Enhanced Multi-Head Self-Attention Graph Neural Networks for Session-based Recommendation

Wenhao Pan, Kai Yang

***Abstract*—Session-based recommendation aims to predict following user behaviors based on short-term anonymous sessions. Although there are many types of existing recommendation models, it is still a challenging problem to dig out the profound relationship between users and items from short sessions. Therefore, complex transformation patterns and the dynamic evolution of user interests are considered in the session, and this paper proposes a new method, named Enhanced Multi-Head Self-Attention Graph Neural Networks for Session-based Recommendation (EMSAGNN). The model first converts sequence data into graph structure and feeds them into graph neural networks to dynamically learn the complex transition patterns and capture user preferences. Then, an enhanced multi-head self-attention network further learns sessions to capture rich hidden information in items. Finally, with the learned information, EMSAGNN calculates the probability scores of different items to recommend more suitable items for users. We conduct extensive experiments and comparisons on two public e-commerce datasets. The experimental results show that our proposed model is superior to the state-of-the-art methods.**

***Index Terms*—session-based recommendation, graph neural networks, enhanced multi-head self-attention, deep learning**

I. Introduction

With the improvement of information technology, users are often easily lost in massive information and cannot find target items. To solve these problems, personalized recommendation systems emerge. In recent years, users' private information has been protected more strictly. Meanwhile, many users are reluctant to leave factual information when browsing or searching. When users' personal information is not available, traditional recommendation methods cannot capture users' interest preferences based on a simple session sequence, which leads to inaccurate recommendation results. Therefore, session-based recommendation systems [1] are essential. Because users' preferences may be affected by multiple factors, such as age, gender, region, and season, user-based or content-based methods [2]–[4] cannot detect or capture dynamic changes in user interest. In contrast, session-based methods have natural advantages in learning the dependencies between items in a short-term session. Moreover, they can use the characteristics of the session to explore the current preference of users. Therefore, the main task of the session recommendation system is to extract information from sessions and learn users' interests. As a result, the session-based recommendation has gradually become a study hotspot in recommendation systems [11]–[13].

Most traditional session recommendation models are based on the Markov chain [8]–[10]. They assume that the user's subsequent action depends only on the previous one or several previous ones so that the Markov chain model can only capture short-term item information. However, it is impossible to learn the global dependencies of the entire session. With the rapid development of deep learning [29], neural network methods have been widely applied in various fields. Inspired by the successful cases of recurrent neural networks (RNNs) [25] in sequence modeling [14], [15], more and more researchers use RNNs to learn the session information. Compared to Markov chain methods, recurrent neural network models have natural advantages—they learn the entire session and improve performance. Among them, GRU4REC [11] and NARM [12] are two typical representatives. They improve the accuracy of recommendations from the perspective of the items and the users, taking advantage of the rich information in the items. At the same time, RNNs-based models also have apparent shortcomings. They consider complete session information but ignore the dynamic changes of user interests and are very time-consuming.

After Transformer model [16] is proposed by Google in 2017, the self-attention mechanism gradually becomes the mainstream method for session-based recommendations, e.g., SASRec [17] and STAMP [18]. They apply attentional mechanisms to learn global information about sessions and further improve the models' performance. With the progress of graph neural networks (GNNs) [19], [20], Wu *et al.* propose the SR-GNN [21] model that applies GNNs to session-based recommendations. The model builds a session sequence into graph structure data and considers the interrelationships between items, which provides a new direction for session-based recommendations.

Although the above methods have achieved exciting and advanced results, they still have some limitations. First of all, they do not consider the complex transition patterns between

Manuscript received June 12, 2021; revised November 8, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant No. 51874171.

Wenhao Pan is a master student of School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan, 114051, China (e-mail: pwh5253@163.com).

Kai Yang is an associate professor of School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan, 114051, China (Corresponding author, e-mail: asyangkai@126.com).

all items. The global and local preferences of users both play crucial roles in improving the model performance [8], [9]. Secondly, although some models consider complete sessions, the learning effect of session information is still poor. As the length of a sequence increases, it is easy to ignore previous items' information [11], [12]. Then, most models learn static preferences to predict the results and do not consider dynamic changes in users' interests [17], [18]. Finally, a session cannot be viewed as a simple time sequence, and the transition pattern between items should be more complicated [23].

We propose a new session-based recommendation model applying a graph neural network and an enhanced multi-head self-attention network to solve the above problems. First, we construct the session sequence as a session graph and use GNN to calculate the embedding vector of each item node. Secondly, to better capture the dynamic changes of users' interest preferences, we propose an enhanced multi-headed self-attention network to learn complex transition patterns and global dependencies in session. Third, we combine users' global and local preferences to predict their item needs accurately. Finally, we conducted a number of experiments on two benchmark e-commerce datasets. Experimental results show that this method performs well in session-based recommendation tasks.

II. Related Work

A. Traditional Session Recommendation Methods

Traditional session-based recommendation models include sequential pattern mining [5]–[7] and Markov chain methods [8]–[10]. They have natural advantages in taking the dependencies between users and items. Moreover, it is the most intuitive solution for session-based recommendations. For example, Rendel *et al.* propose the FPMC [8] model that combines matrix decomposition and first-order Markov chain methods. It calculates the probability of recommendation by learning users' global interest preferences and the transition relationship between items. Hidasi *et al.* make improvements to FPMC by adding a nonlinear transformation and propose the HRM [9] model. Since Markov chain methods assume that a recommended item depends on the most recent interacted items. Therefore, they can only capture point-based dependencies, ignoring the collective global dependencies between users and items. Instead, we construct a session graph to capture more complex transition patterns across inherent session information.

B. Deep Learning-based Session Recommendation Methods

In recent years, deep learning has achieved excellent computer vision, pattern recognition, and natural language processing. With the widespread application of neural network methods, more and more models have begun to use them in recommendation systems [24]. As an essential branch of recommendation systems, session-based recommendations are gradually becoming a deep learning technology research hotspot.

Inspired by the successful case of RNNs [25] in sequence modeling. [14], [15]. Hidasi *et al.* first propose the GRU4REC [11] model based on recurrent neural networks. It uses multi-layer gated recurrent units (GRUs) to learn the session sequence. The NARM [12] model propose by Li *et al.* stacks GRUs as an encoder to capture more transitional information of items and adds an attention mechanism. Compared with traditional methods, these models have achieved excellent results and apply RNNs in session-based recommendations. Inspired by Transformer [16], Kang *et al.* propose a SASRec [17] model, which applies a self-attention mechanism to model users' historical behavior sequences. Liu *et al.* propose the STAMP [18] model with a new attention mechanism, which can reduce the impact of time sequences by capturing users' global and local preferences for joint prediction. Compared with Markov chain and recurrent neural network models, the attention model is more efficient and easier to understand. It has been well applied in AFM [26] and DIN [27] models.

C. Graph Neural Network-based Session Recommendation Methods

Graph neural network is a generalized neural network based on graph structure that has emerged in recent years [19], [20]. Because of its unique computing ability, it has attracted the attention and research of many researchers. An early application of GNNs in session-based recommendation is the SR-GNN [21] model proposed by Wu *et al.* They use GNNs to process transitional relationships between users and items. The proposed model provides a new perspective for session recommendations. Xu *et al.* propose a graph contextual self-attention model (GC-SAN) [30], which can capture the long-term dependencies of items. Yu *et al.* propose a new target attention graph neural network (TAGNN) [22]. In TAGNN, target-aware attention adaptively activates users' different levels of interest towards various target items. The model considers the diversity of users' interests, improving its representation ability.

Compared with the above methods, we propose a new network model to conduct deep mining of sessions. In addition, we consider dynamic changes in users' interests. Therefore, the model can simultaneously learn users' global and local interest preferences and is unlimited to inherent behavior sequences.

III. Methods

In this section, we mainly introduce the construction of the EMSAGNN model. First, we apply graph neural networks to calculate the embedding vector of each item. Secondly, in a Transformer-like structure [16], the decoder layer is deleted, and the encoder layer is modified to be more suitable for session recommendations. Finally, build a prediction layer to calculate the predicted scores of different items for the next click. Next, we will introduce each part of the model in detail.

A. Problem Statement

The main task of session-based recommendation is to use the anonymous behavior sequence to predict the user's next interaction. Here, we define the formula of the problem. Let $V = \{v_1, v_2, \ldots, v_m\}$ represent a set of all unique items in session set. The session s can be represented by the list $s = [v_1, v_2, \ldots, v_n]$, where $v_i \in V$ represents the clicked items in session s. This problem can be expressed as predicting the user's next click v_{n+1} based on the session s. We input the

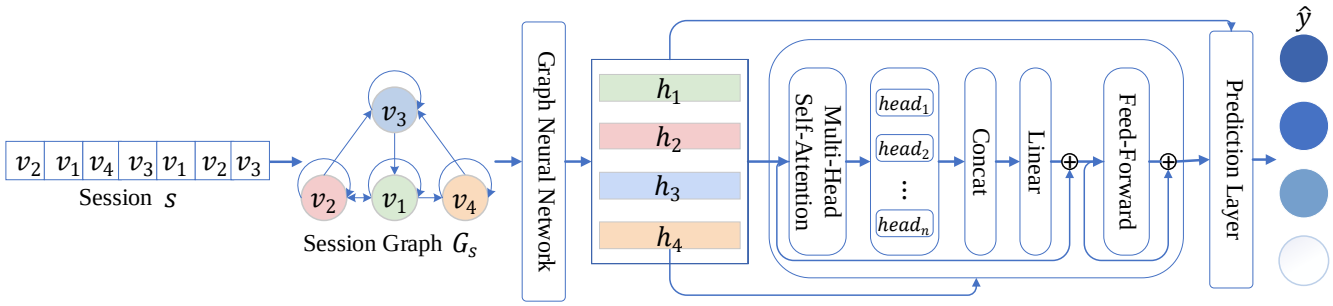

Fig. 1. The architecture of EMSAGNN. First, it uses graph neural networks to calculate the embedding vector of each item. Secondly, a new enhanced multi-head self-attention network is constructed. Finally, we use the prediction layer to calculate the predicted scores of different items for the next click.

session s into the model, and the output $\hat{y}$ is obtained, where $\hat{y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m\}$ is the predicted scores of all items and $\hat{y}_i$ is the probability score of item v_i. Thus, we construct and train a classifier that learns the probability score of each candidate in item set V. The top-N items in $\hat{y}$ are the candidate items recommended to users.

B. Constructing Session Graphs

We construct a session into a session graph $G_s = (V_s, E_s)$. In session graph G_s, the node set V_s contains all nodes in the graph, i.e., $v_i \in V_s$. The E_s represents the set of all directed edges in the graph, and each directed edge $(v_i, v_{i+1}) \in E_s$ means that the user clicks item v_i after v_{i+1}. Since there may be multiple repeatedly clicked items in the session, assigning a weight to each directed edge is necessary. Finally, we use the GNN to calculate the embedding vector of each item in the session graph.

C. Item Representation Learning

We input the session graph into the GNN to learn the embedding vectors of the nodes. Then, use the connection matrix of items to represent a session sequence. This paper uses the gated sequence graph neural network (GGNN) [28] to learn the embedding vectors of nodes, a variant of graph neural network. Formally, the update function of node v_i of session graph G_s is as follows:

$$h_i^1 = [e_i, 0]^T \tag{1}$$

$$a_i^t = A_{i:}[h_1^{t-1}, \cdots, h_n^{t-1}]^T H + b \tag{2}$$

$$z_i^t = \sigma(W_z a_i^t + U_z h_i^{t-1}) \tag{3}$$

$$r_i^t = \sigma(W_r a_i^t + U_r h_i^{t-1}) \tag{4}$$

$$h_i^t = \tanh(W_o a_i^t + U_o(r_i^t \odot h_i^{t-1})) \tag{5}$$

$$h_i^t = (1 - z_i^t) \odot h_i^{t-1} + z_i^t \odot h_i^t \tag{6}$$

Here we take session $s = [v_2, v_1, v_4, v_3, v_1, v_2, v_3]$ as an example, where h_i represents the embedding vector of node v_i. In Equation (1), e_i is the initial state of node v_i. When the dimension of the input feature e_i of the node is less than d, zeros are padded such that $h_i \in R^d$. In Equation (2), $A_{i:} \in R^{1\times 2n}$ selects two columns corresponding to node v_i from matrix A. The matrix $A \in R^{n\times 2n}$ determines the communication mode of the nodes in the graph. The series connection of A_{out} and A_{in} represents the weighted connection of the in-degree and the out-degree. The following $[h_1, \ldots, h_n]^T$ aggregates the features of all nodes at time t - 1 to form an n-dimensional vector. $a_i \in R^{2d}$ represents the interaction between a node and its adjacent node through a directed edge. The result here considers the two-way information transfer since two columns of A_{out} and A_{in} are taken in the matrix A. $H \in R^{d\times 2d}$ represents the control weight, and b represents the bias vector. Equations (3)-(6) are similar to the calculation process of GRU, z_i controls the forgotten information, and r_i controls the newly generated information. Then, $\sigma(\cdot)$ represents the Sigmoid function and $\odot$ represents the dot product operator. Finally, learn the embedding representation of all nodes in the session graph, and get the final node vector.

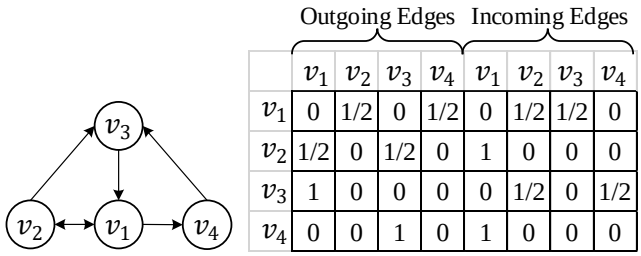

Fig. 2. Session graph and the connection matrix A

D. Enhanced Multi-Head Self-Attention Networks

Compared with traditional models, we fully consider the relevance of items and the dynamic evolution of user interests and introduce an enhanced multi-head self-attention network. Therefore, it can divide the vector space into multiple heads to learn more item information. The improved model can better learn the dependencies between items and extract transition patterns by splitting the sequence.

The self-attention mechanism maps the query and key-value pairs to the output and calculates it as a weighted sum of values, where the corresponding key and query determine the assigned weight. The matrices Q, K, and V are obtained by the linear transformation of the embedding matrix H, i.e., $H = [h_1, \ldots, h_n]$. In practice, the input of the self-attention network is the embedding matrix H or the previous encoder block output, which is calculated as follows [16]:

$$Q = linear(H) = HW^Q \tag{7}$$

$$K = linear(H) = HW^K \tag{8}$$

$$V = linear(H) = HW^V \tag{9}$$

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (10)$$

where W^Q, W^K, W^V represent the projection matrices. We calculate the inner product of each row pair of the matrices Q and K. To prevent too large inner products, they are divided by the square root of d_k. The matrix obtained by multiplying Q by the transpose of K represents the intensity of attention on items. Then, a softmax function calculates the attention coefficients of items. An item in the session adaptively assigns weights to other items through the self-attention mechanism. In this way, each item learns interdependency with other items.

In the previous section, we introduce how to calculate the output matrix through self-attention mechanisms. However, the multi-head self-attention network is constituted of many self-attention mechanisms. The multi-head self-attention network divides the embedding vector into multiple subspaces and can focus on information in different spaces and integrate them. Combining multiple self-attention blocks enhances the network's learning ability, which helps the network capture rich spatial information, which is calculated as follows [16]:

$$S = MH(Q,K,V) = Concat(head_1,\cdots,head_h)W^O \quad (11)$$

$$head_i = Attention(HW^Q, HW^K, HW^V) \quad (12)$$

where W^Q, W^K, W^V represent the parameter matrices, H represents the embedding matrix, and h represents the number of self-attention heads.

After that, we add a point-wise feed-forward network, which consists of two linear transformations and a ReLU activation function, and increases the nonlinearity of the model, which is calculated as follows [16]:

$$FFN(S) = ReLU(SW_1 + b_1)W_2 + b_2 \quad (13)$$

Following the multi-head self-attention and feed-forward networks, we apply residual connection and dropout technology to solve the overfitting problem of multi-layer networks, which is calculated as follows:

$$S' = S + Dropout(MH(S)) \quad (14)$$

$$F = S' + Dropout(FFN(S')) \quad (15)$$

where W_1, W_2 represent the weight matrices, b_1, b_2 represent the bias vectors.

For simplicity, we represent the above enhanced multi-head self-attention networks as a whole:

$$F = EMSAN(H) \quad (16)$$

After the first enhanced multi-head self-attention network block, the embedding vectors of all the previous items are synthesized. Finally, to further learn the transition patterns of the session, we stack the multiple network blocks, and the b-th block is calculated as:

$$F^b = EMSAN(F^{b-1}) \quad (17)$$

We observe in experiments that b = 1 obtains better performance compared to other values.

E. Prediction Layer

The prediction layer is constituted of dot product score calculation and softmax function. We use the embedding vector of session s to calculate the score $\hat{s}_i$ of each candidate item v_i:

$$\hat{s}_i = (F^b)^T h_i \quad (18)$$

where h_i represents the embedding vectors of all items in the session.

Then, we use the softmax function to $\hat{s}$ to get the probability score $\hat{y}$ of the item:

$$\hat{y} = softmax(\hat{s}) \quad (19)$$

where $\hat{s} \in R^m$ represents the predicted scores of all candidate items, and $\hat{y} \in R^m$ represents the probability of item to be clicked next.

For each session graph G_s, we apply the cross-entropy loss function:

$$L(\hat{y}) = -\sum_{i=1}^{m} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i) \quad (20)$$

where y represents the one-hot vector of a ground truth item.

In the end, we apply back-propagation through time algorithms to train the EMSAGNN.

IV. Experiments

This section mainly introduces datasets, baselines, and evaluation indicators. Then, the performance and parameters of the EMSAGNN model are studied.

A. Datasets

We choose two typical benchmark e-commerce datasets to evaluate the performance of the models.

- **Yoochoose** is the dataset of RecSys Challenge 2015 [34]. It records the click flows of an e-commerce website within 6 months.
- **Diginetica** is the dataset of CIKM Cup 2016 [35]. It contains transaction data suitable for session-based recommendations.

TABLE I
Statistic Details of the Datasets In the Experiments

Dataset	Yoochoose	Diginetica
# clicks	557,248	982,961
# train sessions	369,859	719,470
# test sessions	55,898	60,858
# items	16,766	43,097
Average length	6.16	5.12

For a fair comparison, we follow the preprocessing method proposed by the predecessors and filter out the sessions with length 1 and the items with fewer than 5 clicks in the two datasets [21], [22]. After data preprocessing, the Yoochoose dataset has 7,985,580 sessions and 37,483 items, and the Diginetica dataset has 204,771 sessions and 43,097 items. Then, we divide a session of length n into n - 1 training sequences of lengths 2 to n to expand the dataset, and the last item is used as the label item. The Yoochoose dataset is too large, and we choose the most recent parts 1/64 of the training data.

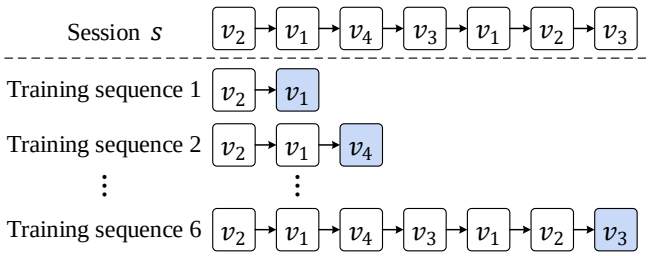

Fig. 3. The preprocessing of training sequences

B. Baselines

We chose the following representative methods for performance comparison to prove the effectiveness of the EMSAGNN model.

- **POP** recommends items with many interactions with users, which is a standard baseline method in Top-N recommendations.
- **S-POP** recommends the most popular items in the session for users and takes the top-N frequent items as the recommended targets.
- **BPR-MF** [31] is a commonly used matrix factorization algorithm, which uses the BPR objective function to calculate the pairwise ranking loss.
- **FPMC** [8] applies the combination of matrix factorization and Markov chain to capture the dependencies between items.
- **Item-KNN** [32] is a classic collaborative filtering algorithm that recommends products by calculating the cosine similarity between items.
- **GRU4REC** [11] uses recurrent neural network modeling to predict the next possible interactive item through item dependencies.
- **NARM** [12] applies attention mechanisms and recurrent neural networks to capture users' primary purposes and sequence behaviors.
- **STAMP** [18] splits user sessions into global and local interests for joint prediction.
- **RepeatNet** [33] adds the probability calculation of repeated recommendations and in a very clever way.
- **SR-GNN** [21] applies graph neural networks to capture the dependencies between users and items.
- **GC-SAN** [30] dynamically constructs the graph structure data and captures rich local information through graph neural networks.
- **TAGNN** [22] introduces target-aware attention adaptively activates users' different interest levels towards various target items.

C. Evaluation Metrics

We choose the two most popular evaluation metrics to estimate the predicted performance of all models.

HR@20 (Hit Rate): This metric takes into account the accuracy of predicted items. When the ground truth is 1, the hit ratio is equal to recall, which is the proportion of correctly predicted items.

$$HR@20 = \frac{n_{hit}}{N} \tag{21}$$

where N represents the total of test sets, and n_{hit} represents the number of correct items in the top 20 ranking lists.

MRR@20 (Mean Reciprocal Rank): This metric takes into account the ranks of predicted items. When the ranking exceeds 20, MRR@20 is set to zero. Therefore, the higher the MRR@20 value, the correct prediction is closer to the top ranking list.

$$MRR@20 = \frac{1}{N}\sum_{v_i \in s_{test}} \frac{1}{Rank(v_i)} \tag{22}$$

where v_i represents the prediction items, and s_{test} represents the test sets.

D. Experiment Setting

We have conducted many experiments on the EMSAGNN model and used the Adam optimizer to optimize these parameters to obtain optimized performance. The initial learning rate is set to 0.0005 and decays by 0.1 after every 3 epochs. At the same time, the L2 regularization parameter is set to 10^{-5} to alleviate overfitting.

E. Comparison with Baseline Methods

We compared EMSAGNN with representative baseline methods to evaluate the performance of the model. All methods use two evaluation indicators for evaluation, and the experimental results are shown in Table II. Experimental results show that the performance of this model is better than other baseline methods, which verifies the advantages of the model.

For traditional non-personalized recommendation models, such as POP and S-POP, they cannot mine the in-depth information in the session because the modeling method is too simple. Item-KNN outperforms the BPR-MF and FPMC models. The main reason is that Item-KNN is based on the similarity matrix for modeling, which takes advantage of the similarity between the items. However, Item-KNN ignores the order relationship between items, and the prediction results are not accurate. With the development of deep learning technology, the prediction accuracy of the model has been dramatically improved. The GRU4REC model is the first method to use recurrent neural networks. It takes advantage of the natural advantages of recurrent neural networks to process sequence data and achieve proud performance. This shows that deep learning technology has a great potential for session-based recommendations. The NARM and STAMP models use an attention mechanism, which effectively improves the models' performance.

TABLE II
PERFORMANCE COMPARISON OF EMSAGNN WITH BASELINE METHODS

Methods	Yoochoose		Diginetica	
	HR@20	MRR@20	HR@20	MRR@20
POP	6.71	1.65	0.89	0.20
S-POP	30.44	18.35	21.06	13.68
BPR-MF	31.31	12.08	5.24	1.98
FPMC	45.62	15.01	26.53	6.95
Item-KNN	51.60	21.81	35.75	11.57
GRU4REC	60.64	22.89	29.45	8.33
NARM	68.32	28.63	49.70	16.17
STAMP	68.74	29.67	45.64	14.32
RepeatNet	70.71	31.03	47.79	17.66
SR-GNN	70.57	30.94	50.73	17.59
GC-SAN	70.66	30.04	50.90	17.63
TAGNN	71.02	31.12	51.31	18.03
EMSAGNN	**71.43**	**31.88**	**51.48**	**18.04**

RepeatNet considers the phenomenon of repeated purchases and improves the performance through repeated recommendations, which is better than the previous methods. SR-GNN constructs the session as a session graph and considers the translation patterns between items. The GC-SAN model applies a self-attention behind the GNN to learn the long-term dependence of the session. However, it is not easy to obtain in-depth local information in the session. The TAGNN model proposes a target perception module, and overall performance is better than the SR-GNN and GC-SAN. The comprehensive performance of the method GNN-based is better than that RNN-based.

In summary, we propose a model that shows the best performance under two evaluation metrics. Furthermore, these results demonstrate the efficiency of the EMSAGNN model.

F. Ablation Studies

This section conducts further research and analysis on the model's architecture to prove its validity.

We compare the model in this paper with no graph neural network (EMSAGNN-baseEmbedding) and basic attention (EMSAGNN-baseAttention) models. From Table III, the performance of EMSAGNN is better than EMSAGNN-baseEmbedding and EMSAGNN-baseAttention models. This further verified the complementarity of the graph neural network and enhanced multi-head self-attention network, which plays an essential role in improving the model's prediction performance.

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT ARCHITECTURES

Methods	Yoochoose		Diginetica	
	HR@20	MRR@20	HR@20	MRR@20
EMSAGNN-baseEmbedding	70.56	30.31	50.32	16.25
EMSAGNN-baseAttention	69.08	29.52	48.85	16.20
EMSAGNN	**71.43**	**31.88**	**51.48**	**18.04**

G. Hyperparameter Studies

The values of hyperparameters play important roles in improving model performance. However, for different models, the values of hyperparameters are also different. This section selects some common hyperparameters to discuss and observe the influence on model performance.

The model uses an enhanced multi-head self-attention network to learn and aggregate information from multiple spaces, which helps the network to capture richer session information. We change the number of attention heads from 1 to 12 and observe the model performance. In Fig. 4, the increase in attention heads does not improve the model's performance. However, too many attention heads limit the expressive ability of attention. The experimental results show that the best number of heads in the Yoochoose dataset is 4, and the Diginetica dataset is 10. When the number of attention heads is bigger or smaller than the optimal value, it will affect the model's performance.

The enhanced multi-head self-attention network can learn the deep dependencies of items to get the user's dynamic interest preferences. We choose 1-6 layers of network for experimental testing. In Fig. 5, the best number of network layers in both datasets is 1. As the number of network layers increases, the performance of the model will also decrease.

We also try to select the appropriate embedding size for specific tasks. We choose the embedding size from {70, 80, 90, 100, 110, 120} for the Yoochoose dataset, and {30, 40, 50, 60, 70, 80} for the Diginetica dataset. Generally, there is a dimension with the best effect in the range. In Fig. 6, the optimal embedding size of the Yoochoose dataset is 100, and that of the Diginetica dataset is 60. The embedding size limits learning ability when too small, and is easy to overfit when too large.

In Fig. 7, we consider the impact of the number of layers of feed-forward networks on the model's performance. We choose 1-6 layers of feed-forward networks to test the model. It is observed that the optimal number of network layers is 4 on both datasets. Too many or too few network layers will affect the learning ability of the model.

V. CONCLUSIONS

In situations where user information and long-term history are not available, session-based recommendations are essential. This paper proposes a new model combining enhanced multi-head self-attention networks and graph neural networks. It considers the complex transition patterns between items and the dynamic evolution of user interests. The prediction is jointly made through users' global and local interest preferences. The experimental results show that our proposed model achieves state-of-the-art results compared with representative methods on two public e-commerce datasets.

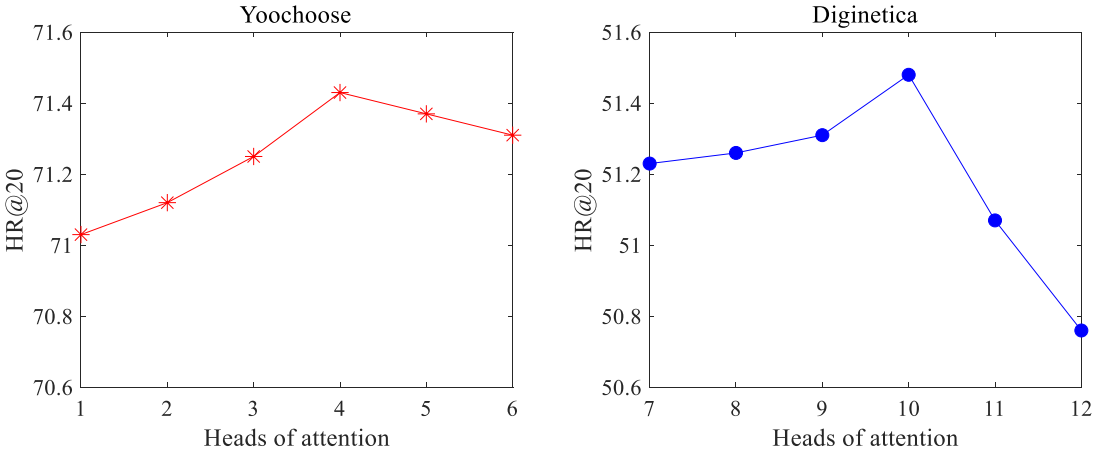

Fig. 4. The performance under different numbers of attention heads

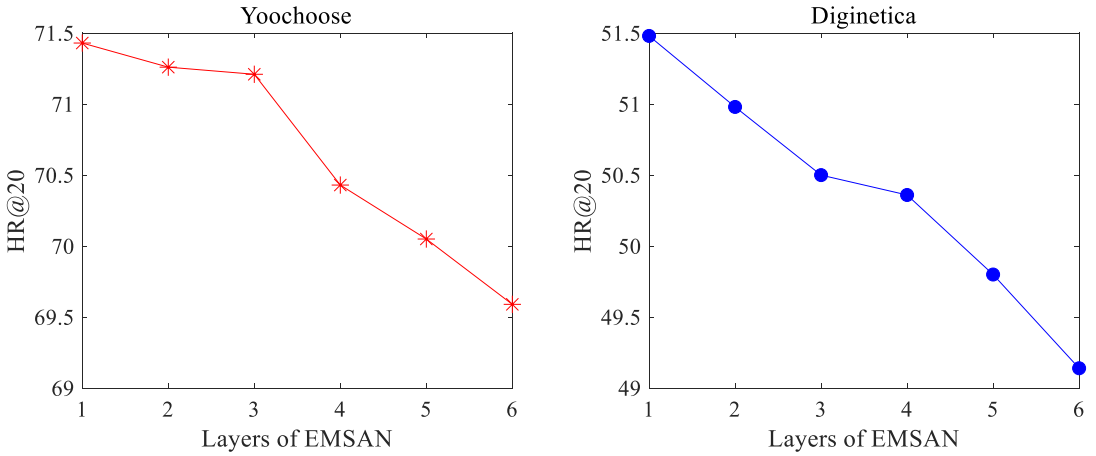

Fig. 5. The performance under different numbers of layers of EMSAN

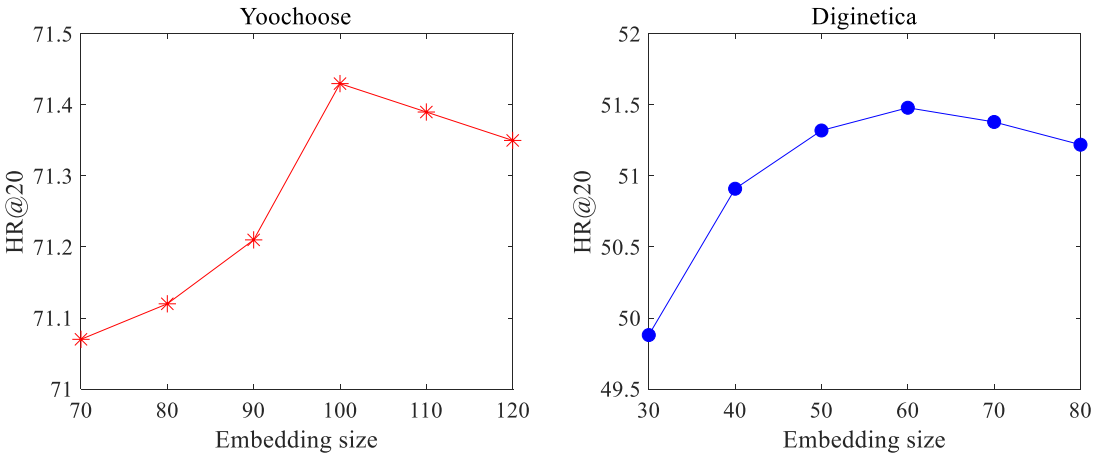

Fig. 6. The performance under different embedding sizes

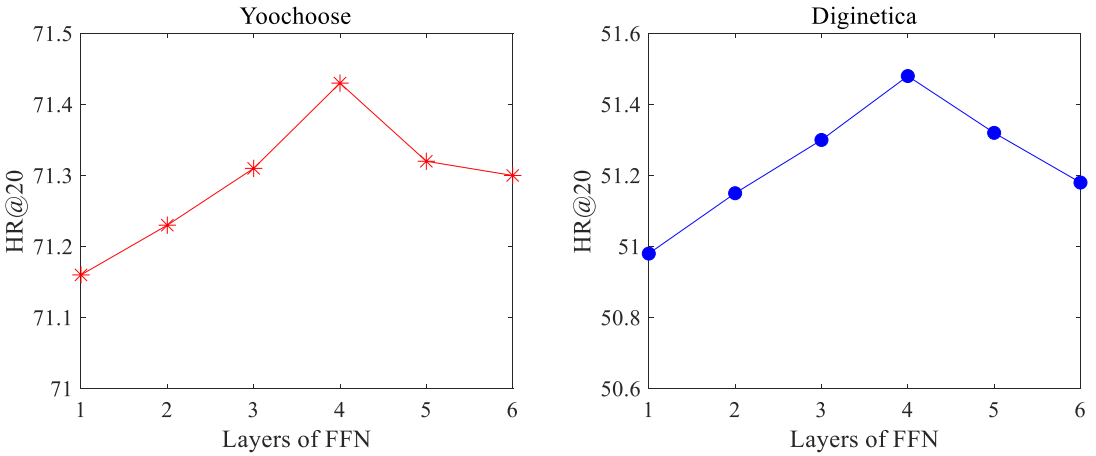

Fig. 7. The performance under different numbers of layers of feed-forward network

REFERENCES

[1] J. B. Schafer, J. A. Konstan and J. Riedl, "Recommender systems in e-commerce," in *Proc. ACM Conf. Electronic Commerce*, pp. 158–166, 1999.

[2] Y. Hu, Y. Koren and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE International Conference on Data Mining*, pp. 263–272, 2008.

[3] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 426–434, 2008.

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

[5] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994.

[6] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. 11th International Conference on Data Engineering*, pp. 3-14, 1995.

[7] J. Pei, J. W. Han, B. Mortazavi-Asl and H. Pinto, "PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proc. International Conference of Data Engineering*, pp. 215-224, 2001.

[8] S. Rendle, C. Freudenthaler and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proc. 19th International Conference on World Wide Web*, pp. 811–820, 2010.

[9] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan and X. Cheng, "Learning hierarchical representation model for next basket recommendation," in *Proc. 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 403-412, 2015.

[10] R. He and J. McAuley, "Fusing similarity models with markov chains for sparse sequential recommendation," in *Proc. IEEE 16th International Conference on Data Mining*, pp. 191-200, 2016.

[11] B. Hidasi, A. Karatzoglou, L. Baltrunas and D. Tikk, "Session-based recommendations with recurrent neural networks," in *Proc. International Conference on Learning Representations*, 2015.

[12] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian and J. Ma, "Neural attentive session-based recommendation," in *Proc. 26th ACM on Conference Information and Knowledge Management*, pp. 1419–1428, 2017.

[13] Y. K. Tan, X. Xu and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *Proc. 1st Workshop on Deep Learning for Recommender Systems*, pp. 17–22, 2016.

[14] Zahra Berradi, Mohamed Lazaar, Hicham Omara and Oussama Mahboub, "Effect of architecture in recurrent neural network applied on the prediction of stock price," *IAENG International Journal of Computer Science*, vol. 47, no.3, pp436-441, 2020.

[15] H. Sak, A. Senior, K. Rao and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," arXiv preprint arXiv:1507.06947, 2015.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, "Attention is all you need," in *Proc. Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[17] W. C. Kang and J. Mcauley, "Self-Attentive Sequential Recommendation," *in Proc. 2018 IEEE International Conference on Data Mining*, pp. 198-206, 2018.

[18] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: Short-term attention/memory priority model for session-based recommendation," in *Proc. 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1831–1839, 2018.

[19] S. Wu, W. Zhang, F. Sun and B. Cui, "Graph neural networks in recommender systems: a survey," arXiv preprint arXiv:2011.02260, 2020.

[20] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61-80, 2008.

[21] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie and T. Tan, "Session-based recommendation with graph neural netwyooorks," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 346-353, 2019.

[22] F. Yu, Y. Zhu, Q. Liu, S. Wu, L. Wang and T. Tan, "TAGNN: Target attentive graph neural networks for session-based recommendation," in *Proc. 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1921–1924, 2020.

[23] R. Qiu, J. Li, Z. Huang and H. Yin, "Rethinking the item order in session-based recommendation with graph neural networks," in *Proc. 28th ACM International Conference on Information and Knowledge Management*, pp. 579-588, 2019.

[24] M. R. Minar and J. Naher, "Recent advances in deep learning: an overview," arXiv preprint arXiv:1807.08169, 2018.

[25] W. Zaremba, I. Sutskever and O. Vinyals, "Recurrent neural network regularization," arXiv preprint arXiv:1409.2329, 2014.

[26] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, T. Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," arXiv preprint arXiv:1708.04617, 2017.

[27] G. Zhou, C. Song, X. Zhu, X. MA, Y. Yan et al., "Deep interest network for click-through rate prediction," in *Proc. 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1059-1068, 2018.

[28] Y. Li, R. Zemel, M. Brockschmidt and D. Tarlow, "Gated graph sequence neural networks," in *Proc. International Conference on Learning Representations*, 2016.

[29] Y. LeCun, Y. Bengio and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no.7553, pp. 436–444, 2015.

[30] C. Xu, P. Zhao, Y. Liu, V.S. Sheng, J. Xu, F. Zhuang, J. Fang and X. Zhou, "Graph contextualized self-attention network for session-based recommendation," in *Proc. 28th International Joint Conference on Artificial Intelligence*, vol. 19, pp. 3940–3946, 2019.

[31] S. Rendle, C. Freudenthaler, Z. Gantner and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, 2009.

[32] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th International World Wide Web Conference*, pp. 285–295, 2001.

[33] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma and M. D. Rijke, "Repeatnet: a repeat aware neural recommendation machine for session-based recommendation," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 4806-4813, 2019.

[34] Yoochoose dataset, RecSys Challenge 2015, Available: https://2015.recsyschallenge.com/challenge.html, 2015.

[35] Diginetica dataset, CIKM cup 2016, Available: http://cikm2016.cs.iupui.edu/cikm-cup, 2016