ITC 4/54

Information Technology and Control
Vol. 54 / No. 4/ 2025
pp. 1259-1270
DOI 10.5755/j01.itc.54.4.42577

Relationship-Enhanced Session-based Recommendation with Graph Neural Networks

Received 2025/08/18 | Accepted after revision 2025/10/18

HOW TO CITE: Ma, L., Liu, J. (2025). Relationship-Enhanced Session-based Recommendation with Graph Neural Networks. *Information Technology and Control*, 54(4), 1259-1270. https://doi.org/10.5755/j01.itc.54.4.42577

Relationship-Enhanced Session-based Recommendation with Graph Neural Networks

Lin Ma*
Shanxi Conservancy Technical Institute, China; e-mail: 19935513769@163.com

Jie Liu
The Computer Science and Technology College, Harbin Engineering University, Harbin 150000, China; e-mail: liujie@hrbeu.edu.cn

Corresponding author: 19935513769@163.com

The session-based recommendation systems analyze anonymous users' recent behavior data to infer their preferences and provide accurate recommendations. However, existing methods often fail to adequately leverage the click order and frequency of items within a session, thus lacking the ability to fully capture complex dependencies among items. To address these limitations, we propose a novel model named RESR-GNN. The key innovations of our work include: (1) a relation-enhanced session graph that incorporates click information to differentiate the importance of items; and (2) an integrated soft attention and multi-layer self-attention mechanism to comprehensively model pairwise item preferences within the entire session. Extensive experiments on two public datasets demonstrate that RESR-GNN consistently outperforms state-of-the-art baseline models. On the Diginetica and Yoochoose1/64 datasets, the evaluation metrics P@20 and MRR@20 were respectively improved by 3.15%, 5.51%, 1.53% and 2.36%, which proved the effectiveness of the proposed model.

KEYWORDS: Session-based recommendation, Graph neural networks, Session graph, Attention mechanism.

1. Introduction

The scale of data in the network era continues to expand with the advancement of Internet technology, and this data contains valuable information resources. Most traditional recommendation systems rely on users' basic profiles and their long-term interaction histories to make recommendations. However, such long-term interactions only reflect general user preferences, which tend to evolve over time. Moreover, personal information is often difficult to acquire. In contrast, Session-based Recommendation (SR) systems can capture short-term user interests, thereby providing more relevant recommendations. As a result, new techniques and methods in session-based recommendation are rapidly emerging and gaining widespread adoption.

Early approaches ranked items based on timestamps and employed Markov chains [13], [37], [26], [9], [19], to model transition relationships between items within a session, using current behavior to predict subsequent actions. While these methods achieved some success, they struggled to effectively leverage historical behavior information in longer session sequences.

This limitation was addressed through the introduction of recurrent neural networks (RNNs) [1], [32], [17], [2], [18]. Hidasi et al. [1] were the first to apply RNNs to model session sequences, improving the utilization of historical behavior information across long sessions. Subsequently, deep learning techniques have been widely adopted in SR. For instance, Li et al. [10] used global and local encoders to capture user interests, while Liu et al. [21] employed attention mechanisms and multi-layer perceptrons (MLPs) to model both general and current user interests. Attention mechanisms, which have been applied across various AI fields [4], [14], [34], [22], enable session-based recommenders to assign importance weights to items, reflecting user preferences.

Despite these advances, capturing complex transition interactions within session sequences remains challenging. Recent work by Wu et al. [29] proposed using graph neural networks (GNNs) for session-based recommendation. Their model, SR-GNN, constructs a graph for each session and uses GNNs along with a soft attention mechanism for training. Compared to linear sequences, session graphs can represent richer relational information, leading to better recommendation performance. Consequently, GNNs have become increasingly prominent in session-based recommendation research [5], [24].

However, existing methods still have limitations. First, session graphs often only record whether an item was clicked, failing to fully utilize click information. This can result in identical graph representations for different sessions, obscuring the user's degree of interest in specific items. Second, when using soft attention for global encoding, weight allocation typically considers only the relationship between the last item and all other items, neglecting potential interests among other items in the session.

To address these issues, we propose a Relationship-Enhanced Session-based Recommendation model with Graph Neural Networks (RESR-GNN). This model aims to improve recommendation performance by making better use of session click information and capturing intricate interest relationships among items within a session. RESR-GNN integrates techniques such as graph neural networks and attention mechanisms to enhance session representation.

Figure 1 illustrates the overall structure of RESR-GNN. First, all session sequences are converted into session graphs, where structural information is stored using a relationship-enhanced matrix that incorporates click information. Item node embeddings are learned through a gated graph neural network (GGNN), which aggregates feature information from the node itself and its neighbors. Then, a self-attention mechanism captures dependencies among all nodes in the session, while a soft attention mechanism computes correlations between the last node and other nodes. Finally, short-term and long-term interests are concatenated to form a session interest representation, which is used to compute recommendation scores for all candidate items.

The main contributions of this paper are as follows:

1 Click information about click orders and click counts in a given session is used to distinguish the importance of individual items, with the graph neural network applied to train model;

2 The self-attention mechanism and the soft attention mechanism are utilised to calculate item nodes' embedding vectors, which ensures the relevance of all items to the last item while obtaining the dependency between each item in the session;

3 The proposed model underwent comparison with publicly available datasets. Based on the two selected evaluation metrics, our experiments depicted notable enhancement in the recommendation performance of RESR-GNN.

2. Related Work

Traditional machine learning methods

Based on Markov chains, these methods model the transitions between items within a session. The system's subsequent state is exclusively tied to the current state, not the preceding one. Eirinaki et al. [13] used the Markov method and PageRank algorithm to provide personalized recommendations for users by analyzing the session information of users accessing web pages. The FPMC, which integrates matrix factorization and Markov chains to capture both long- and short-term user behavior information simultaneously, was proposed by Rendle et al [26] The k-nearest Neighbor algorithm (KNN) is a classification algorithm that is easy to understand and often used in machine learning algorithms. The session-based KNN algorithm [6], [15] compares the past sessions with the current whole session to make recommendations, and the item-based KNN algorithm only considers the similarity with the last item in a session.

Deep learning methods

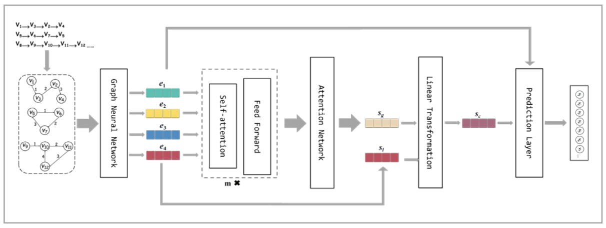
Deep learning is now widely employed in disciplines such as semantic segmentation [33], machine translation [16], object detection [28], image classification [35], recommender systems [23], [36], [11], and sentiment analysis [12]. Because of their superior sequence modeling capabilities, recurrent neural networks are commonly employed; Hidasi et al. [1] proposed the GRU4Rec, which makes use of the gated recurrent unit (GRU) to predict the behaviour of users based on their interactions. Subsequently, Tan et al. [32] proposed the improved RNN recommendation technique that utilizes data augmentation technology to boost the model's recommendation capability. Quadrana et al. [17] proposed an HRNN method for personalized recommendation by adding user identification attributes and using the session-level and the user-level GRU to provide personalized advice for users. Li et al. [10] suggested the Neural Attentive Session- based Recommendation, or NARM, employing the gated recurrent unit and attention mechanism to model session sequences. Convolutional neural networks (CNNs) may learn contextual information from each session through filtering and pooling operations. Yuan et al. [8] proposed the GRec to model sessions. This method designs a new encoder- decoder framework, which first inputs the session sequence after deleting some items into the encoder, and then predicts deleted components by the decoder. Tuan et al. [30] proposed the 3D CNN method to model sessions to obtain different content features.

Graph neural networks methods

The graph neural network is a kind of neural network which makes it possible to directly process unstructured data, such as graph data. Graph neural network combine convolution, gradient calculation, and other operations in deep neural networks with iterative graph propagation. When computing the features of a node, the hidden state of the current node must be updated by integrating the feature information of its neighbours to obtain richer relational information. Wu et al. [29] proposed a SR-GNN method. This method uses GGNN to learn each node's embedding representation. Get final session representation by combining short-term and long-term interests. In order to get the complicated transformation relationship between items in the customized scenario, the A-PGNN method proposed by Zhang et al [20] using the personalized graph neural network. Wang et al. [31] suggested the MGNN-SPred method. This method constructed the Multi-Relational Item Graph (MRIG) for all sessions and utilized the gated mechanism to fuse target and auxiliary behaviors of the user. Qiu et al. [25] proposed an FGNN method using the Weighted Graph Attentional Layer (WGAT) and the readout function to compute the session representation. Based on the SR-GNN method, Yu et al. [7] proposed the TA-GNN method by using the target attention mechanism to focus on the user's interest in different items in the session.

Figure 1

Structure diagram of the RESR-GNN model.

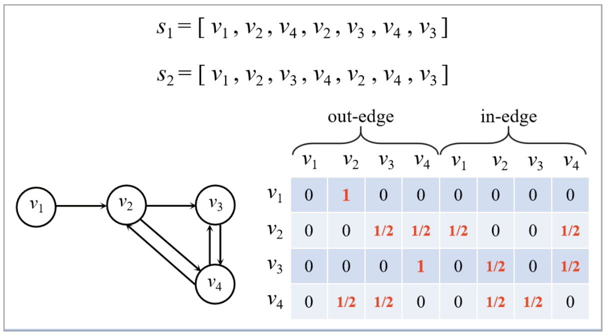


3. Method

In this chapter, we introduce the specific implementation process of the RESR-GNN model. By constructing a relationship enhancement matrix, we can obtain more comprehensive information relationships among various items. Using a combination of soft attention mechanism and self-attention mechanism, we can fully capture the interest dependencies among items in the entire conversation sequence. We also use residual connections to alleviate the problem of gradient disappearance that may occur in multi-layer neural networks. Next, we will introduce each part:

Notations Definition

Session-based recommendation according to current session behavioral data to predict a user's next click. $V=\{v_1,v_2,v_3,\ldots,v_n\}$ represents the set of items in all sessions. A session s is denoted as $s=[v_{s,1},v_{s,2},\ldots,v_{s,m}]$, where $v_{s,i}\in V$, m represents the length of the current conversation. The model predicts the following click behavior $v_{s,m+1}$ through session information of the user and finally outputs the recommendation probability $\hat{y}$ of all candidates. Finally, we choose the Top-k items in $\hat{y}$ as candidates for recommendation.



$s_1=[v_1,v_2,v_4,v_2,v_3,v_4,v_3]$

$s_2=[v_1,v_2,v_3,v_4,v_2,v_4,v_3]$

	out-edge				in-edge			
	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4
v_1	0	1	0	0	0	0	0	0
v_2	0	0	1/2	1/2	1/2	0	0	1/2
v_3	0	0	0	1	0	1/2	0	1/2
v_4	0	1/2	1/2	0	0	1/2	1/2	0

Figure 2
Examples of sessions s1 and s2.

The relationship-enhanced session graph

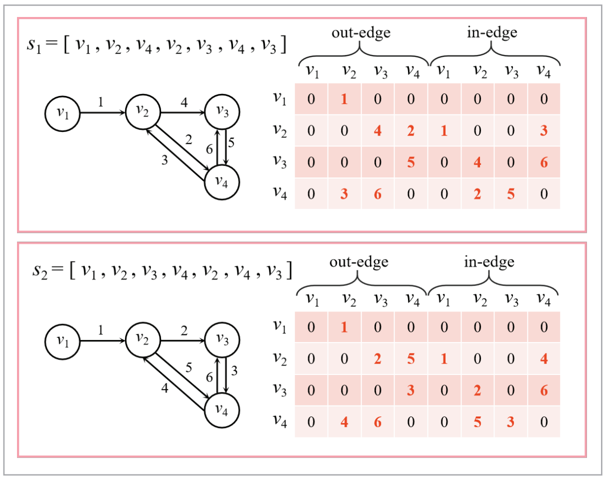
The click orders and counts contained in click information of the session affect the recommendation effect. Missing records of item click orders and counts in the session graph may result in different session sequences producing the same session graph and connection matrix. The importance of items cannot be distinguished. As shown in Figure 2, session s_1 and session s_2 construct the same session graph and connection matrix. For instance, node 2 has outgoing edges to both node 3 and node 4. Therefore, after regularization, the outgoing edge matrix of node 2 has a value of 1/2 for both node 3 and node 4.

We predict the next click of a session according to the order in which each item is clicked. This means that lower ranked items are more important. In addition, the frequency of the item's occurrence within the session sequence can indicate the degree of focus of the item in the mind of the user. Therefore, we design the relationship-enhanced session graph corresponding to each session, and all click information of items is integrated. For session graph, we assign edge weights to each edge, and values are incremented from 1. The edges of the items that are clicked later have greater weights. Accumulate the edge weights of the items clicked multiple numbers to obtain the relationship-enhanced session graph $\mathrm{G}_s=\{\mathrm{E}_s,\mathrm{V}_s\}$. Here V_s represents all session graph nodes and E_s represents all session graph edges.

It defines a multiset E_{set} for the session $s=[v_{s,1},v_{s,2},v_{s,3},\ldots,v_{s,m}]$, stores connection edges in the relationship-enhanced session graph G_s into E_{set} in order, and defines a boundary-finding function named *Index*.

$$\mathrm{E}_{set}=\{e(v_{s,1},v_{s,2}),e(v_{s,2},v_{s,3}),\ldots\} \tag{1}$$

Figure 3
Relationship-enhanced graphs and relationship-enhanced matrixes corresponding to sessions s_1 and s_2.



$s_1=[v_1,v_2,v_4,v_2,v_3,v_4,v_3]$

	out-edge				in-edge			
	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4
v_1	0	1	0	0	0	0	0	0
v_2	0	0	4	2	1	0	0	3
v_3	0	0	0	5	0	4	0	6
v_4	0	3	6	0	0	2	5	0

$s_2=[v_1,v_2,v_3,v_4,v_2,v_4,v_3]$

	out-edge				in-edge			
	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4
v_1	0	1	0	0	0	0	0	0
v_2	0	0	2	5	1	0	0	4
v_3	0	0	0	3	0	2	0	6
v_4	0	4	6	0	0	5	3	0

$$I = (e(v_{s,1}, v_{s,2})) = \begin{cases} x \mid x = Index(e(v_{s,1}, v_{s,2}), \mathrm{E}_{\mathrm{set}}) \\ \text{, if } e(v_{s,1}, v_{s,2}) \ in \ \mathrm{E}_{\mathrm{set}} \end{cases} \tag{2}$$

A function calculation is used to obtain the corresponding value of edge $e(v_{s,1}, v_{s,2})$ in $\mathrm{E}_{\mathrm{set}}$, denoted as *x*, where $e(v_{s,1}, v_{s,2})$ means that the user clicks item $v_{s,2}$ after $v_{s,1}$; *I* is the set that holds the edge values found.

The relationship-enhanced matrix R_s stores the information in the corresponding session graph. If edge *e* belongs to $\mathrm{E}_{\mathrm{set}}$, the accumulated edge weight values are the values in the corresponding matrix. Otherwise, it is marked as 0. The relationship-enhanced session diagram and matrix for sessions s_1 and s_2 are shown in Figure 3. The out- and in- edge matrix, $\mathrm{R}_s^{(out)}$ and $\mathrm{R}_s^{(in)}$, are distinguished from the relationship-enhanced matrix and concatenated, where the $\mathrm{R}_s^{(out)}$ is normalized, $\mathrm{R}_s^{(in)} = \mathrm{R}_s^{(out)\mathrm{T}}$.

$$\mathrm{R}_s^{(out)} = \begin{cases} sum(I(e(v_{s,i}, v_{s,j}))), e(v_{s,i}, v_{s,j}) \in \mathrm{E}_s \\ 0 \qquad\qquad\qquad , e(v_{s,i}, v_{s,j}) \notin \mathrm{E}_s \end{cases} \tag{3}$$

Learning item embedding on RESR-GNN

The embedding vectors of each item node are then computed by gated graph neural networks. First, we randomly initialize all item nodes to obtain an initial vector representation $e \in R^d$ for each node $v \in V$, d represents the embedding dimension of the item. Model uses GGNN to update nodes' embedding in a session G_s. The specific process of $v_{s,i}$ is as follows:

$$a_{s,i}^{(t)} = \mathrm{R}_{s,i:}[e_1^{(t-1)}, e_2^{(t-1)}, ..., e_m^{(t-1)}]^{\mathrm{T}} W + b_i \tag{4}$$

$$z_{s,i}^{(t)} = \sigma(W_z a_{s,i}^{(t)} + U_z e_i^{(t-1)}) \tag{5}$$

$$r_{s,i}^{(t)} = \sigma(W_r a_{s,i}^{(t)} + U_r e_i^{(t-1)}) \tag{6}$$

$$\tilde{e}_i^{(t)} = \tanh(W_t a_{s,i}^{(t)} + U_t(r_{s,i}^{(t)} \odot e_i^{(t-1)})) \tag{7}$$

$$e_i^{(t)} = (1 - z_{s,i}^{(t)}) \odot e_i^{(t-1)} + z_{s,i}^{(t)} \odot \tilde{e}_i^{(t)} \tag{8}$$

In Equation (4): $a_{s,i}^{(t)} \in R^{2d}$ is the vector representation of node v_i after aggregating the information of its neighbors at time *t*; $\mathrm{R}_{s,i}$ are the two columns corresponding to node $v_{s,i}$ in the R_s; $[e_1^{(t-1)}, e_2^{(t-1)}, ..., e_m^{(t-1)}]$ represents the vector representation of each item node of the session *s* at time *t*–1;

Equations (5)-(6) show the calculation formulas of update gate $z_{s,i}^{(t)}$ and reset gate $r_{s,i}^{(t)}$. They determine what information is to be retained and discarded. The candidate states are shown in Equation (7). The final state is shown in Equation (8), jointly determined by the previous time state and the current candidate state. $W_r, W_z, W_t \in R^{2d \times d}$ and $U_r, U_z, U_t \in R^{d \times d}$ control the weights.

Generating session embedding

In this section, we compute the final session representation. The item embeddings trained by GGNN will be fed to the self-attention mechanism for training. This reduces the reliance on external information by utilizing self-attention mechanisms to better focus on all inputs in the session. The process is as follows:

$$Q = W^q E = W^q [e_1, e_2, e_3, ..., e_m] \tag{9}$$

$$K = W^k E = W^k [e_1, e_2, e_3, ..., e_m] \tag{10}$$

$$V = W^v E = W^v [e_1, e_2, e_3, ..., e_m] \tag{11}$$

$$H = \mathrm{soft\,max}\left(\frac{(W^q E)(W^k E)^{\mathrm{T}}}{\sqrt{d}}\right)(W^v E) \tag{12}$$

In Equations (9)-(11): *Q* denotes the query matrix, *K* denotes the key matrix, and *V* denotes the value matrix. W^q, W^k and W^v are the parameter matrixes of the linear mapping. Softmax represents the activation function. *H* represents the output.

We add a feedforward network layer after a self-attention mechanism layer. We utilize a non-linear activation function to improve the model's non-linear capability, add residual connection to mitigate the loss incurred during model transmission, and add dropout to reduce the potential issue of overfitting.

$$P = H + \text{Dropout}(\text{LeakyRELU}(HW_1 + b_1)W_2 + b_2 \quad (13)$$

In order to collect additional feature information from the nodes, we created a multi-layered self-attention mechanism. The definition is as follows: the first layer input are the item vectors obtained by training the gated graph neural network.

$$P^{(m)} = multi - layer(P^{(m-1)}) \quad (14)$$

User's interests and preferences change dynamically. In a session, the long-term interest is reflected in all clicked items, which recently clicked can better reflect the short- term interest. We obtain the final interest representation by combining two interests. The embedding vector e_m of last item $v_{s,m}$ in session s is used to represent the short-term interest. Applying soft attention mechanism to calculate long-term interests.

$$s_l = e_m \quad (15)$$

$$\alpha_i = q^{\mathrm{T}}\sigma(W_1 p_m + W_2 p_i + b) \quad (16)$$

$$s_g = \sum_{i=1}^{m} \alpha_i p_i \quad (17)$$

where W_1, $W_2 \in R^{d\times d}$ and $q \in R^d$ are the training parameters. Finally, a linear transformation of the short- and long-term interests was used to obtain the final session interest representation.

$$s_c = W_3[s_l ; s_g] \quad (18)$$

Recommendation prediction

For each candidate item, we then calculate the score. Calculation of the model's output vector by means of a softmax function.

$$\widehat{z}_i = s_c^{\mathrm{T}} e_i \quad (19)$$

$$\hat{y} = \text{softmax}(\hat{z}) \quad (20)$$

where $\hat{y}$ is the probability of selecting item nodes next time; $\hat{z}$ represents all candidates' recommendation scores.

$$L = -\sum_{i=1}^{n} y_i \log(\widehat{y_i}) + (1 - y_i)\log(1 - \widehat{y_i}) \quad (21)$$

The cross-entropy loss function is selected for the experiment, where y_i and $\hat{y}_i$ represent the truth and predicted value, respectively. This model is trained using Back-Propagation Through Time (BPTT).

4. Experiments

Datasets

We select two commonly representative datasets, Diginetica and Yoochoose, for experiments. Table 1 shows the datasets information. The Diginetica dataset is from CIKM Cup 2016. In our experiments, we selected only published transaction data. First, the data must be pre-processed according to the previous studies [10], [21]. All sessions that contain an item in the session and items that appear less than five times in the dataset are filtered out. At the same time, filter out the conversation information with a length of 1. Next, another dataset of the experiment is introduced. The Yoochoose dataset is from the RecSys Challenge 2015. We selected users' browsing activities on e-commerce websites in the past six months. We are using the same method to preprocess the dataset. Due to the importance of the Yoochoose dataset, in order to minimise the possible impact of time changes on user preferences, we choose the Yoochoose1/64 dataset with the most recent split time for the experiments.

Table 1
Specific statistics of datasets.

Statistics	Diginetica	Yoochoose1/64
Clicks	982,961	557,248
Items	43,097	16,766
Training sessions	719,470	369,859
Test sessions	60,858	55,898
Average length	5.12	6.16

Furthermore, we refer to the pre-processing method of [32]. Multiple sequences and corresponding label data are generated by segmenting each input session sequence. For example, we divide a input session

$s=[v_{s,1},v_{s,2},...,v_{s,m}]$ into many subsequences. Such as $([v_{s,1},v_{s,2}],L(v_{s,3}))$, where $L(v_{s,3})$ is the label data corresponding to the session, representing the next clicked item. $[v_{s,1},v_{s,2}]$ is the session sequence generated by segmentation.

Evaluation metrics

P@20 is the accuracy of correct recommendations among the top 20 recommendations. The mathematical formula is as follows:

$$P@20=\frac{H_{hit}}{N} \tag{22}$$

where N is total items; H_{hit} is the number of correct recommendations in the recommended list.

MRR@20 is the average ranking of correct recommendations in the top 20 recommendations. Recommendations that are not within 20 will be shown with 0. The mathematical formula is as follows:

$$MRR@20=\frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i} \tag{23}$$

where N is total items; $rank_i$ is the rank given to the i recommended item in the list.

Parameter setup

Next, we introduce the experimental parameter Settings. Based on the parameter settings of the previous model, the parameters of the RESR-GNN model are designed. Set the hidden layer dimension to 100. Selecting appropriate parameter values can improve the running speed and achieve the optimal convergence accuracy. The initial vector is randomly dimensioned to 100. Set the batch size to 100 and model epochs to 30. The dropout parameter is set to 0.2. 0.001 is chosen as the learning rate coefficient and 0.1 is the learning rate decay coefficient. In the experiments, the learning rate is changed every three epochs. This experiment gives a penalty of 10-5 for L2 to prevent overfitting. The Adam optimiser is utilized to optimiser the parameters in the model.

Baseline models

To provide a more perceptible evaluation of the recommendation performance of models, we contrast the proposed RESR-GNN with baseline models.

- **POP.** This model thinks about the popularity of every item. Items clicked on will be ranked by how often they have been accessed and recommendations will be made based on the ranking. Items that are clicked on more often are more likely to be recommended.
- **Item-KNN** [3]. It computes the similarity between the clicked item and candidate items, finding out the K items that are most similar to the clicked item for the user to recommend.
- **BPR-MF** [27]. It optimizes the ranking of items using stochastic gradient descent (SGD), which is appropriate for implicit feedback data.

Table 2

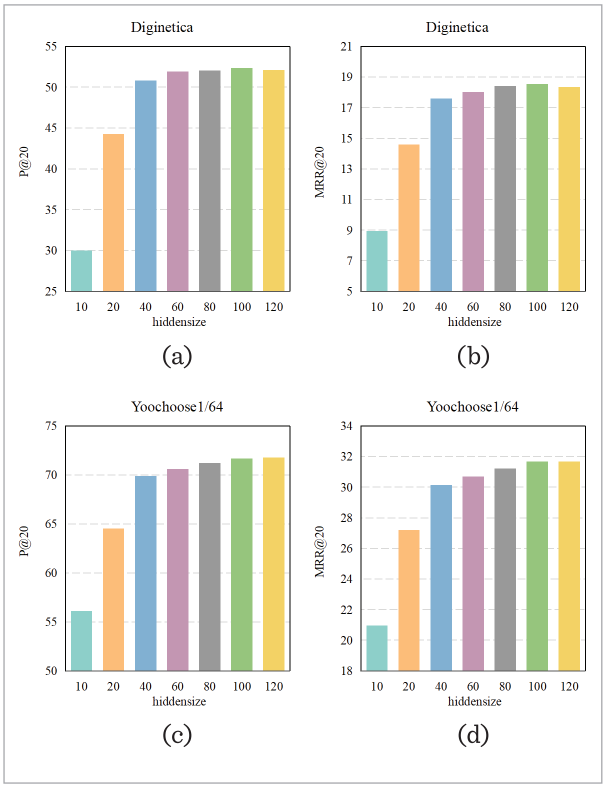
Comparison of the performance of the RESR-GNN with other baseline models.

Method	Diginetica		Yoochoose1/64	
	P@20(%)	MRR@20(%)	P@20(%)	MRR@20(%)
POP	0.89	0.20	6.71	1.65
Item-KNN	35.75	11.57	51.60	21.81
BPR-MF	5.24	1.98	31.31	12.08
FPMC	26.53	6.95	45.62	15.01
GRU4Rec	29.45	8.33	60.64	22.89
NARM	49.70	16.17	68.32	28.63
STAMP	45.64	14.32	68.74	29.67
SR-GNN	50.73	17.59	70.57	30.94
RESR-GNN	52.33	18.56	71.65	31.67

- **FPMC** [26]. This model simultaneously collects data on the user's long-term interest and recent behaviour using the two techniques of Markov chain and matrix factorisation. This model is a classic hybrid method applied to session-based recommendation.
- **GRU4Rec** [1]. By stacking multiple GRUs to improve performance, this is the first application of RNN for session-based recommendations.
- **NARM** [10]. It makes use of GRU to model sessions while adding an attention mechanism that causes the method to pay closer attention to the user's primary interests.
- **STAMP** [21]. To make recommendations to users, it uses a short-term attention priority mechanism that combines short-term and long-term interests.
- **SR-GNN** [29]. It applies GNN to the session-based recommendation. Using GGNN encodes items to obtain complex transformation features between items.

Figure 4

Performance of various hidden size dimensions.



Comparison with baseline models

On both the Diginetica and Yoochoose1/64 datasets, we compare RESR-GNN with the aforementioned baseline models. We use P@20 and MRR@20 as uniform metrics to evaluate the recommendation performance of all models. The experimental results on the above two dataset show that RESR-GNN achieves the best result. The specific experimental results are shown in Table 2.

Among the traditional recommendation methods, BPR-MF outperforms POP in recommendation performance, demonstrating the value of personalized recommendation. Item-KNN and FPMC are superior to POP and BPR-MF, which indicates the validity of calculating the similarity between items in Item-KNN and using the Markov chain in FPMC.

Model performance can be improved by applying deep learning technology. GRU4Rec is the first to use RNN, for the most part, it outperforms traditional recommendation models. In the experimental results, GRU4Rec performs worse than NARM and STAMP. This indicates that NARM has more advantages in obtaining the main interest. Using short-term interest can improve model performance and provide more accurate recommendations.

By contrasting the recommendation effect of SR-GNN with the previous models, the feasibility of applying GNN and the importance of further research are explained. SR- GNN builds the session sequence as the graph structure, which can fully consider the complex transition relationship between the present node and its adjacent nodes compared with the sequential structure. RESR-GNN classifies item's importance by clicking information. Compared with SR- GNN, it further reflects the user's degree of interest in different items. Furthermore, the self-attention mechanism also captures more inter-item dependencies. Our proposed RESR-GNN model performs effectively on both P@20 and MRR@20.

Method analysis

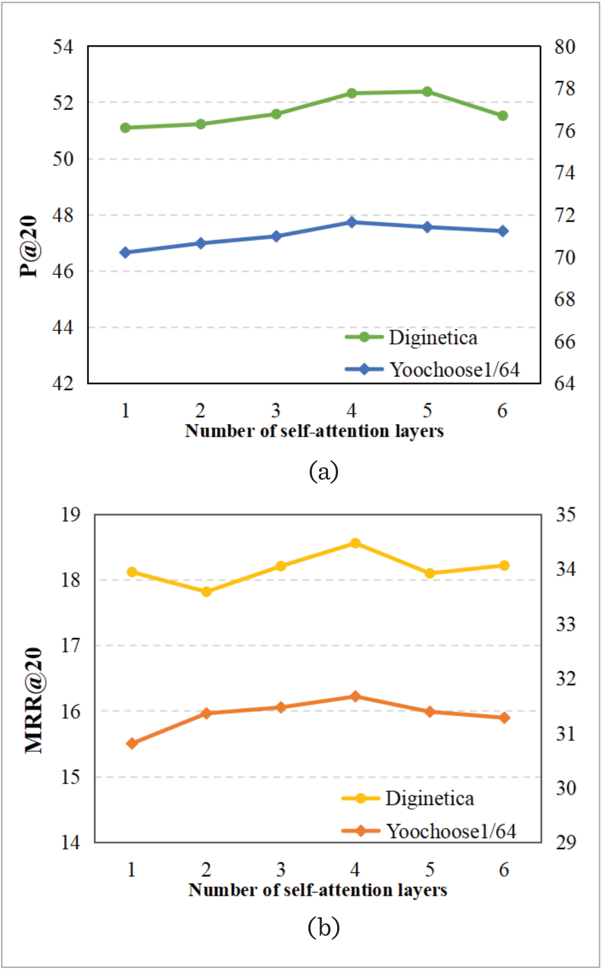
Impact of the dimension of hidden size.

We set different hidden layer dimensions to explore our model's performance and select seven dimensions for experiments. Under different hidden layer dimensions, the recommendation performance of RESR-GNN is shown in Figure 4. From the experiment, we can conclude that our model can learn more feature information by choosing an appropriate hidden layer's dimension. However, if we select a sizeable hidden layer dimension, the training complexity will increase, making the model's performance decline.

The choice of a hidden layer size of 100 was based on a combination of common practice in the field, empirical validation on our validation set, and a balance between model capacity and computational efficiency.

Figure 5

Performance of different layers of self-attention.



Impact of the number of self-attention layer

We choose the number of layers from 1 to 6 to train RESR-GNN and analyze its recommendation performance. The experimental results of different layers are shown in Figure 5.

The results show that selecting the appropriate number of layers of the self-attention mechanism can improve RESR- GNN recommendation performance, and the multi-layer networks can allow the model to learn richer abstract features through multiple information updates. Model performance may degrade as the number of layers increases. The reason that affects the recommendation effect may be the problem of disappearing gradients caused by deepening network layers.

Table 3

Comparison of the performance of the relationship-enhanced matrixes.

Method	Diginetica		Yoochoose1/64	
	P@20(%)	MRR@20(%)	P@20(%)	MRR@20(%)
SR-GNN	50.73	17.59	70.57	30.94
RESR-GNN (soft attention)	51.17	17.84	70.87	31.18

Ablation experiment

To verify that the proposed relationship-enhanced matrix can improve the performance of the conversation recommendation model, we conducted an ablation experiment. We compared the model with SR-GNN. The specific experimental results are shown in the table 3. From the table, it can be seen that by using the relationship-enhanced matrix, the recommendation performance of the model has improved to a certain extent on the Diginetica and Yoochoose1/64 datasets. RESR-GNN (with soft attention) indicates that the current model is consistent with other models and is trained solely using the soft attention mechanism.

5. Conclusion

We proposed a new RESR-GNN model for session-based recommendation. RESR-GNN can better distinguish the importance of different items and make full use of session information We use a multi-layered self- attention mechanism to focus on the interdependencies of all items in a session. To make full use of the correlation between items in the session and improve our model's comprehensive recommendation performance, using the soft attention mechanism to calculate correlations between other items and the last one. The experimental results show that two evaluation metrics of RESR-GNN have achieved ideal experimental results on different datasets. Empirical results on the Diginetica and Yoochoose1/64 datasets show that the proposed model consistently enhances performance, improving P@20 and MRR@20 by 3.15% and 5.51% on the former and by 1.53% and 2.36% on the latter. These gains across diverse datasets underscore the model's robustness and generalizability in effectively capturing user behavior patterns for session-based recommendation. In the following work, we will explore the integration of external information or adaptive mechanisms to enhance its robustness in short conversations and explore other implicit relationships in the conversation. At the same time, the existing experiments will also be further expanded.

References

1. Bahdanau, D., Cho, K., Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv preprint, arXiv:1409.0473, 2014.
2. Chen, D., Zhang, X., Wang, H., Zhang, W. TEAN: Timeliness Enhanced Attention Network for Session-Based Recommendation. Neurocomputing, 2020, 411, 229-238. https://doi.org/10.1016/j.neucom.2020.06.063
3. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., Salakhutdinov, R. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. arXiv preprint arXiv:1901.02860, 2019. https://doi.org/10.18653/v1/P19-1285
4. Eirinaki, M., Vazirgiannis, M., Kapogiannis, D. Web Path Recommendations Based on Page Ranking and Markov Models. Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management, 2005, 2-9. https://doi.org/10.1145/1097047.1097050
5. Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., Zhang, S.-H., Martin, R. R., Cheng, M.-M., Hu, S.-M. Attention Mechanisms in Computer Vision: A Survey. Computational Visual Media, 2022, 8(3), 331-368. https://doi.org/10.1007/s41095-022-0271-y
6. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D. Session-Based Recommendations with Recurrent Neural Networks. arXiv preprint, arXiv:1511.06939, 2015.
7. Hidasi, B., Quadrana, M., Karatzoglou, A., Tikk, D. Parallel Recurrent Neural Network Architectures for Feature-Rich Session-Based Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems, 2016, 241-248. https://doi.org/10.1145/2959100.2959167
8. Jannach, D., Ludewig, M. When Recurrent Neural Networks Meet the Neighborhood for Session-Based Recommendation. Proceedings of the Eleventh ACM Conference on Recommender Systems, 2017, 306-310. https://doi.org/10.1145/3109859.3109872
9. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J. Neural Attentive Session-Based Recommendation. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, 1419-1428. https://doi.org/10.1145/3132847.3132926
10. Li, R., Wu, Z., Jia, J., Bu, Y., Zhao, S., Meng, H. Towards Discriminative Representation Learning for Speech Emotion Recognition. Proceedings of IJCAI, 2019, 5060-5066. https://doi.org/10.24963/ijcai.2019/703
11. Liu, Q., Zeng, Y., Mokhosi, R., Zhang, H. STAMP: Short-Term Attention/Memory Priority Model for Session-Based Recommendation. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, 1831-1839. https://doi.org/10.1145/3219819.3219950
12. Ludewig, M., Jannach, D. Evaluation of Session-Based Recommendation Algorithms. User Modeling and User-Adapted Interaction, 2018, 28, 331-390. https://doi.org/10.1007/s11257-018-9209-6
13. Ma, R., Liu, N., Yuan, J., Yang, H., Zhang, J. CAEN: A Hierarchically Attentive Evolution Network for Item-Attribute-Change-Aware Recommendation in the Growing E-Commerce Environment. Proceedings of the 16th ACM Conference on Recommender Systems, 2022, 278-287. https://doi.org/10.1145/3523227.3546773
14. Mo, Y., Wu, Y., Yang, X., Liu, F., Liao, Y. Review of the State-of-the-Art Technologies of Semantic Segmentation Based on Deep Learning. Neurocomputing, 2022, 493, 626-646. https://doi.org/10.1016/j.neucom.2022.01.005

15. Popel, M., Tomkova, M., Tomek, J., Kaiser, Ł., Uszkoreit, J., Bojar, O., Žabokrtskỳ, Z. Transforming Machine Translation: A Deep Learning System Reaches News Translation Quality Comparable to Human Professionals. Nature Communications, 2020, 11(1), 4381. https://doi.org/10.1038/s41467-020-18073-9

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

17. Qiu, R., Yin, H., Huang, Z., Chen, T. GAG: Global Attributed Graph Neural Network for Streaming Session-Based Recommendation. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, 669-678. https://doi.org/10.1145/3397271.3401109

18. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P. Personalizing Session-Based Recommendations with Hierarchical Recurrent Neural Networks. Proceedings of the Eleventh ACM Conference on Recommender Systems, 2017, 130-137. https://doi.org/10.1145/3109859.3109896

19. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L. BPR: Bayesian Personalized Ranking from Implicit Feedback. Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, 2009, 452-461.

20. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L. Factorizing Personalized Markov Chains for Next-Basket Recommendation. Proceedings of the 19th International Conference on World Wide Web, 2010, 811-820. https://doi.org/10.1145/1772690.1772773

21. Ruocco, M., Skrede, O. S. L., Langseth, H. Inter-Session Modeling for Session-Based Recommendation. Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems, 2017, 24-31. https://doi.org/10.1145/3125486.3125491

22. Sarwar, B., Karypis, G., Konstan, J., Riedl, J. Item-Based Collaborative Filtering Recommendation Algorithms. Proceedings of the 10th International Conference on World Wide Web, 2001, 285-295. https://doi.org/10.1145/371920.372071

23. Shani, G., Heckerman, D., Brafman, R. I., Boutilier, C. An MDP-Based Recommender System. Journal of Machine Learning Research, 2005, 6(9).

24. Tan, Y. K., Xu, X., Liu, Y. Improved Recurrent Neural Networks for Session-Based Recommendations. Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, 2016, 17-22. https://doi.org/10.1145/2988450.2988452

25. Tavakol, M., Brefeld, U. Factored MDPs for Detecting Topics of User Sessions. Proceedings of the 8th ACM Conference on Recommender Systems, 2014, 33-40. https://doi.org/10.1145/2645710.2645739

26. Tuan, T. X., Phuong, T. M. 3D Convolutional Networks for Session-Based Recommendation with Content Features. Proceedings of the Eleventh ACM Conference on Recommender Systems, 2017, 138-146. https://doi.org/10.1145/3109859.3109900

27. Wang, W., Zhang, W., Liu, S., Liu, Q., Zhang, B., Lin, L., Zha, H. Beyond Clicks: Modeling Multi-Relational Item Graph for Session-Based Target Behavior Prediction. Proceedings of The Web Conference 2020, 2020, 3056-3062. https://doi.org/10.1145/3366423.3380077

28. Wang, Z., Zhao, H., Shi, C. Profiling the Design Space for Graph Neural Networks Based Collaborative Filtering. Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, 1109-1119. https://doi.org/10.1145/3488560.3498520

29. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T. Session-Based Recommendation with Graph Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33(1), 346-353. https://doi.org/10.1609/aaai.v33i01.3301346

30. Yu, F., Zhu, Y., Liu, Q., Wu, S., Wang, L., Tan, T. TAGNN: Target Attentive Graph Neural Networks for Session-Based Recommendation. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, 1921-1924. https://doi.org/10.1145/3397271.3401319

31. Yuan, F., He, X., Jiang, H., Guo, G., Xiong, J., Xu, Z., Xiong, Y. Future Data Helps Training: Modeling Future Contexts for Session-Based Recommendation. Proceedings of The Web Conference 2020, 2020, 303-313. https://doi.org/10.1145/3366423.3380116

32. Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., Lee, B. A Survey of Modern Deep Learning-Based Object Detection Models. Digital Signal Processing, 2022, 126, 103514. https://doi.org/10.1016/j.dsp.2022.103514

33. Zhang, L., Ji, Z., Xu, R., Zhao, T., Chen, Q., Wang, Y., Li, F. Saliency Detection Algorithm for Foggy Images Based on Deep Learning. Information Technology and Control, 2023, 52(3), 581-593. https://doi.org/10.5755/j01.itc.52.3.32258

34. Zhang, L., Wang, S., Liu, B. Deep Learning for Sentiment Analysis: A Survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018, 8(4), e1253. https://doi.org/10.1002/widm.1253
35. Zhang, M., Wu, S., Gao, M., Jiang, X., Xu, K., Wang, L. Personalized Graph Neural Networks with Attention Mechanism for Session-Aware Recommendation. IEEE Transactions on Knowledge and Data Engineering, 2020, 34(8), 3946-3957. https://doi.org/10.1109/TKDE.2020.3031329
36. Zhang, Z., Nasraoui, O. Efficient Hybrid Web Recommendations Based on Markov Clickstream Models and Implicit Search. IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), IEEE, 2007, 621-627. https://doi.org/10.1109/WI.2007.4427162
37. Zhao, K., Zheng, Y., Zhuang, T., Li, X., Zeng, X. Joint Learning of E-Commerce Search and Recommendation with a Unified Graph Neural Network. Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, 1461-1469. https://doi.org/10.1145/3488560.3498414