- Item-KNN [17] suggests items depending on the extent to which items in the current session are related to those from past sessions.
- FPMC [16] merges matrix decomposition and first-order Markov chains to acquire sequential information on the user’s interests. It neglects the potential content of users when calculating recommendation scores.
- GRU4Rec [6] uses the gated recurrent neural network to obtain dependencies with long sequential distances.
- NARM [10] is an RNN-based model that uses the attention mechanism to acquire the user’s intent in the current session.
- STAMP [11] substitutes the RNN with an attention layer, and relies on the last item of the current session to acquire the user’s preference.
- SR-GNN [25] models sessions as graphs to capture item associations, and then uses an attention mechanism to record user preferences.
- FGNN [14] uses a multi-headed attention mechanism that aggregates information on an item's neighbors to derive the item representation.
- GCE-GNN [23] learns the session and global item-embedding layers by creating a session graph and a global graph.
- DHCN [27] uses a hypergraph to obtain higher-order information between items, and self-supervised learning to enhance the expressiveness of the diagram.

4.3. Hyperparameters Settings

Following previous studies [23, 29], we used a hidden vector with 100 dimensions, 20 iterations, and a batch size of 100 to training the FCI-GNN model. We assumed that a 10% random subset of the training set was the verification set. The Gaussian distribution was used to initialize all parameters, where their mean and standard deviation were 0 and 0.1, respectively. Our model used the Adam optimizer, with a degradation in the learning rate of 0.1 and a learning rate of 0.001 per three iterations, and a regularization factor of 10-5. Wang *et al*. [23], the number of our neighbors was set to 12. Note that we directly used the report results [23, 27] for the baseline models, because we used the same datasets and evaluation indicators as the relevant studies.

4.4. Overall Performance Comparison (RQ1)

Table 3 presents the experimental outcomes of the baseline methods and the FCI-GNN model on the three public datasets. We used 12 evaluation indicators, and our model outperformed the baseline models in 10 of them.

Among the traditional methods, POP delivered the weakest performance as it simply considers widely chosen items. The FPMC approach surpassed the POP algorithm, which demonstrates the significance of considering user interests in recommendation algorithms. Item-KNN delivered the best results on the diginetica and nowplaying datasets because it applies similarities between items to capture session-related information. This shows that there is some interdependence between sessions. Nevertheless, it struggled to obtain the relationship of consecutive transitions among items because it does not consider the chronological sequence of items within a session.

The first recurrent neural network model used for session-based recommendation was GRU4Rec. It uses recurrent neural networks on a session to achieve excellent predictive performance. Although the scores of GRU4Rec are worse than item-KNN in the diginetica dataset, it is pretty significant in all other datasets. It can be demonstrated that considering the sequence of sessions as a factor has a positive effect. However, session-based recommendation in general does not only consider the task of sequence modeling, but also the fact that the preferences of users may alter during the session. It is not sufficient to consider the sequential factor. The attention mechanisms used in NARM and STAMP perform much better than the one in GRU4Rec, which further suggests that attention mechanisms play a key role in enhancing the efficacy of recommendations.

The GNN-based approach was the most effective of all baseline approaches on the three datasets, which also indicates that the GNN is superior to the traditional method and deep learning-based methods. This is because when a GNN simulates relationships of item conversion in a session to model a graph, it gains better item dependencies than the other methods. It is thus better suited to session-based recommendation. Moreover, the GCE-GNN and DHCN outperformed the SR-GNN and FGNN because the SR-GNN and FGNN model only the current session, whereas the GCE-GNN and DHCN consider all sessions as well as the current one.

Because our approach incorporates the GNN, our proposed FCI-GNN recorded substantially superior metrics than traditional recommendation and deep learning-based recommendation on all three datasets. A detailed comparison with GNN-based methods follows:

Compared with the FGNN and SRGNN, which model only the current session, our approach achieved significant improvements on all metrics. The proposed FCI-GNN outperformed SRGNN by 17.66% on average on the Tmall dataset, by 7.09% on average on Diginetica, and by 18.61% on average on Nowplaying. This is because our method considers more cross-session information and information on the candidate items than the FGNN and SRGNN.

Our method outperformed the best-performing baseline algorithms, namely the GCE-GNN and DHCN, on 10 of the 12 metrics. The FCI-GNN surpassed the DHCN and GCE-GNN on both the Tmall and the Diginetica datasets. It outperformed GCE-GNN by 1.86% in terms of MRR@10 and 1.49% in terms of MRR@20 on the Tmall dataset. This demonstrates the efficiency of introducing attention mechanisms for candidate items while considering session- and global-level information. There is an exception on the Nowplaying dataset: DHCN delivered the best precision while FCI-GNN delivered the best MRR. It exceeded the DHCN by 5.84% on MRR@10 and 6.36% on MRR@20, and was superior to GCE-GNN by 3.74% on MRR@10 and 3.57% on MRR@20.

On the one hand, the significant improvement in the precision of DHCN occurred because hypergraph modeling can capture potential information about cross-sessional interactions on datasets with longer average session lengths. On the other hand, the FCI-GNN outperformed the DHCN on MRR because the addition of candidate items enriches the information on the current session. This indicates that our model has considerable room for improvement in terms of cross-session information. The improvement in MRR was more evident than that in precision, which reveals that the FCI-GNN model can significantly enhance the accuracy of rankings of recommended items.

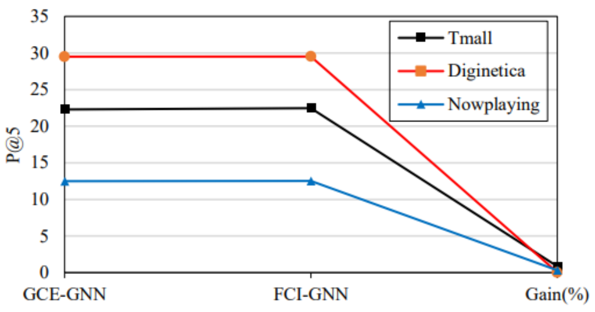
Table 3. Comparison of the methods. The best traditional model is marked in italics, the best deep learning method is underlined, and the best GNN model is marked in bold (we directly used the report results (Wang *et al*. [23], Xia *et al*. [27]) for the baseline models).

		Tmall		**Diginetica**		**Nowplaying**	
	Methods	**P@10**	**MRR@10**	**P@10**	**MRR@10**	**P@10**	**MRR@10**
Traditional method (a)	POP	1.67	0.88	0.76	0.26	1.86	0.83
	Item-KNN	6.65	3.11	25.07	10.77	10.96	4.55
	FPMC	13.10	7.12	15.43	6.20	5.28	2.68
Deep learning method (b)	GRU4Rec	9.47	5.78	17.93	7.73	6.74	4.40
	NARM	19.17	10.42	35.44	15.13	13.60	6.62
	STAMP	22.63	13.12	33.98	14.26	13.22	6.57
GNN method	SR-GNN(c)	23.41	13.45	38.42	16.89	14.17	7.15
	FGNN(d)	20.67	10.07	37.72	15.95	13.89	6.80
	GCE-GNN(e)	28.01	15.08	41.16	18.15	16.94	8.03
	DHCN(f)	26.22	14.60	40.21	17.59	17.35	7.87
ours	FCI-GNN	28.18	15.36	41.30	18.17	17.21	8.33
Gain (%)	c	20.38	14.20	7.50	7.58	21.45	16.50
	d	36.33	52.53	9.49	13.92	23.90	22.50
	e	0.61	1.86	0.34	0.11	1.59	3.74
	f	7.48	5.21	2.71	3.30	-0.81	5.84
	Methods	**P@20**	**MRR@20**	**P@20**	**MRR@20**	**P@20**	**MRR@20**
Traditional method (a)	POP	2.00	0.90	1.18	0.28	2.28	0.86
	Item-KNN	9.15	3.31	35.75	11.57	15.94	4.91
	FPMC	16.06	7.32	22.14	6.66	7.36	2.82
Deep learning method (b)	GRU4Rec	10.93	5.89	30.79	8.22	7.92	4.48
	NARM	23.30	10.70	48.32	16.00	18.59	6.93
	STAMP	26.47	13.36	46.62	15.13	17.66	6.88
GNN method	SR-GNN(c)	27.57	13.72	51.26	17.78	18.87	7.47
	FGNN(d)	25.24	10.39	50.58	16.84	18.78	7.15
	GCE-GNN(e)	33.42	15.42	54.22	19.04	22.37	8.40
	DHCN(f)	31.42	15.05	53.66	18.51	23.50	8.18
ours	FCI-GNN	33.63	15.65	54.32	19.08	22.65	8.70
Ain (%)	c	21.98	14.07	5.97	7.31	20.03	16.47
	d	33.24	50.63	7.39	13.30	20.61	21.68
	e	0.63	1.49	0.18	0.21	1.25	3.57
	f	7.03	3.99	1.23	3.08	-3.62	6.36

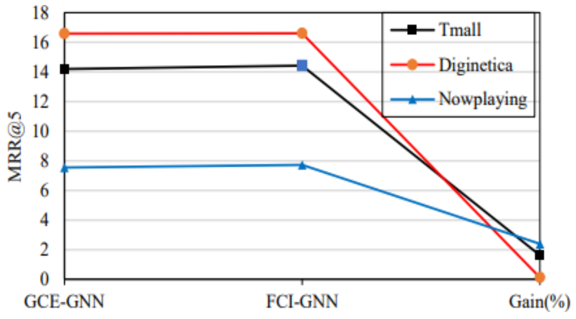
4.5. Effect of K on Recommendation Performance (RQ2)

The number of items suggested to the user may vary in number in practice. To simulate this, we compared the proposed model with the benchmark model GCE-GNN, which delivered the best performance on the Diginetica and Tmall datasets. To further assess the performance of the model, we change the values of K of MRR and precision to K=5 and K=30 for the three datasets.

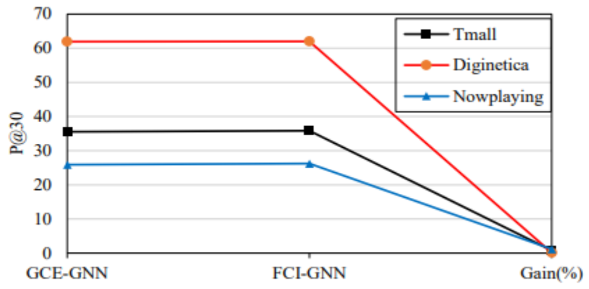
Table 4 presents the experimental findings for K=5 and K=30 on the three datasets. The FCI-GNN model outperformed the GCE-GNN on all evaluation metrics. Moreover, while its performance was only slightly superior on the Diginetica dataset, it was considerably better on the other two datasets. This shows that combining information on candidate items and global information in the Tmall and Nowplaying datasets had a significant influence on obtaining item information. The difference between the FCI-GNN model and the GCE-GNN model is that the FCI-GNN model introduces a candidate layer that considers the correlation between the user's historical behavior and the items in the current session. Figure 6 shows that this additional layer improves the recommendation performance, which enhances the effectiveness of the model.



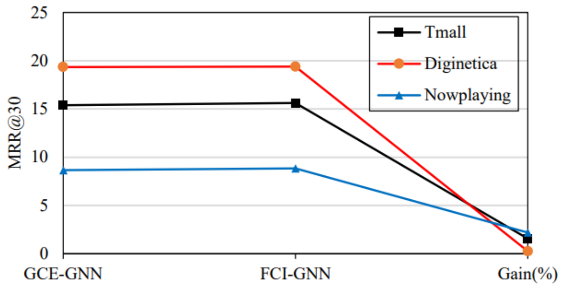
a) P@5 on datasets of tmall, diginetica and nowplaying.



b) MRR@5 on datasets of tmall, diginetica and nowplaying.



c) P@30 on datasets of tmall, diginetica and nowplaying.



d) MRR@30 on datasets of tmall, diginetica and nowplaying.

Figure 6. Experimental results on the Diginetica, Tmall and Nowplaying datasets in terms of P@5 and MRR@30.

Table 4. Experimental results for the proposed method for K=5 and 30 on the three datasets in comparison with the GCE-GNN.

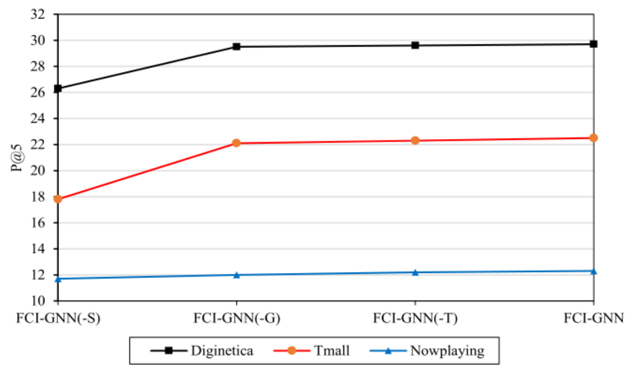
	Tmall		Diginetica		Nowplaying	
Algorithm	**P@5**	**MRR@5**	**P@5**	**MRR@5**	**P@5**	**MRR@5**
GCE-GNN	22.27	14.19	29.50	16.58	12.48	7.54
FCI-GNN	22.46	14.42	29.51	16.60	12.52	7.72
Gain(%)	0.85	1.62	0.03	0.12	0.32	2.39
Algorithm	**P@30**	**MRR@30**	**P@30**	**MRR@30**	**P@30**	**MRR@30**
GCE-GNN	35.56	15.38	61.93	19.34	25.92	8.65
FCI-GNN	35.83	15.62	62.00	19.39	26.22	8.84
Gain(%)	0.76	1.56	0.11	0.26	1.16	2.19

4.6. Ablation Study (RQ3)

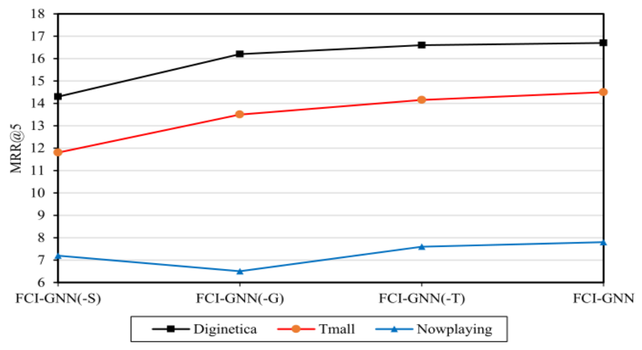
To determine the impact of each module on the performance of the FCI-GNN, three variants of it were studied on the three datasets: FCI-GNN(-G), FCI-GNN(-S), and FCI-GNN(-T).

- FCI-GNN(-G) simply removes the information contained in the global-level representation layer.
- FCI-GNN(-S) does not use the information of the session-level representation layer.
- FCI-GNN(-T) does not use information from the candidate-level representation layer, and is practically equivalent to the GCE-GNN.

Figure 7 shows the comparison between the proposed FCI-GNN and its variants. FCI-GNN(-S) delivered the worst performance because it could not capture information on the sequence of items and the relationship between items without session-related information. This highlights the necessity of using session-related information for analyzing the complicated transformation-related information on items. The removal of global information affected the performance of FCI-GNN(-G) as it provides information across sessions. When the candidate layers were deleted, the performance of FCI-GNN(-T) declined. This affected its results on Tmall and Nowplaying because the candidates items provide the most relevant information on users with a wide variety of interests.



a) P@5 on datasets of tmall, diginetica and nowplaying.

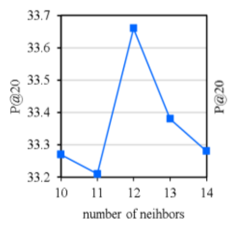


b) MRR@5 on datasets of tmall, diginetica and nowplaying.

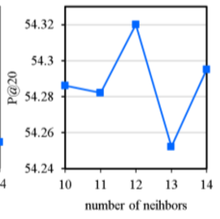
Figure 7. Comparison of different variants of the FCI-GNN in terms of P@5 and MRR@5 on the three datasets.

4.7. Effect of Different Numbers of Neighbors (RQ4)

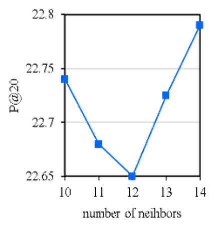
The number of neighbors also has a significant influence on the performance of the model. We used the metric P@20 to explore the impact of different numbers of neighbors on the performance of the proposed model on the Diginetica and Tmall datasets.



a) P@20 on Tmall dataset.



b) P@20 on Diginetica dataset.



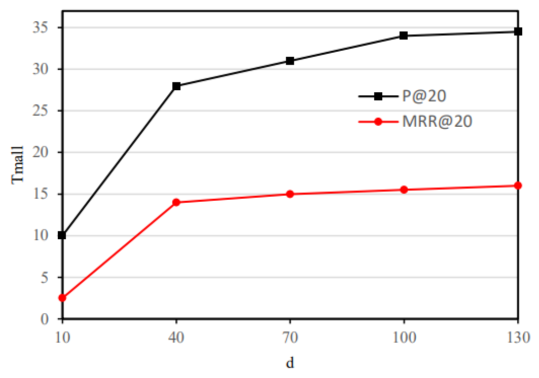
c) P@20 on Nowplaying dataset.

Figure 8. Experimental results on the Diginetica, Tmall and Nowplaying datasets in terms of P@20, obtained by using different numbers of neighbors.

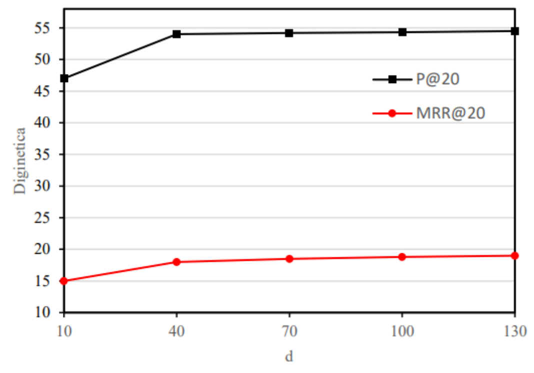
Figure 8 shows that when the number of neighbors was 12, our model was able to take the maximum value in the Diginetica and Tmall dataset. This is because when an item is clicked by a user, it is difficult to obtain reliable information if the number of items neighboring it is too small; if there are too many neighbors, the obtained information is noisy. As a result, we think that setting the number of neighbors of the diginetica and tmall datasets to 12 is appropriate in most circumstances. However, the model delivered its worst performance on the nowplaying dataset when the number of neighbors was 12. This is consistent with RQ1. Specifically, because the Nowlaying dataset had a long average session, it was difficult for our model to extract useful information on the sessions, and some of the information contained noise. We plan to examine this issue in future work.

4.8. Impact of Number of Dimensions of Embedding (RQ5)

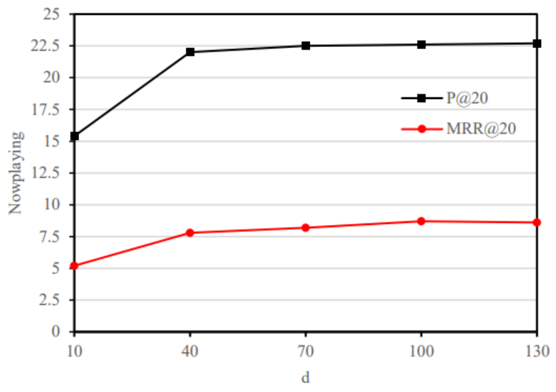
We trained the model with d=10, 40, 70, 100, and 130 as the number of dimensions of the embedding to investigate the influence on its overall performance.



a) P@20 and MRR@20 on tmall dataset.



b) P@20 and MRR@20 on diginetica dataset.



c) P@20 and MRR@20 on nowplaying dataset.

Figure 9. Experimental results in terms of MRR@20 and P@20 when using embeddings with varying numbers of dimensions d on the three datasets.

Figure 9 shows that the proposed model delivered the worst performance on all three datasets when the number of dimensions of the embedding was d=10. This is because the number of dimensions was too small, and led to insufficient information for the neural network to be trained. As the number of dimensions was increased, the values of P@20 and MRR@20 rose. However, after d=100, the results did not improve significantly. We thus think that 100 dimensions of the embedding should be used in the FCI-GNN model to ensure good performance.

5. Conclusions

In this paper, we offered a reconsideration of representations based on session-related information to generate recommendations for users. We argued that a user with a wide range of interests should have a rich set of candidate recommendations, rather than being limited to one type of content. We proposed the FCI-GNN, which first applies a graph neural network to obtain global- and session-level representations, and then uses an attention mechanism to acquire the representations of candidate items. Finally, the candidate-, session-, and global-level information is fused to acquire information on the complex dependencies among items in a session. The results of extensive experiments verified the validity of the FCI-GNN model. It outperformed state-of-the-art models in a vast majority of the cases considered. In the future we have a lot of work to do. First, we hope to further explore the useful information of the FCI-GNN model in extracting useful information from sessions with a long average session. Second, GNN-based models take a long time to train and consume a lot of resources. We want to reduce the complexity of the model without sacrificing its performance.

Declarations

Ethical Approval Written informed consent was obtained from all the participants before the publication of this study.

Competing interests The authors have no relevant interests to disclose.

Authors' contributions Wanhua Li wrote the main manuscript text. Yingjuan Sun, Bangzuo Zhang and Dongbing Pu gave comments and guidance. Yingjuan Sun and Dongbing Pu provided financial support. Qian Liu, Jingqi Xing and Yinghui Sun examined manuscripts. All authors reviewed the manuscript.

Funding This work was supported by the Jilin Province Industrial Technology Research and Development Funds of China (Grant Nos. 2019C052-9).

Availability of data and materials the raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

References

[1] Balabanovic M. and Shoham Y., "Fab: Content-Based, Collaborative Recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66-72. 1997. https://doi.org/10.1145/245108.245124

[2] Bonnin G. and Jannach D., "Automated Generation of Music Playlists: Survey and Experiments," *ACM Computing Surveys*, vol. 47, no. 2, pp. 1-35, 2014. https://doi.org/10.1145/2652481

[3] Chen T. and Wong R., "Handling Information Loss of Graph Neural Networks for Session-Based Recommendation," *in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, California, pp. 1172-1180, 2020. DOI:10.1145/3394486.3403170

[4] Fang H., Zhang D., Shu Y., and Guo G., "Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations," *ACM Transactions on Information Systems*, vol. 39, no. 1, pp. 1-42, 2020. https://doi.org/10.1145/3426723

[5] Fang J., "Session-based Recommendation with Self-Attention Networks," *arXiv Preprint*, vol. arXiv:2102.01922, pp. 1-12, 2021. https://doi.org/10.48550/arXiv.2102.01922

[6] Hidasi B., Karatzoglou A., Baltrunas L., and Tikk D., "Session-based Recommendations with Recurrent Neural Networks," *in Proceedings of the International Conference on Learning Representations*, Banff, pp. 1-10, 2014. https://www.semanticscholar.org/reader/e0021d61c2ab1334bc725852edd44597f4c65dff

[7] Huang Y., Huang T., Wang K., and Hwang W., "A Markov-based Recommendation Model for Exploring the Transfer of Learning on the Web," *Educational Technology and Society*, vol. 12, no. 2, pp. 144-162, 2009.

[8] Jannach D., Ludewig M., and Lerche L., "Session-Based Item Recommendation in E-Commerce: On Short-Term Intents, Reminders, Trends and Discounts," *User Modeling and User-Adapted Interaction*, vol. 27, no. 3, pp. 351-392. 2017. https://doi.org/10.1007/s11257-017-9194-1

[9] Li A., "Transition Information Enhanced Disentangled Graph Neural Networks for Session-based Recommendation," *arXiv Preprint*, vol. arXiv:2204.02119, pp. 1-34, 2022. https://doi.org/10.48550/arXiv.2204.02119

[10] Li J., Ren P., Chen Z., Ren Z., Lian T., and Ma J., "Neural Attentive Session-based Recommendation," *in Proceedings of the ACM on Conference on Information and Knowledge Management*, Singapore, pp. 1419-1428, 2017. https://doi.org/10.1145/3132847.313292

[11] Liu Q., Zeng Y., Mokhosi R., and Zhang H., "STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation," *in Proceedings of the 24th ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining, London, pp. 1831-1839, 2018. https://doi.org/10.1145/3219819.3219950

[12] Pang Y., Wu L., Shen Q., Zhang Y., Wei Z., Xu F., Chang E., Long B., and Pei J., "Heterogeneous Global Graph Neural Networks for Personalized Session-Based Recommendation," *in Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, Arizona, pp. 775-783, 2022. https://doi.org/10.1145/3488560.3498505

[13] Pazzani M. and Billsus D., *The Adaptive Web*, Springer, 2007.

[14] Qiu R., Li J., Huang Z., and Yin H., "Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks," *in Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, Beijing, pp. 579-588, 2019. https://doi.org/10.1145/3357384.3358010

[15] Quadrana M., Cremonesi P., and Jannach D., "Sequence-Aware Recommender Systems," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1-36, 2018. https://doi.org/10.1145/3190616

[16] Rendle S., Freudenthaler C., and Schmidt-Thieme L., "Factorizing Personalized Markov Chains for Next-Basket Recommendation," *in Proceedings of the 19th international Conference on World Wide Web*, Raleigh, pp. 811-820, 2010. https://doi.org/10.1145/1772690.177277

[17] Sarwar B., Karypis G., Konstan J., and Reidl J., "Item-Based Collaborative Filtering Recommendation Algorithms," *in Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, pp. 285-295, 2001. https://doi.org/10.1145/371920.372071

[18] Shani G., Heckerman D., and Brafman R., "An MDP-Based Recommender System," *in Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence*, Alberta, pp. 1265-1295, 2005.

[19] Wang H., Liu G., Liu A., Li Z., and Zheng K., "DMRAN: A Hierarchical Fine-Grained Attention-Based Network for Recommendation," *in Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, pp. 3698-3704, 2019. https://doi.org/10.24963/ijcai.2019/513

[20] Wang M., Ren P., Mei L., Chen Z., and Rijke M., "A Collaborative Session-based Recommendation Approach with Parallel Memory Modules," *in Proceedings of the 42nd International ACM SIGIR Conference*, Paris, pp. 345-354, 2019. https://doi.org/10.1145/3331184.333121

[21] Wang S., Cao L., Wang Y., Sheng Q., Orgun M., and Lian D., "A Survey on Session-Based Recommender Systems," *ACM Computing Surveys*, vol. 54, no. 7, pp. 1-38, 2021. https://doi.org/10.1145/3465401

[22] Wang S., Hu L., Wang Y., Sheng Q., and Cao L., "Modeling Multi-Purpose Sessions for Next-Item Recommendations via Mixture-Channel Purpose Routing Networks," *in Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, pp. 3771-3777, 2019. https://doi.org/10.24963/ijcai.2019/523

[23] Wang Z., Wei W., Cong G., Li X., Mao X., and Qiu M., "Global Context Enhanced Graph Neural Networks for Session-based Recommendation," *in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, China, pp. 169-178, 2020. https://doi.org/10.1145/3397271.340114

[24] Wang Z., Wei W., Cong G., Li X., Mao X., Qiu M., and Feng S., "Exploring Global Information for Session-based Recommendation," *arXiv Preprint*, vol. arXiv:2011.10173, pp. 1-14, 2020. https://doi.org/10.48550/arXiv.2011.10173

[25] Wu S., Tang Y., Zhu Y., Wang L., Xie X., and Tan T., "Session-based Recommendation with Graph Neural Networks," *in Proceedings of the AAAI Conference on Artificial Intelligence*, Vancouver, pp. 346-353, 2019. DOI:10.1609/aaai.v33i01.3301346

[26] Wu X., Liu Q., Chen E., He L., Lv J., Cao C., and Hu G., "Personalized Next-Song Recommendation in Online Karaokes," *in Proceedings of the 7th ACM Conference on Recommender Systems*, Hong Kong, pp. 137-140, 2013. https://doi.org/10.1145/2507157.2507215

[27] Xia X., Yin H., Yu J., Wang Q., Cui L., and Zhang X., "Self-Supervised Hypergraph Convolutional Networks for Session-Based Recommendation," *in Proceedings of the AAAI Conference on Artificial Intelligence*, Vancouver, pp. 4503-4511, 2021. https://doi.org/10.1609/aaai.v35i5.16578

[28] Xu C., Zhao P., Liu Y., Sheng V., Xu J., Zhuang F., Fang J., and Zhou X., "Graph Contextualized Self-Attention Network for Session-based Recommendation," *in Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, pp. 3940-3946, 2019. https://doi.org/10.24963/ijcai.2019/547

[29] Yu F., Zhu Y., Liu Q., Wu S., Wang L., and Tan T., "TAGNN: Target Attentive Graph Neural Networks for Session-Based Recommendation," *in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, China, pp. 1921-1924, 2020. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[30] Zangerle E., Pichl M., Gassler W., and Specht G., "Nowplaying Music Dataset: Extracting Listening

Behavior from Twitter," *in Proceedings of the 1st International Workshop on Internet-Scale Multimedia Management*, Orlando, pp. 21-26, 2014. https://doi.org/10.1145/2661714.266171

[31] Zhang M., Wu S., Gao M., Jiang X., Xu K., and Wang L., "Personalized Graph Neural Networks with Attention Mechanism for Session-Aware Recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pp. 3946-3957, 2020. DOI:10.1109/TKDE.2020.3031329

[32] Zhou G., Zhu X., Song C., Fan Y., Zhu H., Ma X., Yan Y., Jin J., Li H., and Gai K., "Deep Interest Network for Click-Through Rate Prediction," *in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, London, pp. 1059-1068, 2018. https://doi.org/10.1145/3219819.3219823

Yingjuan Sun was born in 1972. She received her Bachelor degree and Master degree in Computer Science from Northeast Normal University in 1995 and 2004, Ph.D. degree in Computer Application Technology from Jilin University in 2011, respectively. She currently is a Professor in the College of Computer Science and Technology, Changchun Normal University. Her main research interests include Recommender Systems, Machine Learning, and Data Mining.

Wanhua Li was born in 1998. In 2023, she obtained her master's degree from the School of Computer Science and Technology of Changchun Normal University. She is currently a Teacher at Pingsha New City School, Jinwan District, Zhuhai City. Her main research interests are Artificial Intelligence, Recommendation System, etc.

Jingqi Xing was born in 2001. She is currently studying at the College of Computer Science and Technology, Changchun Normal University., Changchun Normal University. Her main research interests are Artificial Intelligence, Recommendation System.

Bangzuo Zhang was born in 1971. He received his Bachelor degree in Computer Science from Northeast Normal University in 1995, and the Master degree and Ph.D. degree in Computer Application Technology from Jilin University in 2002 and 2009, respectively. He currently is an Associate Professor in the School of Information Science and Technology, Northeast Normal University. His main research interests include Information Retrieval, Recommender Systems, Machine Learning, and Data Mining.

Dongbing Pu was born in 1970. He received the Bachelor degree from the Northeast Normal University, China, in 1995, and the Master degree and Ph.D degree in Computer System Architecture from Jilin University in 2003 and 2010, respectively. He is currently an Associate Professor in the School of Information Science and Technology, Northeast Normal University. His research interests are Pattern Recognition, Intelligent Computation, Intelligent Control, and Embedded Systems.

Qian Liu was born in 1995. In 2024, he obtained his Master's degree from the College of Computer Science and Technology, Changchun Normal University. He main research interests are Artificial Intelligence, Recommendation System.

Yinghui Sun was born in 1975. She received her Master degree in computer science from Northeast Normal University in 2005. She currently is an Associate Professor in the College of Computer, Jilin Normal University. Her main research interests are Image Processing, Data Mining.