Information Sciences 624 (2023) 324–343

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Dynamic global structure enhanced multi-channel graph neural network for session-based recommendation

Xiaofei Zhu [a,*], Gu Tang [a], Pengfei Wang [b], Chenliang Li [c], Jiafeng Guo [d], Stefan Dietze [e,f]

[a] College of Computer Science and Engineering, Chongqing University of Technology, Chongqing 400054, China
[b] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
[c] School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China
[d] Institute of Computing Technology, Chinese Academy of Science, Beijing 100190, China
[e] Knowledge Technologies for the Social Sciences, Leibniz Institute for the Social Sciences, Cologne 50667, Germany
[f] Institute of Computer Science, Heinrich-Heine-University Düsseldorf, Düsseldorf 40225, Germany

ARTICLE INFO

Article history:
Received 3 March 2022
Received in revised form 6 August 2022
Accepted 3 October 2022
Available online 17 October 2022

Keywords:
Recommendation system
Session-based recommendation
Graph neural network
Behavior modeling
Attention model
Representation learning

ABSTRACT

Session-based recommendation is a challenging task, which aims at making recommendation for anonymous users based on in-session data, i.e. short-term interaction data. Most session-based recommendation methods only model user's preferences with the current session sequence, which ignore rich information from a global perspective. Meanwhile, previous works usually apply GNN to capture the transformation relationship between items, however the graph used in GNN is built through a static mode, which may introduce noise to the graph structure if user's preferences shift. In this paper, we propose a novel method called Dynamic Global Structure Enhanced Multi-channel Graph Neural Network (DGS-MGNN) to learn accurate representations of items from multiple perspectives. In DGS-MGNN, we propose a novel GNN model named Multi-channel Graph Neural Network to generate the local, global and consensus graphs dynamically and learn more informative representations of items based on the corresponding graph. Meanwhile, in order to reduce the noise information within sessions, we utilize the graph structure to assist the attention mechanism to filter noisy information within each session, so as to generate an accurate intention representation for the user. Finally, combined with a repeat and explore module, a more accurate prediction probability distribution is generated. We conduct extensive experiments on three widely used datasets, and the results demonstrate that DGS-MGNN is consistently superior to the state-of-the-art baseline models.

1. Introduction

Session-based recommendation (SBR) aims at predicting user's preferences for recommending future items based on anonymous behavior sequences, and plays a critical role in streaming media platforms or e-commerce platforms such as Tiktok, YouTube and Taobao. Within a session, it usually only has short-term historical interactions of a user (*e g.* users that are new or not logged in), and conventional recommendation algorithms (e.g., Collaborative Filtering [21,7,3] or Markov Chains

* Corresponding author.
E-mail addresses: zxf@cqut.edu.cn (X. Zhu), gutang@2020.cqut.edu.cn (G. Tang), wangpengfei@bupt.edu.cn (P. Wang), cllee@whu.edu.cn (C. Li), guojiafeng@ict.ac.cn (J. Guo), stefan.dietze@gesis.org (S. Dietze).

https://doi.org/10.1016/j.ins.2022.10.025
0020-0255/© 2022 Published by Elsevier Inc.

[20,23]) which heavily rely on user's long-term history interactions or user profiles would perform poorly when applied on in-session data.

In recent years, some research efforts [8,11,13] attempt to leverage Recurrent Neural Networks (RNNs) and attention mechanism [36] for the task of SBR and achieve encouraging results. For example, GRU4REC [8] addresses the task of SBR by utilizing a recurrent neural network to exploit the temporal shift of user behavior. NARM [11] further incorporates the attention mechanism [36] to model the sequential behavior of users, which captures both user's sequential behavior and her main interest in the current session. STAMP [13] utilizes a simple multilayer perception and an attentive network to model user's general interests from the long-term memory of a session and her current interests from the short-term memory of the last-click. Since these works mainly focus on modeling sequential transitions between consecutive items, the complex transitions between distant items in the session are largely ignored. To deal with this issue, graph neural networks (GNNs) based methods [33,37] are proposed to model the complex transition relationships. SR-GNN [33] first constructs session graph using items in historical session sequences, and then applies the gated graph neural network (GGNN) [12] to model the complex transitions among distant items. TAGNN [37] extends SR-GNN by proposing a target attentive network, which can explore the relevance of historical actions given a target item, to adaptively activate users' diverse interests in sessions. DSAN [38] introduces a dual sparse attention network which applies an adaptively sparse transformation function to alleviate the effect of noise items.

These studies mentioned above model user preference only based on the current session while useful item-transition patterns from other sessions are overlooked, i.e., collaborative information within neighborhood sessions that have been generated by other users. Very recently, there are few research efforts [29,32] have been devoted to model neighborhood sessions to improve the representation of the current session. CSRM [29] consists of two parallel modules, i.e., Inner Memory Encoder (IME) and Outer Memory Encoder (OME). The former leverages RNNs to model the current session, while the latter exploits neighborhood sessions to better capture the intent of the current session. CSRM models sessions based on a session-level granularity and extracts collaborative information via similarities between the latest m sessions and the current session, which may deteriorate the performance by involving irrelevant information of other sessions [31]. GCE-GNN [32] employs a more fine-grained manner which exploits the item transitions from other sessions to learn representations of items. Specifically, GCE-GNN models the user preference of the current session by exploiting both session-level and global-level pairwise item transitions.

Although the above research efforts have achieved encouraging performance by modeling item transition patterns, they still face some limitations: (1) the item transition information usually contain noisy information caused by users' accidental or wrong clicks, which may be injected into the model information propagation process and inevitably lead to inferior preference representation learning; (2) some previous efforts aim to explore neighborhood sessions for enhancing the representation of the current session, the consensus signals between both the local session graph and the global session graph is not well maintained; (3) the position information of items within a session is mostly modeled in an absolute manner, which may not be suitable for the task of SBR.

To address the above issues, we propose a novel model named Dynamic Global Structure Enhanced Multi-channel Graph Neural Network (DGS-MGNN). The main idea of our solution is to leverage the item similarity patterns rather than the item transition patterns to capture high-quality relationships between items, where negative impact caused by users' accidental or wrong clicks will be largely alleviated. In addition, three kinds of graphs (i.e., session graph, global graph, and consensus graph) are constructed in a dynamic manner in order to effectively capture item representation from different perspectives, where the consensus graph is incorporated to maintain consensus between both the session graph and the global graph. At last, we enhance the position embedding of each item within a session by integrating the session length information as well as capturing the inherent topological position structure of the session. DGS-MGNN mainly consists of four modules, including Dynamic Global Structure Enhanced Multi-Channel GNN, Graph Position Encoder (GPE), Repeat Module, and Explore Module. The first module is comprised of two sub-modules, i.e., Dynamic Global Neighbor Attention (DGNA) and Multi-Channel Graph Neural Network (MC-GNN). DGNA is proposed to learn a global representation of each item in the current session from a global perspective, i.e., capturing neighboring items from the entire item space which consists of items from all sessions. And MC-GNN is designed to fuse information from different perspectives, i.e., local representation, global representation, and consensus representation. Note that the three types of representations will be dynamically updated by a multi-layer MC-GNN. As conventional absolute position embedding methods [15,9] overlook the personalized position relationship within a session, it would lead to inferior performance. To address this issue, we further propose the module *Graph Position Encoder (GPE)* to generate a graph position embedding for each item in the current session by capturing both the session length information and the inherent topological position structure of the current session. The *Repeat Module* attempts to generate the probability distribution of items that appear in the current session based on representations from three perspectives (i.e., local, global and consensus) together with the graph position embedding information. The *Explore Module* consists of two sub-modules, i.e., *Graph-Enhanced Attention Network (GEA)* and *Explore Prediction*. As conventional attention mechanism would assign attention weights to the accidental or wrong clicks of users, and bring noise to the representation of session. To alleviate this problem, *GEA* is designed to filter out noisy items within a session and obtain the long-term structure representation of the current session. *Explore Prediction* first utilizes the Bi-GRU to get the sequential representation of the current session. Then it combines both the long-term structure representation and the sequential representation to generate the probability distribution of items that didn't appear in the current session. Finally, we employ the gate mechanism to integrate the probability distribution of candidate items generated by Repeat Module and Explore Module in order to generate the final predict probability distribution.

We conduct extensive experiments on three widely used datasets (i.e., Diginetica, Yoochoose, and Retailrocket) and the results show that our method significantly outperforms other state-of-the-art baseline methods in terms of both P@20 and MRR@20. We conduct further experiments to analyze each component of DGS-MGNN in depth so as to explore how each component affects the performance of session-based recommendation. At last, we also conduct experiments to investigate the computational complexity of our method. We summarize the main contributions as follows.

- We propose a novel Dynamic Global Structure Enhanced Multi-Channel Graph Neural Network (DGS-MGNN) which dynamically models information from three different perspectives, including the global representation, the local representation, and the consensus representation.
- We develop a novel position embedding method, i.e., the Graph Position Encoder (GPE), by further capturing the session length information as well as the inherent topological position structure within a session.
- We design the Graph-Enhanced Attention Network (GEA) to filter out noisy items within a session and obtain the long-term structure representation of the current session.
- We conduct extensive experiments on three widely used datasets, and the results demonstrate that our proposed approach DGS-MGNN is consistently superior to the state-of-the-art baseline methods.

The rest of the paper is organized as follows. Section 2 gives a brief description of the related work. We introduce our proposed approach DGS-MGNN in Section 3, and discuss the experimental results in Section 4. In Section 5, we conclude the paper.

2. Related work

We review the related work of session-based recommendation from three aspects: Traditional recommendation methods, RNN and attention based methods, and GNN-based methods.

Traditional recommendation methods. Traditional methods based on Collaborative Filtering (CF) [21,7,3] are widely used in recommendation. Since CF mainly applies matrix factorization on the user-item interaction matrix to obtain general preferences of users, it is unable to effectively capture the user's interest shift. Some works also investigate the chain-based [20,23] models to explore the sequential transaction data for the task of SBR. FPMC [20] extends matrix factorization by incorporating a first-order Markov chain to model both sequential behavior and long-term user preference. One major limitation of FPMC is that it linearly combines all components which indicates that it makes a strong independent assumption (i.e., each component affects user's next interaction independently) among multiple factors [30]. To deal with this issue, Wang et al. [30] propose a hierarchical representation model (HRM), which combines user's representation and user's behavior sequence information hierarchically to improve recommendation performance.

RNN and Attention based methods. In recent years, neural network based methods have achieved encouraging results in SBR. Hidasi et al. [8] first propose to apply a RNN-based model GRU4REC to the session-based recommendation setting by taking temporal shift of user behavior into consideration [26]. NARM [11] extends GRU4REC by combining GRU and attention mechanism [36] to model the sequential behavior of users. It leverages a hybrid encoder to model a user's sequential behavior and her main interest in the current session. Similar to NARM, STAMP [13] attempts to address the user interests drift issue by capturing both a user's general interests from the long-term memory of a session and her current interests from the short-term memory of the last-click.

Inspired by the great success of *Transformer* [27,5], SASRec [9] proposes to build a self-attention based sequential recommendation model, which tends to model long-range dependencies on dense data and focus on more recent actions on sparse data. Chen et al. [2] utilize a co-attention network to model interactions between actions in a user's long-term and short-term interaction histories and generate co-dependent representations of their long-term and short-term interests. Pan et al. [16] explore a user's long-term and his current interest to make recommendations. To accurately capture a user's long-term preference, they propose an important extraction module (IEM) to extract the importance of each item in the current session based on a modified self-attention mechanism. Ren et al. [19] propose a novel encoder-decoder framework RepeatNet which incorporates a repeat-explore mechanism in a regular neural recommendation approach. Yuan et al. [38] propose a dual sparse attention network for the task of SBR, which leverages a self-attention network and a vanilla attention network to generate the target item embedding and the entire session representation respectively. Wang et al. [29] propose a collaborative session-based recommendation machine, named CSRM, by exploring neighborhood information. It consists of two components, i.e., an Inner Memory Encoder (IME) and an Outer Memory Encoder (OME), where the former captures a user's own information in the current session and the latter models collaborative information from neighborhood sessions. A fusion gating mechanism is leveraged to combine the representations produced by IME and OME. Luo et al. [14] learn the session representation by designing a collaborative self-attention network (CoSAN). They inject the collaborative information in neighborhood sessions into the representation of the item in the current session, and obtain a dynamic item representation. RCNN-SR [39] combines GRU and attention mechanism to learn the user's general interest, meanwhile it exploits convolutional operation with horizontal filter and vertical filter to search for user's current interest and dynamic interest.

Graph neural network based methods. Very recently, a series of graph neural network (GNN) based models have been used for SBR, which can effectively capture the complex transitional patterns within sessions. For instance, Wu et al. [33] propose SR-GNN, which applies the gated graph neural network (GGNN) [12] to model the complex transitions among distant items. Inspired by SR-GNN, a number of variants [35,17,15,37,32] have been proposed. Xu et al. [35] propose graph contextual self-attention model based on graph neural network (GC-SAN). GC-SAN strengthens self-attention network with graph neural network (GNN), which can well capture the complementary strengths of GNN and self-attention. FGNN [17] considers the order relationship between items and casts the recommendation task as a graph classification task. SGNN–HN [15] constructs a star graph to enrich the connection between items and incorporates the highway networks [25] to alleviate the over smoothing problem that caused by deep GNN. Yu et al. [37] propose TAGNN which jointly models user interests given a certain target item as well as complex item transitions in sessions to make recommendation. CIAM [4] exploits different types of graph (i.e., local graph and global graph) and combine superposition and a weighted graph convolutional network to obtain users' general and temporal interests. It relies on utilizing many common features, such as category features, as auxiliary information. GCE-GNN [32] exploits the pairwise item-transition information from two levels of graph models, i.e., session graph and global graph. It employs a GNN model on session graph to learn session-level item embeddings within the current session and leverages a session-aware attention mechanism on global graph to learn global-level item embeddings over all sessions. COTREC [34] exploits the session-based graph to augment two views on both the internal and external connectivities of sessions, and combines self-supervised learning with co-training based on graph neural network to enhance session-based recommendation.

Our work differs with the above state-of-the-art approaches in threefold. First, only few efforts have been devoted to handle the noise issue in the graph structure caused by accidental or wrong clicks of users, and it is the research gap we attempt to bridge in this work. Specifically, our proposed method alleviates the influence of noise items by exploiting the structure of session graph to identify noise items within the session and then adjusting the attention mechanism to eliminate the influence of them. Second, we propose a novel multi-channel graph neural network (MC-GNN) to capture rich information from different perspectives, i.e., global, local, and consensus perspectives. Third, we introduce a novel position embedding methods by integrating the session length information as well as the inherent topological position structure information.

3. Approach

In this section, we first define the session-based recommendation task, and then introduce our proposed method Dynamic Global Structure Enhanced Multi-Channel Graph Neural Network (DGS-MGNN) in detail. The DGS-MGNN framework is demonstrated in Fig. 1, and it mainly consists of four modules, i.e., *Dynamic Global Structure Enhanced Multi-Channel GNN*, *Graph Position Encoder*, *Repeat Module*, and *Explore Module*.

3.1. Problem definition

Let $V = \{v_1, v_2, \cdots, v_{|V|}\}$ denote the unique items appearing in all sessions, where $|V|$ is the number of all unique items. Denote $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{|V|})$ as the corresponding embeddings for all items in V, where $\mathbf{h}_j$ $(j \in \{1, \cdots, |V|\})$ is the embedding of the j-th item v_j. Each session (*i.e*, items clicked by an anonymous user) can be represented as $S = (v^s_1, v^s_2, \cdots, v^s_n)$, which is an item sequence in chronological order, where v^s_i denotes the i-th item clicked by the corresponding user of the session S and n is the length of the session S. Let $\mathbf{H}_s = (\mathbf{h}^s_1, \mathbf{h}^s_2, \cdots, \mathbf{h}^s_n)$ denote the corresponding embedding of the session S, the goal of session-based recommendation is to recommend the next item that a user is most likely to click based on the current session.

3.2. Dynamic global structure-enhanced multi-channel GNN

In this subsection, we aim to learn a powerful representation for each item in the session S by proposing a *Dynamic Global Structure Enhanced Multi-Channel GNN module*. It consists of two sub-modules, named *Dynamic Global Neighbor Attention (DGNA)* and *Multi-Channel Graph Neural Network (MC-GNN)*, which are designed to enhance the representation of items from different perspectives.

Dynamic Global Neighbor Attention (DGNA). DGNA aims at dynamically learning a global representation of each item v^s_i in the current session S from a global perspective. To this end, we first obtain the K nearest neighbors $\mathcal{N}^i = \{n^i_1, n^i_2, \cdots, n^i_K\}$ of v^s_i from V based on a distance metric function (e.g., *Euclidean Distance*, *Cosine Similarity*,*Pearson Correlation*, etc). In particular, for an item v^s_i in S, we calculate its cosine similarity $r_{i,j}$ with each item v_j in V as follows:

$$r_{i,j} = \frac{\mathbf{h}^{sT}_i \cdot \mathbf{h}_j}{||\mathbf{h}^s_i|| \, ||\mathbf{h}_j||}, \tag{1}$$

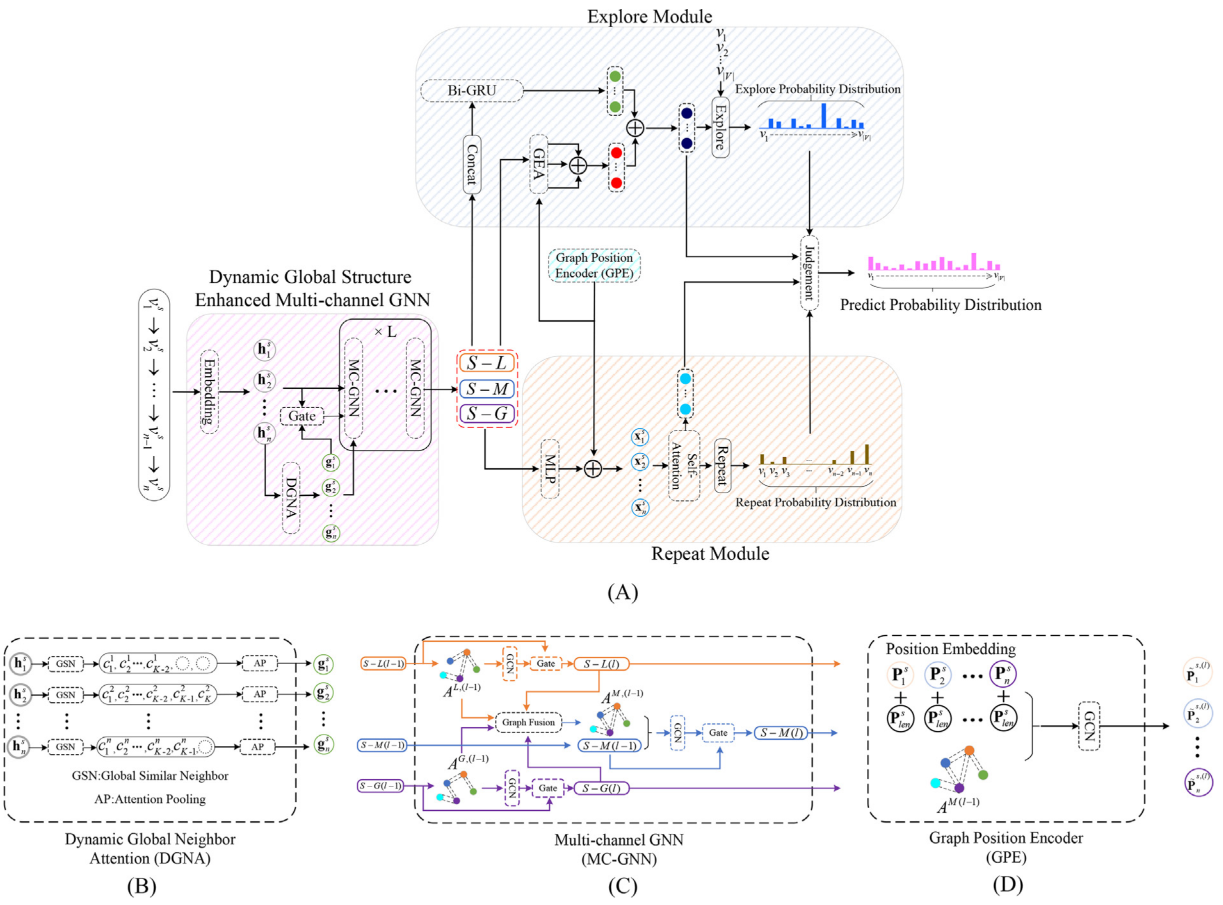
Explore Module
Bi-GRU
Concat
GEA
Explore
Explore Probability Distribution
Judgement
Predict Probability Distribution
Graph Position Encoder (GPE)
Dynamic Global Structure Enhanced Multi-channel GNN
Embedding
Gate
DGNA
MC-GNN
× L
S − L
S − M
S − G
MLP
Self-Attention
Repeat
Repeat Probability Distribution
Repeat Module
(A)
GSN:Global Similar Neighbor
AP:Attention Pooling
Dynamic Global Neighbor Attention (DGNA)
(B)
Graph Fusion
GCN
Multi-channel GNN (MC-GNN)
(C)
Position Embedding
Graph Position Encoder (GPE)
(D)

Fig. 1. Overall framework of DGS-MGNN (A) and model sub-components of DGS-MGNN (B–D).

where $\mathbf{h}_i^s \in \mathbb{R}^d, \mathbf{h}_j \in \mathbb{R}^d$ are the corresponding embeddings for $v_i^s \in S$ and $v_j \in V$, respectively. Then we take the K most similar items to v_i^s as $\mathcal{N}^i$. It is worth noting that we discard neighbors in $\mathcal{N}^i$ which are uncorrelated or negatively correlated (i.e., $r_{i,j} \leqslant 0$) to v_i^s.

Denote the corresponding embedding of $\mathcal{N}^i$ as $\mathbf{C}^i = (\mathbf{c}_1^i, \mathbf{c}_2^i, \cdots, \mathbf{c}_K^i)$, where $\mathbf{c}_j^i \in \mathbb{R}^d$. Note that $\mathbf{C}^i$ is a subset of $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{|V|})$, and it corresponds to the embeddings of the K most similar items to the node v_i^s. We utilize the Attention Pooling [18] to aggregate the global neighbors of v_i^s, and obtain its global representation $\mathbf{g}_i^s$ as follows:

$$\mathbf{g}_i^s = \sum_{j=1}^{K} \alpha_j^i \mathbf{c}_j^i, \tag{2}$$

$$\alpha_j^i = \frac{exp\left(\mathbf{w}_a^T \left[\mathbf{c}_j^i; \mathbf{h}_i^s\right]\right)}{\sum_{j=1}^{K} exp\left(\mathbf{w}_a^T \left[\mathbf{c}_j^i; \mathbf{h}_i^s\right]\right)}, \tag{3}$$

where $\mathbf{w}_a \in \mathbb{R}^{2d}$ is learnable parameter, [;] is the concatenate operation. Thus, we obtain the global representation $\mathbf{G}_s = (\mathbf{g}_1^s, \mathbf{g}_2^s, \cdots, \mathbf{g}_n^s)$ for the session S. The architecture of DGNA is shown in Fig. 1(B).

Multi-channel Graph Neural Network (MC-GNN). The goal of MC-GNN is to fuse information from different perspectives, including the global representation $\mathbf{G}_s$, the local representation $\mathbf{H}_s$, as well as the consensus representation $\mathbf{M}_s$ which will be discussed in the following. It is worth noting that all three kinds of representations are dynamically updated in our model. The architecture of MC-GNN is demonstrated in Fig. 1(C).

For each session $S = (v_1^s, v_2^s, \cdots, v_n^s)$, we construct a session graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$, where $\mathcal{V}_s$ denotes all items in S and $e_{i,j} \in \mathcal{E}_s$ denotes the edge of item-pair (v_i^s, v_j^s) in S. First, we apply *Cosine Similarity* to calculate the connection strength for item-pairs within the session S. Specifically, the connection strength $(\tau_{i,j})$ of an item-pair (v_i^s, v_j^s) is defined as follows:

$$\tau_{i,j} = \frac{\mathbf{h}_i^{sT} \cdot \mathbf{h}_j^s}{||\mathbf{h}_i^s|| \, ||\mathbf{h}_j^s||}, \tag{4}$$

where $\mathbf{h}_i^s, \mathbf{h}_j^s \in \mathbb{R}^d$. Then, we apply *softmax* to normalize the connection strength $\tau_{i,j}$ to get the edge weight:

$$e_{i,j} = \begin{cases} 0 & if : \tau_{i,j} \leqslant 0, \\ \frac{exp(\tau_{i,j})}{\sum_{j=1}^{n} exp(\tau_{i,j})} & if : \tau_{i,j} > 0. \end{cases} \tag{5}$$

It is worth noting that we set the edge weight between $\mathbf{h}_i^s$ and $\mathbf{h}_j^s$ to 0 ($e_{i,j} = 0$) if $\mathbf{h}_i^s$ and $\mathbf{h}_j^s$ are semantically uncorrelated or negatively correlated ($\tau_{i,j} \leqslant 0$). This is used to prevent the negative semantic shift of item representation learning during the graph propagation. For simplicity, we define the whole process on building a session graph (BSG) above as:

$$A^L = BSG(\mathbf{H}_s), \tag{6}$$

where A^L is the session graph that is generated by the local representation $\mathbf{H}_s = (\mathbf{h}_1^s, \mathbf{h}_2^s, \cdots, \mathbf{h}_n^s)$.

After that, we utilize the Graph Convolution Network (GCN) [10] on the session graph to enrich the representation of items. Inspired by the Light-GCN [6], we remove the feature transformation and nonlinear activation in the conventional GCN. Meanwhile, we also introduce the highway networks [25] to alleviate the problem of over-smoothing caused by the GCN with deep layers. The graph propagation process of the l-th layer is defined as follows:

$$\widetilde{\mathbf{h}}_i^{s,(l)} = g_i \cdot \mathbf{h}_i^{s,(l)} + (1 - g_i) \cdot \mathbf{h}_i^{s,(0)}, \tag{7}$$
$$\mathbf{h}_i^{s,(l)} = A_i^{L,(l-1)} \left(\widetilde{\mathbf{h}}_1^{s,(l-1)}, \widetilde{\mathbf{h}}_2^{s,(l-1)}, \cdots, \widetilde{\mathbf{h}}_n^{s,(l-1)} \right), \tag{8}$$
$$g_i = \sigma\left(\mathbf{w}_g^T \left[\mathbf{h}_i^{s,(l)}; \mathbf{h}_i^{s,(0)} \right] \right), \tag{9}$$

where $\mathbf{w}_g \in \mathbb{R}^{2d}$ is a trainable parameter, σ is the *sigmoid* function, and $A_i^{L,(l-1)}$ denotes the i-th row of $A^{L,(l-1)}$. We define the above graph propagation of the l-th layer as the Highway Graph Convolutional Networks (HWGCN):

$$\widetilde{\mathbf{H}}_s^{(l)} = HWGCN\left(A^{L,(l-1)}, \widetilde{\mathbf{H}}_s^{(l-1)} \right), \tag{10}$$
$$A^{L,(l-1)} = BSG\left(\widetilde{\mathbf{H}}_s^{(l-1)} \right), \tag{11}$$

where $\widetilde{\mathbf{H}}_s^{(l)} = \left(\widetilde{\mathbf{h}}_1^{s,(l)}, \widetilde{\mathbf{h}}_2^{s,(l)}, \cdots, \widetilde{\mathbf{h}}_n^{s,(l)} \right)$ is the session representation $\mathbf{H}_s$ that after the l-th layer of *HWGCN*.

Similarly, for each session $S = (v_1^s, v_2^s, \cdots, v_n^s)$, we construct a global graph ($A^{G,(l-1)}$) which captures the global relationship between items:

$$\widetilde{\mathbf{G}}_s^{(l)} = HWGCN\left(A^{G,(l-1)}, \widetilde{\mathbf{G}}_s^{(l-1)} \right), \tag{12}$$
$$A^{G,(l-1)} = BSG\left(\widetilde{\mathbf{G}}_s^{(l-1)} \right). \tag{13}$$

After the l-th layer *HWGCN*, we can get session representations $\widetilde{\mathbf{H}}_s^{(l)} = \left(\widetilde{\mathbf{h}}_1^{s,(l)}, \widetilde{\mathbf{h}}_2^{s,(l)}, \cdots, \widetilde{\mathbf{h}}_n^{s,(l)} \right)$ and $\widetilde{\mathbf{G}}_s^{(l)} = \left(\widetilde{\mathbf{g}}_1^{s,(l)}, \widetilde{\mathbf{g}}_2^{s,(l)}, \cdots, \widetilde{\mathbf{g}}_n^{s,(l)} \right)$ from the local and global perspectives, respectively.

In order to get more topological structure information of session, we design a Graph Fusion Module to get the consensus graph ($A^{M,(l-1)}$) that integrates the structure information of both local graph ($A^{L,(l-1)}$) and global graph ($A^{G,(l-1)}$). Similar to the process of modeling the local and global structure, we apply the *HWGCN* on the consensus graph to obtain session representation from the consensus perspective as follows:

$$\widetilde{\mathbf{M}}_s^{(l)} = HWGCN\left(A^{M,(l-1)}, \widetilde{\mathbf{M}}_s^{(l-1)} \right), \tag{14}$$
$$A_i^{M,(l-1)} = f_i \cdot A_i^{L,(l-1)} + (1 - f_i) \cdot A_i^{G,(l-1)}, \tag{15}$$
$$f_i = \sigma\left(\mathbf{w}_f^T \left[\widetilde{\mathbf{h}}_i^{s,(l)}; \widetilde{\mathbf{g}}_i^{s,(l)} \right] \right), \tag{16}$$

where $\mathbf{w}_f \in \mathbb{R}^{2d}$ is a trainable parameter, and σ is the *sigmoid* function. The structure of $A^{L,(l-1)}$ and $A^{G,(l-1)}$ complement each other to form the consensus graph $A^{M,(l-1)}$. After feature aggregation through graph $A^{M,(l-1)}$, the consensus session representation is formed as $\widetilde{\mathbf{M}}_s^{(l)} = \left(\widetilde{\mathbf{m}}_1^{s,(l)}, \widetilde{\mathbf{m}}_2^{s,(l)}, \cdots, \widetilde{\mathbf{m}}_n^{s,(l)} \right)$. Noting that when $l = 0$, we have $\widetilde{\mathbf{h}}_i^{s,(0)} = \mathbf{h}_i^s, \widetilde{\mathbf{g}}_i^{s,(0)} = \mathbf{g}_i^s$, and $\widetilde{\mathbf{m}}_i^{s,(0)}$ is the item representation that combines both the local and global representations of item:

$$\widetilde{\mathbf{m}}_i^{s,(0)} = u_i \cdot \widetilde{\mathbf{h}}_i^{s,(0)} + (1 - u_i) \cdot \widetilde{\mathbf{g}}_i^{s,(0)}, \tag{17}$$

$$u_i = \sigma\left(\mathbf{w}_u^T\left(\mathbf{W}_l\widetilde{\mathbf{h}}_i^{s,(0)} + \mathbf{W}_g\widetilde{\mathbf{g}}_i^{s,(0)}\right)\right), \tag{18}$$

where $\mathbf{w}_u \in \mathbb{R}^d, \mathbf{W}_l, \mathbf{W}_g \in \mathbb{R}^{d\times d}$are trainable parameters, and σ is the *sigmoid* function. We denote $\mathbf{M}_s = \widetilde{\mathbf{M}}_s^{(0)}, \widetilde{\mathbf{H}}_s^{(L)}, \widetilde{\mathbf{M}}_s^{(L)}$ and $\widetilde{\mathbf{G}}_s^{(L)}$ correspond to the S-L, S-M and S-G respectively in Fig. 1(A). Note that, we use the symbol L interchangeably when the intention is obvious from the context, e.g., L with parentheses (L) indicates the number of layers in the sub-module MC-GNN.

3.3. Graph position encoder (GPE)

We have noticed that the position of items is also an important factor affecting the performance of the model. Conventional methods modeling the position often focus on absolute position [15,27], which may not be suitable for the SBR task. Considering examples: S_1 = (*iPhone*, *Headset*, *Short sleeve*, *Skirt*, *Sweater*) and S_2 = (*Microphone*, *Headset*), the second item of both S_1 and S_2 is *Headset*. However, the *Headset* in S_2 can better reflect the S_2's current preference than that in S_1, because *Headset* is located in the last position of S_2, while it is the reciprocal fourth position of S_1. Therefore, the absolute position based methods [15,9] ignore the personalized position relationship for sessions and would lead to inferior performance. To overcome the shortcoming, we integrate the session length information to enhance the position embedding, i.e., we add the session length embedding to the position embedding in order to obtain an enhanced position embedding. The framework of Graph Position Encoder is demonstrated in Fig. 1(D).

$$\widetilde{\mathbf{p}}_i^s = \mathbf{p}_i^s + \mathbf{p}_{len}^s, \tag{19}$$

$$\mathbf{p}_{len}^s = Embed_{len}(S_{len}), \tag{20}$$

where $\mathbf{p}_i^s \in \mathbb{R}^d$ is the learnable embedding of position i in session S, S_{len} is the length of session $S, \mathbf{p}_{len}^s \in \mathbb{R}^d$ is the learnable length embedding of session S which can assist $\mathbf{p}_i^s$ to obtain the personalized position relation for sessions with different lengths. It is worth noting that $\widetilde{\mathbf{P}}_s = (\widetilde{\mathbf{p}}_1^s, \widetilde{\mathbf{p}}_2^s, \cdots, \widetilde{\mathbf{p}}_n^s)$ reflects a kind of sequential position structure. In order to capture inherent topological position structure of session, we further apply the consensus relationship graph $A^{M,(l-1)}$ to enhance the embedding of position:

$$\widetilde{\mathbf{P}}_s^{(l)} = HWGCN\left(A^{M,(l-1)}, \widetilde{\mathbf{P}}_s^{(l-1)}\right), \tag{21}$$

and obtain the final position embedding at the L-th layer $\widetilde{\mathbf{P}}_s^{(L)} = \left(\widetilde{\mathbf{p}}_1^{s,(L)}, \widetilde{\mathbf{p}}_2^{s,(L)}, \cdots, \widetilde{\mathbf{p}}_n^{s,(L)}\right)$.

3.4. Repeat module

After we obtain the local representation $\widetilde{\mathbf{H}}_s^{(L)}$, global representation $\widetilde{\mathbf{G}}_s^{(L)}$, consensus representation $\widetilde{\mathbf{M}}_s^{(L)}$ and position embedding $\widetilde{\mathbf{P}}_s^{(L)}$ of items after the last layer (i.e., the L-th layer), we merge them to generate the new session representation $\mathbf{X}_s = (\mathbf{x}_1^s, \mathbf{x}_2^s, \cdots, \mathbf{x}_n^s)$ as follows:

$$\mathbf{x}_i^s = \sigma\left(\mathbf{W}_x\left[\widetilde{\mathbf{h}}_i^{s,(L)}; \widetilde{\mathbf{g}}_i^{s,(L)}; \widetilde{\mathbf{m}}_i^{s,(L)}\right] + \mathbf{b}_x\right) + \widetilde{\mathbf{p}}_i^{s,(L)}), \tag{22}$$

where $\mathbf{W}_x \in \mathbb{R}^{d\times 3d}, \mathbf{b}_x \in \mathbb{R}^d$ are trainable parameters. Inspired by the Repeat-Net [19] and the Copy Mechanism [22], we utilize the Self-Attention mechanism to obtain the representation of the session and score the items that users have clicked:

$$\hat{y}_i^r = \begin{cases} z_i & if: \ v_i \in S, \\ 0 & if: \ v_i \notin S, \end{cases} \tag{23}$$

$$z_i = \frac{exp\left(\mathbf{w}_z^T\mathbf{x}_i^s\right)}{\sum_{j=1}^{n} exp\left(\mathbf{w}_z^T\mathbf{x}_j^s\right)}, \tag{24}$$

where $\mathbf{w}_z \in \mathbb{R}^d$ is a trainable parameter, and $\hat{y}^r$ is the repeat probability distribution that generated by Repeat Module. Meanwhile, we also get the session representation $\mathbf{o}_r$, which is defined as follows:

$$\mathbf{o}_r = \sum_{i=1}^{n} z_i\mathbf{x}_i^s. \tag{25}$$

3.5. Explore module

The Explore Module is designed to score items that do not appear in the current session, which consists of two submodules, i.e., Graph-Enhanced Attention Network (GEA) and Explore Prediction.

Graph-Enhanced Attention Network (GEA). Conventional attention mechanism usually dynamically assigns weights to items within a session, and it has been successfully applied in many applications [13,2,32,27]. However, it still has some defects in the task of SBR. For example, in a session the user may have some accidental or wrong clicks, conventional attention mechanism would assign attention weights to these accidental or wrong clicks, and bring noise to the representation learning process of sessions.

To alleviate this problem, we propose a novel Graph-Enhanced Attention Network (*GEA*). We apply the structure of session graph to identify noise items within a session and adjust the attention mechanism to set the attention weights of noise items to 0, so that the session representation after feature aggregation will be more precise. Taking the local session representation $\widetilde{\mathbf{H}}_{\mathbf{s}}^{(L)} = \left(\widetilde{\mathbf{h}}_1^{s,(L)}, \widetilde{\mathbf{h}}_2^{s,(L)}, \cdots, \widetilde{\mathbf{h}}_n^{s,(L)}\right)$ as an example, we consider a node $\widetilde{\mathbf{h}}_i^{s,(L)} \in \widetilde{\mathbf{H}}_{\mathbf{s}}^{(L)}$ as the accidental or wrong click if the node is negatively correlated or uncorrelated with all it's neighbors ($\left\{\widetilde{\mathbf{h}}_j^{s,(L)} | j \neq i, A_{(i,j)}^{L,(L-1)} \leqslant 0\right\}$). Then $\widetilde{\mathbf{h}}_i^{s,(L)}$ is put into a wrong clicks set S_{wrong}^L for each session. After that, the new attention weights and feature aggregation process will be defined as follows:

$$\mathbf{s}_{long}^L = \sum_{i=1}^{n} \lambda_i \widetilde{\mathbf{h}}_i^{s,(L)}, \tag{26}$$

$$\lambda_i = \frac{exp(\beta_i)}{\sum_{j=1}^{n} exp(\beta_i)} \tag{27}$$

$$\beta_i = \begin{cases} \mathbf{w}_e^T\left(\mathbf{W}_2 \widetilde{\mathbf{h}}_i^{s,(L)} + \mathbf{W}_3 \widetilde{\mathbf{p}}_i^{s,(L)}\right), & if \quad \widetilde{\mathbf{h}}_i^{s,(L)} \in \left(\widetilde{\mathbf{H}}_{\mathbf{s}}^{(L)} - S_{wrong}^L\right), \\ -\xi, & if \quad \widetilde{\mathbf{h}}_i^{s,(L)} \in S_{wrong}^L, \end{cases} \tag{28}$$

where $\mathbf{w}_e \in \mathbb{R}^d, \mathbf{W}_2 \in \mathbb{R}^{d\times d}, \mathbf{W}_3 \in \mathbb{R}^{d\times d}$ are trainable parameters, and ξ is a value near infinity. $\mathbf{s}_{long}^L$ is the local representation of the session after filtering out the user's accidental or wrong clicks. It is worth noting that for a session S_2 = (*Microphone*, *Skirt*), both items would be considered as user's accidental or wrong clicks. GEA handles this issue by setting equal attention weights for each items in the session, e.g., the attention weights of both *Microphone* and *Skirt* will be set to 0.5.

In addition, considering that user's preferences may shift, for example, given a session S_1 = (*iPhone*, *Computer*, *Watch*, *Sweater*), according to above method, the *Sweater* would be judged as user's accidental or wrong click. However, the user clicks *Sweater* just because her preference has shifted from electronic products to clothing products. To deal with the issue, we use the gate mechanism to combine long-term preferences $\mathbf{s}_{long}^L$ with the current preferences $\widetilde{\mathbf{h}}_n^{s,(L)}$ of user.

$$\hat{\mathbf{s}}_{long}^L = r \cdot \mathbf{s}_{long}^L + (1-r) \cdot \widetilde{\mathbf{h}}_n^{s,(L)}, \tag{29}$$

$$r = \sigma\left(\mathbf{w}_r^T\left[\mathbf{s}_{long}^L; \widetilde{\mathbf{h}}_n^{s,(L)}\right]\right), \tag{30}$$

where $\mathbf{w}_r^T \in \mathbb{R}^{2d}$ is a trainable parameter, r is a learnable parameter which reflects the importance of $\mathbf{s}_{long}^L$ and $\widetilde{\mathbf{h}}_n^{s,(L)}$. We define the whole process mentioned above as follows:

$$\hat{\mathbf{s}}_{long}^L = GEA\left(\widetilde{\mathbf{H}}_s^{(L)}, A^{L,(L-1)}\right), \tag{31}$$

where $\widetilde{\mathbf{H}}_s^{(L)}$ is the local representation of session, $A^{L,(L-1)}$ is the local session graph generated by $\widetilde{\mathbf{H}}_{\mathbf{s}}^{(L-1)}$.

Similarly, we can get the global session representation $\hat{\mathbf{s}}_{long}^G$ and the consensus session representation $\hat{\mathbf{s}}_{long}^M$ as follows:

$$\hat{\mathbf{s}}_{long}^G = GEA\left(\widetilde{\mathbf{G}}_s^{(L)}, A^{G,(L-1)}\right), \tag{32}$$

$$\hat{\mathbf{s}}_{long}^M = GEA\left(\widetilde{\mathbf{M}}_s^{(L)}, A^{M,(L-1)}\right). \tag{33}$$

After that we obtain the long-term structure representation of session S as follows:

$$\mathbf{s}_{long}^{str} = \hat{\mathbf{s}}_{long}^L + \hat{\mathbf{s}}_{long}^G + \hat{\mathbf{s}}_{long}^M. \tag{34}$$

Explore Prediction. Although three types of structure information of the session have been effectively modeled, the sequential information of the session also plays a critical role in capturing the user's intention. Here we utilize the Bi-GRU to get the sequential representation of session:

$$\mathbf{d}_i = \mathbf{W}_{seq}\left(\left[\widetilde{\mathbf{h}}_i^{s,(L)}; \widetilde{\mathbf{g}}_i^{s,(L)}; \widetilde{\mathbf{m}}_i^{s,(L)}\right]\right) + \mathbf{b}_{seq}, \tag{35}$$

$$\overrightarrow{\mathbf{d}_\mathbf{i}^{\mathbf{GRU}}} = \overrightarrow{GRU}\left(\overrightarrow{\mathbf{d}_{i-1}^{GRU}}, \mathbf{d}_i, \phi_{gru}\right), \tag{36}$$

$$\overleftarrow{\mathbf{d}_\mathbf{i}^{\mathbf{GRU}}} = \overleftarrow{GRU}\left(\overleftarrow{\mathbf{d}_{i+1}^{GRU}}, \mathbf{d}_i, \phi_{gru}\right), \tag{37}$$

$$\mathbf{d}_i^{GRU} = LayerNorm\left(\mathbf{W}_{gru}\left(\left[\overrightarrow{\mathbf{d}_\mathbf{i}^{\mathbf{GRU}}}; \overleftarrow{\mathbf{d}_\mathbf{i}^{\mathbf{GRU}}}\right]\right),\tag{38}$$

where $\mathbf{W}_{seq} \in \mathbb{R}^{d\times 3d}, \mathbf{W}_{gru} \in \mathbb{R}^{d\times 2d}, \mathbf{b}_{seq} \in \mathbb{R}^d, \phi_{gru}$ are trainable parameters, and $LayerNorm(\cdot)$ is the layer normalization[1]. The long-term sequential representation of session S is represented as:

$$\mathbf{s}_{long}^{seq} = \mathbf{d}_n^{GRU}, \tag{39}$$

where $\mathbf{d}_n^{GRU}$ denotes the last item representation of the session.

Then, we combine the long-term structural representation $\mathbf{s}_{long}^{str}$and long-term sequential representation $\mathbf{s}_{long}^{seq}$ of session to generate the explore probability distribution $\hat{y}_i^e$, which is formulated as follows:

$$\hat{y}_i^e = \frac{exp(\rho_i)}{um_{i=1}^{|V|} exp(\rho_i)}, \tag{40}$$

$$\rho_i = \begin{cases} \mathbf{s}_e^T \mathbf{h}_i & , \quad if \quad v_i \notin S, \\ -\infty & , \quad if \quad v_i \in S, \end{cases} \tag{41}$$

$$\mathbf{s}_e = \mathbf{s}_{long}^{str} + \delta \mathbf{s}_{long}^{seq}, \tag{42}$$

where $\mathbf{h}_i \in \mathbb{R}^d$ is the embedding of the i-th item in V, and δ is a hyperparameter.

3.6. Judgment module

In this subsection, we utilize the gate mechanism to integrate $\hat{y}^r$ and $\hat{y}^e$ to generate the final predict probability distribution, which is formulated as:

$$\hat{y}_i = p \cdot \hat{y}_i^r + (1-p)\cdot \hat{y}_i^e, \tag{43}$$

$$p = \sigma\left(\mathbf{w}_p^T([\mathbf{o}_r; \mathbf{s}_e]\right), \tag{44}$$

where $\mathbf{w}_p \in \mathbb{R}^{2d}$ is trainable parameter. $\mathbf{o}_r$ (see Eq. (25)) and $\mathbf{s}_e$ (see Eq. (42)) are representations of session from Repeat Module and Explore Module respectively. We optimize the model by minimizing the cross-entropy loss:

$$\mathscr{L} = -\sum_{i=1}^{|V|} y_i log(\hat{y}_i) + (1-y_i) log(1-\hat{y}_i), \tag{45}$$

where y_i is the one-hot vector of the ground truth item, and $\hat{y}_i$ is the predicted probability of the user clicking the i-th item.

4. Experiments

In order to verify the effectiveness of DGS-MGNN, we conduct extensive experiments to answer the following research questions:

- **RQ1:** Does the performance of DGS-MGNN beat the best performing baseline?
- **RQ2:** How each model component affects the performance of DGS-MGNN, including the Dynamic Global Neighbor Attention Network (DGNA), Multi-channel Graph Neural Network (MC-GNN), Graph Position Encoder, and Graph-Enhanced Attention Network (GEA).
- **RQ3:** What are the influence of the structure information, the sequence information, and the number of global neighbors on the model performance?
- **RQ4:** How well does DGS-MGNN perform on sessions with different lengths?
- **RQ5:** What is the computational complexity of the proposed DGS-MGNN?

4.1. Datasets

We employ three widely used benchmark datasets, including *Diginetica*, *Yoochoose* and *Retailrocket*, to evaluate the performance of DGS-MGNN and baselines.

- *Diginetcia*[1] is obtained from CIKM Cup 2016. Because of its transaction data, it is often used in session-based recommendation task. Following [15,32,6,33], we extract data of the last week as the test set.
- *Yoochoose*[2] is collected from RecSys Challenge 2015, which contains click streams from an e-commerce website within a 6 month period. Since Yoochoose is very large, following [15,33], we extract the most recent portions 1/64 and 1/4 of the training sequences as the training data, denoted as *"Yoochoose1/64"* and *"Yoochoose1/4"*, data of last day as the testing data.
- *Retailrocket*[3] is obtained from Kaggle competition 2016, which contains user behaviors of e-commerce platforms within 4–5 months. We extract the most recent portions 1/4 of the training sequences as the training data, and data of the last 15 days as the testing data.

Following [33,32,15], sessions of length 1 and items appearing less than 5 times were filtered across all the three datasets. We also use sequence splitting pre-processing to increase training samples, *i.e*, for a session sequence $\mathscr{S} = (v_1, v_2, v_3, \cdots, v_n)$, we generate sequences and labels as $([v_1], v_2), ([v_1, v_2], v_3), \cdots, ([v_1, \cdots, v_{n-1}], v_n)$ for training and testing.

4.2. Baselines

To evaluate the performance of our model comprehensively, we compare it with eleven baseline methods which can be roughly grouped into three categories, i.e., traditional recommendation methods, RNN and Attention based methods, graph neural network based methods. The details of all baseline are briefly described as follows:

Traditional recommendation methods:

- **POP** [8]: This is a frequently used baseline method in recommendation system, which recommends the top-N frequent items in the training set.
- **Item-KNN** [21]: It is based on collaborative filtering to recommend items similar to those in the current session. It is one of the most common item-to-item solutions for recommendation, and is usually considered as a strong baseline.
- **FPMC** [20]: This method leverages matrix factorization (MF) and Markov chains (MC) together, where the sequential data is modeled by the transition matrix and all transition matrices are user-specific. It introduces a factorization model which gives a low-rank approximation to the transition cube where each slice is a user-specific transition matrix of an underlying MC on the users basket history.

RNN and Attention based methods:

- **GRU4REC** [8]: It applies GRU to simulate the user's sequential behavior, and modifies the basic GRU by introducing session-parallel mini-batches, mini-batch based output sampling and ranking loss function.
- **NARM** [11]: It employs RNN to model the user's sequential behavior and captures a user's main preference with the attention mechanism. The recommendation probability for each candidate item is computed by a bi-linear matching scheme based on the unified session representation.
- **STAMP** [13]: This method attempts to address the user interests drift issue by capturing both a user's general interests from the long-term memory of a session and her current interests from the short-term memory of the last-click.
- **CSRM** [29]: This method proposes to leverage collaborative neighborhood information to session-based recommendations. It captures a user's own information in the current session with an inner memory encoder and predicts the intent of the current session by exploiting collaborative information from neighborhood sessions with an outer memory encoder.
- **DSAN** [38]: DSAN leverages a dual sparse attention network for the task of SBR. It first explores the interaction between each item within the session and learns a representation of the target representation by a self-attention network. Then it applies a vanilla attention network to obtain the importance of items within the current session to get the session representation. After that, a neural network is utilized to combine both the target representation and the session representation to get the final representation.

Graph neural network based methods:

- **SR-GNN** [33]: SR-GNN attempts to capture the complex transitions of items by modeling session sequences as graph structured data. It employs the gated graph neural networks to obtain the representation of items and combines a self attention mechanism to generate session representation.
- **TAGNN** [37]: This method captures the complex item transitions within sessions by modeling items in sessions as session graphs and obtains item embeddings using graph neural networks. It also introduces a target attentive module to reveal the relevance of historical actions given a target item to improve the session representations.

[1] https://competitions.codalab.org/competitions/11161.
[2] http://2015.recsyschallenge.com/challege.html.
[3] https://www.kaggle.com/retailrocket/ecommerce-dataset.

- **COTREC** [34]: COTREC combines self-supervised learning with co-training for session-based recommendation. For co-training, it augments two different views, i.e., item view and session view, by exploiting the internal and external connectivities of sessions.
- **GCE-GNN** [32]: GCE-GNN learns item representations from two different levels, i.e., session-level and global-level. The session-level item representation aims to model pairwise item-transitions within the current session while the global-level item representation attempts to model pair-wise item-transitions over all sessions.

4.3. Parameter setting

In DGS-MGNN, we set the training batch size to 256, the dimension of item embeddings is 256 in all experiments. The number of global neighbors K is 10 and the number of $HWGCN$ layers is 1. For the setting of hyperparameters, we select the last 20% of the training data as the validation set which will be used to choose the best hyperparameters. When the same datasets and evaluation settings are utilized in existing literature, we will employ their best hyperparameter settings reported in these works and directly report their results. When the datasets or evaluation settings between the existing literature and ours are different, we will search the set of hyperparameters on the validation set, and report the results with the best hyperparameter settings. More details about the best hyperparameter settings of baseline models are shown in Appendix A. Following [15,16,32,37], all parameters are initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. We use the Adam optimizer with the initial learning rate 0.001, which will decay by 0.1 after every 3 epochs. The value of δ in Eq. (42) is set to 0.4. The dropout [24] is utilized, which is set to 0.4. Meanwhile, the L_2 penalty is set to 10^{-5} in order to avoid overfitting. When investigating the effectiveness of each component of the proposed model, we follow the same hyperparameter settings as our proposed model, except as otherwise specified.

4.4. Evaluation metrics

Following [33,16,37,32], we adopt two widely used ranking based metrics P@20 and MRR@20 to evaluate the performance of all methods. Note that a higher value of P@K and MRR@K indicate a better model performance.

- **P@K**(Precision): It measures the proportion of cases when the target item is ranked within the top-K recommendations.

$$P@K = \frac{n_{hit}}{N}, \tag{46}$$

where N indicates the number of test cases and n_{hit} is the number of cases that the target item is in the top-K items of the ranked list.
- **MRR@K**(Mean Reciprocal Rank): It is the average of the reciprocal ranks of the target item in the recommendation list. This metric considers the position of correct recommended items in a ranked list.

$$MRR@K = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}, \tag{47}$$

where N is the number of test cases and $rank_i$ is the position of the i-th target item in the list of recommended items. Note that if the target item is not in the top-K items, its MRR@K score is set to 0.

4.5. Overall performance (RQ1)

In Table 1,2, we report the overall experimental results of DGS-MGNN (including the significance testing) and all baselines on three benchmark datasets. From Table 2, we can observe that the performance of traditional method POP is worst since it only focuses on the frequency of items and ignores the characteristics of behavior differences and interest shift of user. FPMC performs much better than POP because it captures user's preferences by applying the first-order Markov Chain and Matrix Factorization. Among all traditional methods, Item-KNN usually achieves the best performance and surpasses both POP and FPMC.

Comparing with traditional methods, neural network based methods usually outperform these traditional methods. GRU4REC is the first RNN based approach for the task of session based recommendation, which is superior to all traditional methods only with an exception on the Diginetica dataset. This shows the capability of RNN in modeling sequence data. Both NARM and STAMP perform better than GRU4REC as they further incorporate the attention mechanism in order to deal with the issue where user's preference changes within the session. CSRM achieves a better performance than NARM and STAMP on all datasets as it further introduces auxiliary sessions to enhance the current session representation. DSAN obtains the best performance among all RNN and attention based methods. This mainly is attributed to that DSAN incorporates a dual sparse attention network where a self-attention network is utilized to explore the interaction between each item within the current session, and a vanilla attention network is applied to capture the importance of items within the session.

Among all baseline methods, these graph neural network (GNN) based methods usually show a superior performance. The main reason may be that GNN slightly relaxes the assumption of temporal dependence between consecutive items

Table 1
Statistics of datasets used in the experiment.

Dataset	Diginetica	Yoochoose1/4	Yoochosse1/64	Retailrocket
#click	982961	8326407	557248	2756101
#train	719470	5917745	369859	175325
#test	60858	55898	55898	24283
#unique items	43097	29618	16766	22305
#Average length	5.12	7.42	6.16	3.96

Table 2
Comparison of the performance of all methods on all datasets in terms of P@20 and MRR@20. The best and the second best performing results in each column are in bold and underlined respectively. The significance test between our model DGS-MGCN and the best baseline GCE-GNN has been conducted, and significant improvements over GCE-GNN are marked with † (t-test, p⩽0.01). The scores marked by ∗ are based on our re-implementation when the datasets or evaluation settings in existing literature are different from ours.

Dataset	Diginetica		Yoochoose1/64		Yoochoose1/4		Retailrocket	
Methods	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20
POP	0.89	0.20	6.71	1.65	1.37	0.31	1.97∗	0.75∗
Item-KNN	35.75	11.57	51.60	21.81	52.31	21.70	10.23∗	3.96∗
FPMC	26.53	6.95	45.62	15.01	51.86	17.50	9.65∗	4.32∗
GRU4REC	29.45	8.33	60.64	22.89	59.53	22.60	41.35∗	25.54∗
NARM	49.70	16.17	68.32	28.63	69.73	29.23	59.46∗	41.48∗
STAMP	45.64	14.32	68.74	29.67	70.44	30.00	58.48∗	38.96∗
CSRM	50.55	16.38	69.85	29.71	70.63	29.68	61.09∗	40.28∗
DSAN	53.76	18.99	69.68∗	31.23∗	72.06∗	32.22∗	62.56∗	42.39∗
SR-GNN	50.73	17.59	70.57	30.94	71.36	31.89	60.19∗	39.64∗
TAGNN	51.31	18.03	71.02	31.12	71.32∗	32.11∗	59.31∗	39.65∗
COTREC	54.18	19.07	70.72∗	29.36∗	70.48∗	29.19∗	63.81∗	44.48∗
GCE-GNN	54.22	19.04	70.91∗	30.63∗	71.40∗	31.49∗	63.29∗	40.35∗
DGS-MGNN	**55.54**†	**19.68**†	**72.62**†	**32.49**†	**73.10**†	**33.55**†	**66.61**†	**45.78**†

and models more complex user item transitions as pairwise relations (e.g., directed graph). For example, SR-GNN [33] attempts to capture more implicit connections between items in a session and models session sequences as graph-structured data. TAGNN [37] further takes into account user preferences with target-aware attentions. COTREC [34] obtains a comparable performance to SR-GNN and TAGNN. It augments two different views by exploiting both intra- and inter-connectivity patterns which can alleviate the data sparsity issue. Among all GNN-based methods, GCE-GNN [32] mostly presents the best performance on all datasets. This is because GCE-GNN effectively learn item representations from both global context, i.e., other sessions and local context, i.e., the current session.

Our proposed approach DGS-MGCN consistently outperforms all state-of-the-art baseline methods. Specifically, DGS-MGCN demonstrates a significant improvement (t-test, p⩽0.01) over the best performing baseline GCE-GNN on all datasets, and the performance improvements of DGS-MGC over GCE-GNN on Diginetica, Yoochoose1/64, Yoochoose1/4 and Retailocket in terms of MRR@20 are 3.36%, 6.07%, 6.54% and 13.46%, respectively. Similar performance improvements can also be observed in terms of P@20. The main reason is that DGS-MGNN can effectively integrate the information of global neighbors in a dynamical way. And it introduces a novel MC-GNN to learn more abundant representation of items from multiple perspectives. In addition, it incorporates Graph Position Encoder and GEA to enhance the position embeddings and filter noise items within session, respectively.

4.6. Impact of DGNA and MC-GNN (RQ2)

In order to verify the effectiveness of Dynamic Global Neighbor Attention Network (DGNA) and the superiority of Multi-channel Graph Neural Network (MC-GNN), we designed four comparing models:

- **DGS-MGNN w/o DGNA**: Removing the Dynamic Global Neighbor Attention Network (DGNA) in DGS-MGNN.
- **DGS-MGNN-MLP**: Replacing the MC-GNN in DGS-MGNN with a Multilayer Perceptron (MLP). For this variant, we choose a 2 layers MLP (i.e., the hidden units are 512 and 256).
- **DGS-MGNN-GGNN**: Replacing the MC-GNN in DGS-MGNN with the Gated Graph Neural Network (GGNN) [33], which is used in SR-GNN, and the graph is constructed in the same way as that in SR-GNN [33]. The input dimension of GGNN is set to 256.

- **DGS-MGNN-GAT**: Replacing the MC-GNN in DGS-MGNN with the Graph Attention Neural Network (GAT) [28], which is applied in GCE-GNN, and the graph constructed in DGS-MGNN-GAT is in the same way as that in GCE-GNN [32]. The input dimension of GAT is set to 256.

Table 3 shows the performance of all models. We can observe the removal of DGNA from DGS-MGNN will lead to considerably performance degradation, which verifies the effectiveness of DGNA. Considering the module MC-GNN, we can see that DGS-MGNN with MC-GNN performs consistently better than all other variants, such as replacing MC-GNN with MLP, GGNN, and GAT. Among all three variants, we can observe that DGS-MGNN-GGNN almost has no improvement compared with DGS-MGNN-MLP, the reason is attributed to that DGS-MGNN-GGNN does not distinguish the connection strength between items. Moreover, it also suffers from the accidental or wrong clicks of user, which may bring noise information to user's representation. DGS-MGNN-GAT achieves the best performance among all compared variants since it alleviates the issue of connection strength by introducing attention weight as the connection strength between items. However, similar to DGS-MGNN-GGNN, DGS-MGNN-GAT still encounters the issue of noise information caused by the accidental or wrong clicks of user. Different from DGS-MGNN-GAT, our proposed model DGS-MGNN with MC-GNN can effectively deal with the connection strength and the noise information issue via dynamically adjust the structure of graph and the connection strength between items. In addition, we also introduce the graph correction operation to filter out noisy edges (i.e., we remove connections between items which are semantically uncorrelated or negatively correlated), and design the Graph Fusion Module to enrich the representation of session.

4.7. Impact of graph position encoder (RQ2)

In this section, we investigate the effectiveness of the Graph Position Encoder module. As position embedding technique has been widely used in the SBR task, however the absolute position embedding technique used in SASRec [9] and SGNN–HN [15] would not achieve the ideal performance for the SBR task. To verify the effectiveness of our proposed Graph Position Encoder, we compare our method with two variants:

- **DGS-MGNN w/o GPE**: Removing Graph Position Encoder in DSG-MGNN.
- **DGS-MGNN-POS**: Replacing the Graph Position Encoder with the conventional Position Encoder which is employed in SASRrec [9] and SGNN–HN [15].

Fig. 2 shows the performance of different comparing models. The results demonstrate that DGS-MGNN with the Graph Position Encoder is superior to the two variants DGS-MGNN w/o GPE and DGS-MGNN-POS on all datasets in terms of both metrics. More precisely, the models with a position encoder (i.e., our proposed DGS-MGNN and the variant DGS-MGNN-POS) consistently outperform the model without taking position embedding into consideration (i.e., the variant DGS-MGNN w/o GPE). In addition, our proposed DGS-MGNN further performs considerably better than the variant DGS-MGNN w/o GPE which utilizes the conventional Position Encoder. As described in the subSection 3.3, DGS-MGNN-POS cannot model the personalized position relationship of sessions with different length. On the contrary, Graph Position Encoder introduces the information of session length into to the position embedding to alleviate the above problem. Meanwhile, Graph Position Encoder also captures the inherent topological position structure of session.

4.8. Impact of graph-enhanced attention network (RQ2)

To verify the impact of Graph-Enhanced Attention Network (GEA) in the Explore Module on the performance of recommendation, we employ the following three variants:

- **DGS-MGNN-S**: Replacing Graph-Enhanced Attention Network (GEA) with Sum-Pooling.
- **DGS-MGNN-M**: Replacing Graph-Enhanced Attention Network (GEA) with Mean-Pooling.
- **DGS-MGNN-SA**: Replacing Graph-Enhanced Attention Network (GEA) with Self-Attention Network.

As we can see in Table 4, the sum-pooling based model DGS-MGNN-S obtains the worst performance. In constrast, the mean-pooling based model DGS-MGNN-M demonstrates a better performance than DGS-MGNN-S. Among all three variants, the self-attention network based model DGS-MGNN-SA achieves the best performance, which shows the effectiveness of introducing the attention mechanism to aggregate information dynamically according to the importance coefficient of items.

Comparing to all three variants, our proposed model DGS-MGNN with GEA consistently demonstrates the best performance on all datasets. This is mainly because in a session a user may have some accidental or wrong clicks, conventional attention mechanism will assign attention weights to these accidental or wrong clicks, thus bring noise to the representation of session. On the contrary, the Graph-Enhanced Attention Network (GEA) module attempts to apply the structure of session graph to identify user's accidental or wrong clicks within the session and then adjust the attention mechanism to filter out the noisy items by setting the attention weights of noise items to 0.

Table 3
Performance comparison of the model variants with respect to the two sub-modules DGNA and MC-GNN on all datasets in terms of P@20 and MRR@20. The best performing results in each column are in bold.

Datasets	Diginetica		Yoochoose1/64		Yoochoose1/4		Retailrocket	
Methods	P@20	M@20	P@20	M@20	P@20	M@20	P@20	M@20
DGS-MGNN w/o DGNA	53.95	19.42	71.88	32.15	72.46	32.98	63.92	45.21
DGS-MGNN-MLP	53.51	18.33	71.12	31.10	71.72	32.01	62.85	44.82
DGS-MGNN-GGNN	53.68	17.92	68.85	30.52	69.45	31.23	62.52	45.11
DGS-MGNN-GAT	54.22	18.74	71.35	31.36	72.13	32.33	63.75	44.86
DGS-MGNN	**55.54**	**19.68**	**72.62**	**32.49**	**73.10**	**33.55**	**66.61**	**45.78**

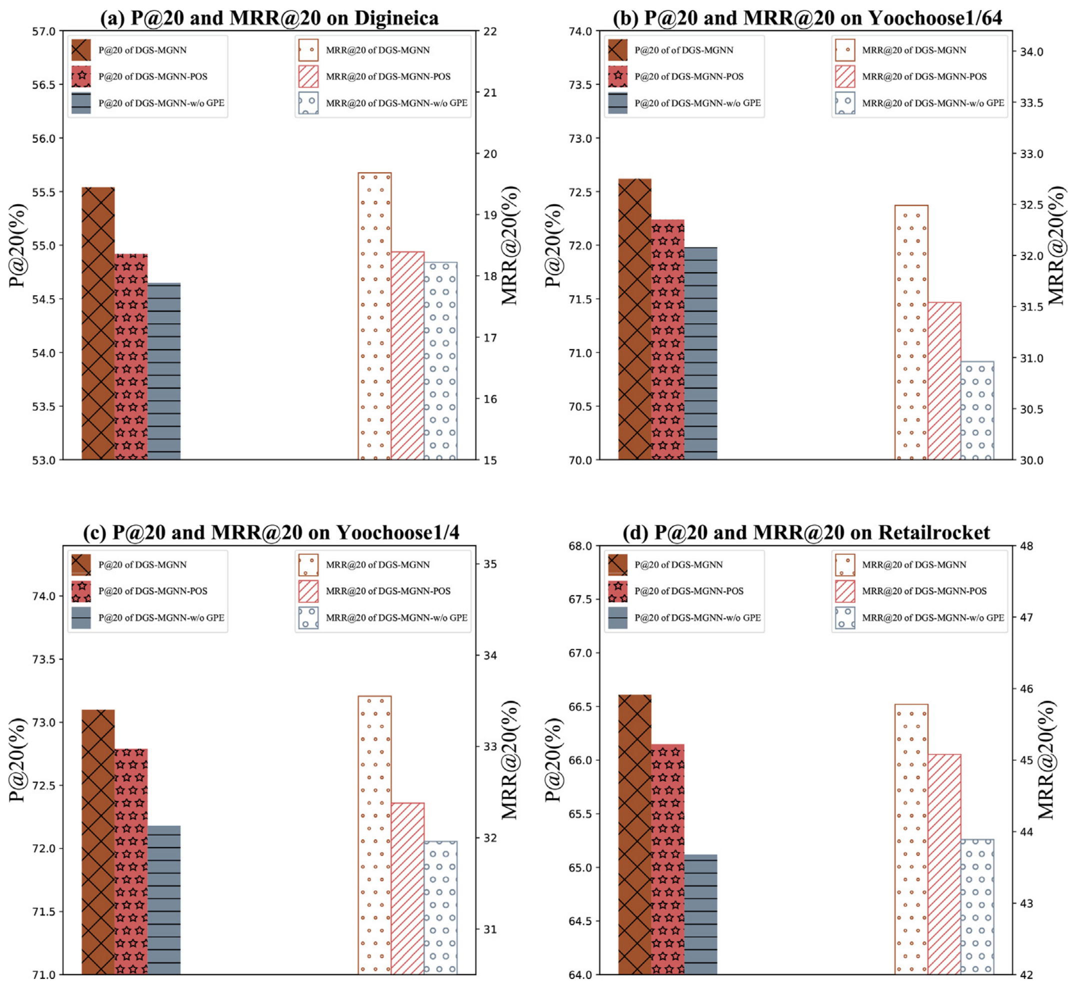

Fig. 2. Performance comparison of the model variants with respect to the graph position encoder on all datasets in terms of P@20 and MRR@20.

Table 4
Performance comparison of the model variants with respect to the sub-module graph-enhanced attention network (GEA) on all datasets in terms of P@20 and MRR@20. The best performing results in each column are in bold.

Datasets	Diginetica		Yoochoose1/64		Yoochoose1/4		Retailrocket	
Methods	P@20	M@20	P@20	M@20	P@20	M@20	P@20	M@20
DGS-MGNN-S	53.07	18.91	70.02	30.81	70.87	31.68	63.82	44.48
DGS-MGNN-M	53.68	19.42	71.95	32.21	72.55	33.12	62.79	45.12
DGS-MGNN-SA	54.35	19.44	72.31	32.38	72.81	33.34	66.45	45.64
DGS-MGNN	**55.54**	**19.68**	**72.62**	**32.49**	**73.10**	**33.55**	**66.61**	**45.78**

4.9. Impact of structure information and sequential information (RQ3)

To explore the impact of structure information $\mathbf{s}_{long}^{str}$ (see Eq. (34)) and sequence information $\mathbf{s}_{long}^{seq}$ (see Eq. (39)) of session in Explore Module on the performance of DGS-MGNN, we compare our model with the following variants:

- **DGS-MGNN-STR**: A variant of DGS-MGNN, which solely keeps the long-term structure representation of the current session while neglects the long-term sequential information in DGS-MGNN.
- **DGS-MGNN-SEQ**: A variant of DGS-MGNN, which only maintains the long-term sequential representation of the current session while discards the long-term structure information in DGS-MGNN.

Fig. 3 reports the experimental results. We can observe that our proposed model DGS-MGNN, which captures both the long-term structure representation as well as the long-term sequential representation of the current session, obtains the best performance on all datasets. When DGS-MGNN is only equipped with either the structure information or sequential information of the current session, its performance will degrade significantly. In addition, among the two variants, DGS-MGNN-STR is superior to DGS-MGNN-SEQ, which demonstrates that: (1) Incorporating the long-term structure representation of the current session is more important than incorporating the long-term sequential representation of the current session; (2) The two long-term representations are complementary to each other, and a combination of them will boost the performance considerably.

4.10. Impact of the number of global neighbors (RQ3)

To explore the impact of different number of global neighbors (i.e., K) on the performance of DGS-MGNN, we analyze the performance of DGS-MGNN with various number of K ranging from 0 to 50, and results are shown in Fig. 4. We can observe

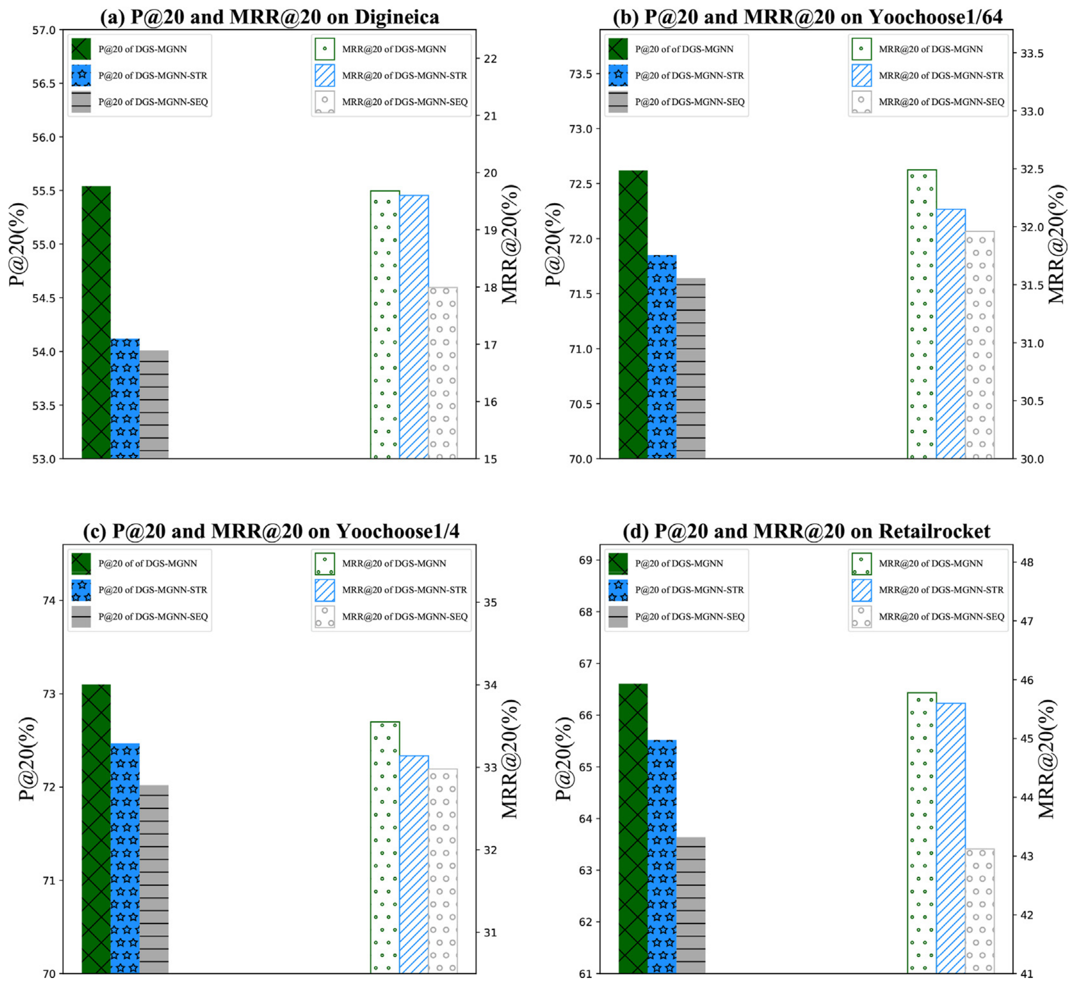

Fig. 3. Performance comparison of the model variants with different structure information on all datasets in terms of P@20 and MRR@20.

that on the Diginetica dataset the performance of our method DGS-MGNN first increases and reaches the peak when $K = 10$. If we further raise K, the performance will become stable or drop slightly. On the Yoochoose1/64 dataset, the performance of DGS-MGNN increases gradually and reaches the peak when $K = 20$, after that there is a performance degradation if we continue to enlarge the number of global neighbors. Similar results can also be observed on other two datasets, i.e., Yoochoose1/4 and Retailrocket. The results reveal that our proposed method DGS-MGNN can obtain promising performance with a relatively small number of global neighbors.

4.11. Model performance on sessions with different lengths (RQ4)

To verify the model performance on sessions with different lengths, we split sessions into two groups (i.e., long sessions and short sessions), where sessions with a length greater than 5 are considered as long sessions and the remaining sessions are considered as short sessions. We compare our proposed method DGS-MGNN with two most competitive baseline models, i.e., DSAN and GCE-GNN, on both long and short sessions. From Fig. 5, we can observe that: First, the performance of all three methods on the short sessions are superior to their corresponding performance on the long sessions. This can be attributed to that long sessions usually contains more complicated user interest (e.g., interest shifts) which is difficult to capture, while user interests within short sessions are usually simple. Second, our proposed model DGS-MGNN consistently performs better than the two state-of-the-art baseline methods on both short and long sessions in terms of both metrics on all datasets. In addition, comparing with DSAN and GCE-GNN, the performance improvements of DGS-MGNN on the long sessions

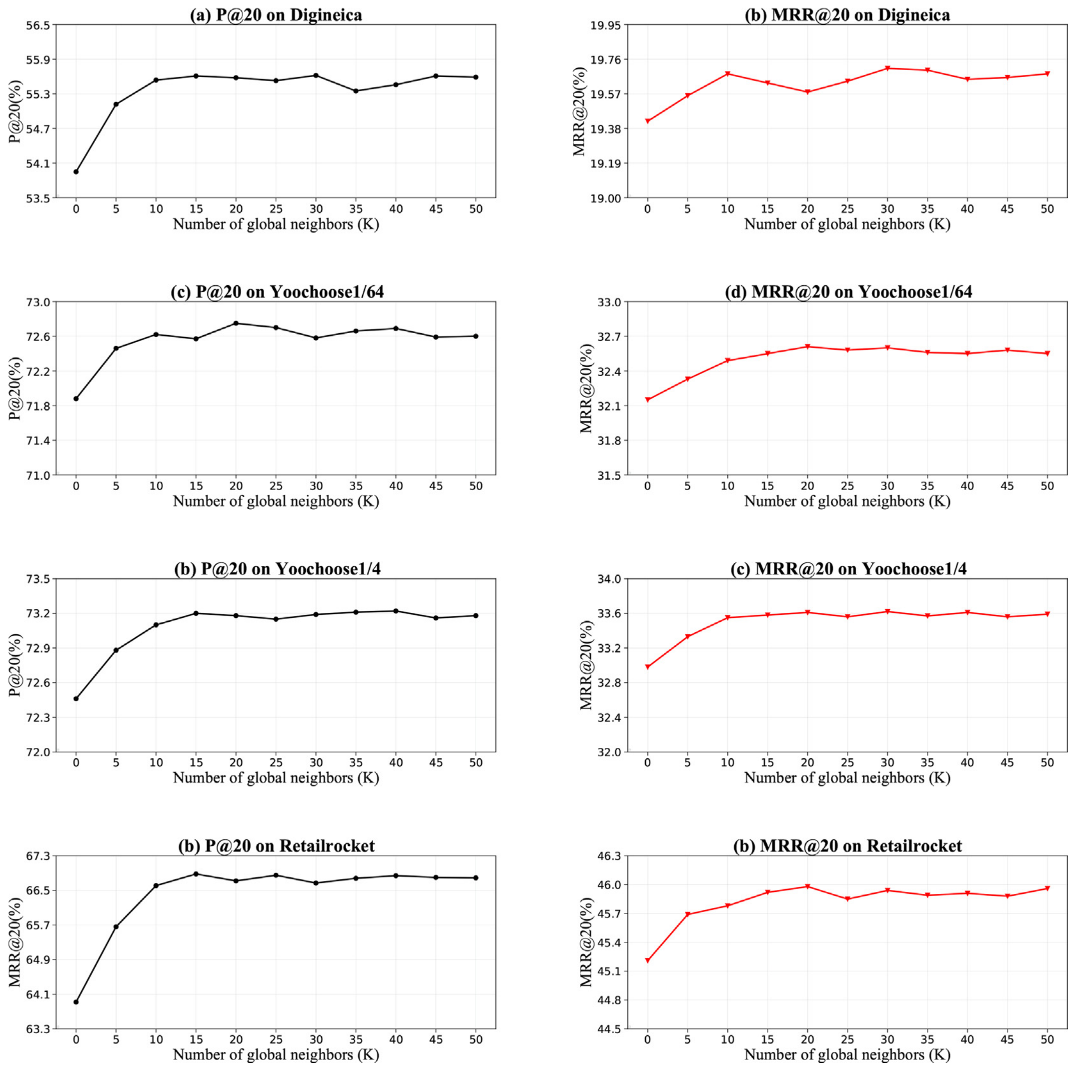

Fig. 4. Performance of DGS-MGNN with different number of global neighbors K on all datasets in terms of P@20 and MRR@20.

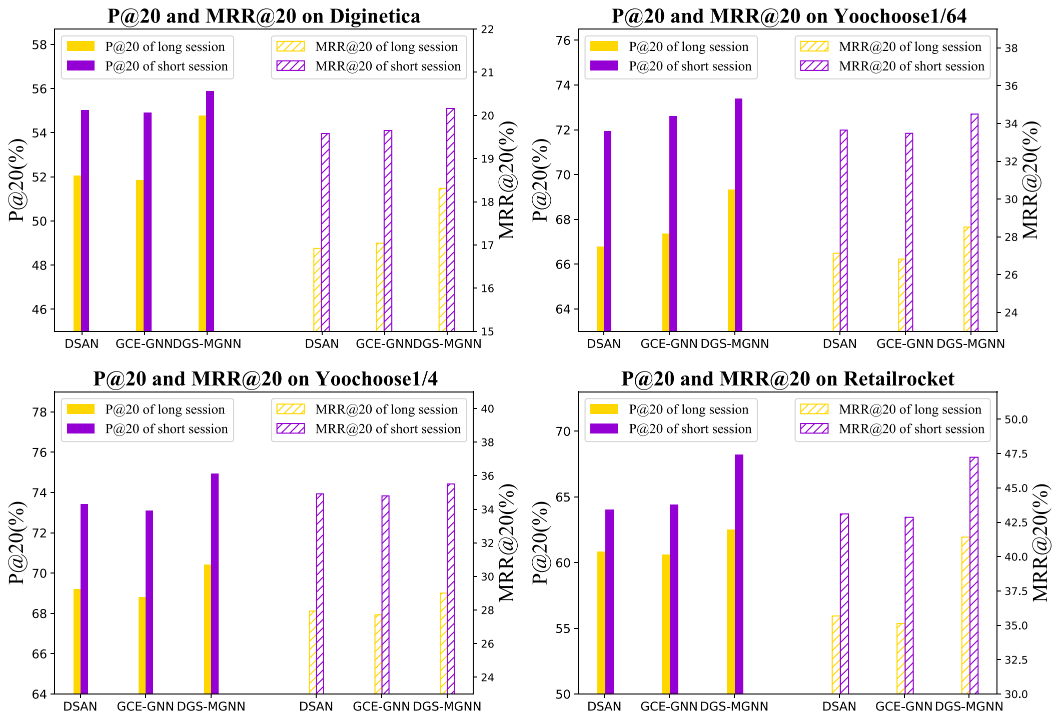

Fig. 5. Performance of DGS-MGNN on sessions with different lengths on all datasets in terms of P@20 and MRR@20.

are considerably larger than the counterparts on the short sessions. For example, on the Diginetica dataset, the performance improvements of DGS-MGNN over DSAN and GCE-GNN on the short sessions are 1.58% (2.96%) and 1.78% (2.60%) in terms of P@20 (MRR@20), respectively. While the corresponding improvements on the long sessions are 5.23% (8.22%) and 5.61% (7.45%), respectively. Similar results are also observed on other three datasets.

4.12. Computational complexity (RQ5)

In this sub-section, we compare the computational complexity of our proposed approach DGS-MGNN with other five most competitive baseline methods, including SR-GNN, TAGNN, DSAN, COTREC and GCE-GNN. In Table 5, we report the theoretical computational complexity as well as training time and memory costs of these methods. The computational complexity of SR-GNN is $O\left(s\left(nd^2+n^3\right)+d^2\right)$ where n is the session length, d is the dimension of item embeddings. For simplicity, we use s to denote the number of layers in different graph neural networks, i.e., the gated graph neural network (GGNN) [12] in both SR-GNN and TAGNN, the graph attention network (GAT) [28] in GCE-GNN, the graph convolution network (GCN) [10] in COTREC, and the multi-channel graph neural network (MC-GNN) in our proposed DGS-MGNN. Similar to SR-GNN, TAGNN also applies the GGNN to learn node vector. Moreover, it further incorporates a local target attentive module to measure

Table 5
Analysis of the computational complexity of different comparing models.

Datasets		Diginetica		Yoochose1/64		Yoochose1/4		Retailrocket	
Method	Complexity	Time	Memory	Time	Memory	Time	Memory	Time	Memory
SR-GNN	$O\left(s\left(nd^2+n^3\right)+d^2\right)$	80s	1359M	72s	1311M	1145s	1311M	69s	1435M
TAGNN	$O\left(s\left(nd^2+n^3\right)+n\lvert V\rvert d^2+d^2\right)$	827s	9745M	331s	8757M	5275s	8757M	250s	10621M
DSAN	$O\left(n^2d+d^2\right)$	128s	1523M	64s	1589M	878s	1589M	18s	1437M
COTREC	$O\left(s\lvert V\rvert^2d^2+sb^2d^2+d^2\right)$	1459s	3851M	961s	3079M	16023s	3079M	293s	2305M
GCE-GNN	$O\left(sn^2d+nKd+d^2\right)$	152s	2066M	76s	1699M	906s	1699M	20s	2208M
DGS-MGNN	$O\left(sn^2d+n\lvert V\rvert d+d^2\right)$	236s	3719M	126s	3371M	2058s	3371M	45s	2479M

attention scores between each item in the current session and all items in V. Therefore, it shows a higher computational complexity, i.e., $O\left(s\left(nd^2+n^3\right)+n|V|d^2+d^2\right)$ where $|V|$ indicates the number candidate items, than that of SR-GNN. For DSAN and GCE-GNN, the computational complexity are $O\left(n^2d+d^2\right)$ and $O\left(sn^2d+nKd+d^2\right)$, respectively. Among all these baseline methods, COTREC takes the highest computational complexity $O\left(s|V|^2d^2+sb^2d^2+d^2\right)$, where b denotes the batch size. For our proposed method DGS-MGNN, the computational complexity is $O\left(sn^2d+n|V|d+d^2\right)$, where the main costs are from MC-GNN module and calculating the dynamic global neighbor attention. From the results, we can observe that the computational complexity of DGS-MGNN is much lower than that of TAGNN and COTREC, and slightly higher than that of SR-GNN, DSAN, and GCE-GNN.

For the consumption of the training time, we can observe that the baseline DSAN costs less training time than other methods on all datasets with an exception on the Diginetica dataset. The baseline COTREC shows the highest consumption of the training time on all datasets. For our proposed approach DGS-MGNN, its cost of training time is much lower than that of TAGNN and COTREC, and slightly higher than that of SR-GNN, DSAN and GCE-GNN on most datasets, which is consistent with the results of the theoretical computational complexity. For the cost of the model memory, the two baselines SR-GNN and TAGNN demonstrate the lowest and highest memory cost on most datasets, respectively. The memory cost of COTREC is lower than TAGNN while it is considerably larger than other baselines. The remaining two baselines DSAN and GCE-GNN shows a comparable or slightly higher memory cost as compared with SR-GNN. The memory cost of our method DGS-MGNN is much lower than that of TAGNN, and competitive to that of COTREC. Based on the analysis, the computational complexity DGS-MGNN is moderate and can be practicable for potential applications due to its relative low costs of computational complexity.

5. Conclusion

In this paper, we propose a novel method, named DGS-MGNN, for session-based recommendation. Specifically, we develop a dynamic global structure enhanced multi-channel graph neural network to dynamically capture information from three different perspectives, i.e., local, global and consensus representation. Moreover, we design a graph-enhanced attention network to filter out noisy items within a session and obtain better long-term structure representation of the current session. A novel position embedding embedding methods, i.e., Graph Position Encoder, is proposed by enhancing the embedding processing with the session length information and the inherent topological position structure within a session. The experimental results demonstrate that our proposed approach significantly outperforms the state-of-the-art methods in terms of both metrics on all datasets. In addition, we also analyse the computational costs of the proposed methods, and the results show that DGS-MGNN has a moderate computational complexity as compared with the state-of-the-art methods, which makes it practicable for potential applications.

CRediT authorship contribution statement

Xiaofei Zhu: Conceptualization, Writing – original draft, Writing – review & editing, Supervision. **Gu Tang:** Conceptualization, Methodology, Software, Writing – original draft. **Pengfei Wang:** Writing – review & editing, Supervision. **Chenliang Li:** Writing – review & editing, Supervision. **Jiafeng Guo:** Writing – review & editing, Supervision. **Stefan Dietze:** Writing – review & editing, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the National Natural Science Foundation of China [Grant No. 62141201]; the Major Project of Science and Technology Research Program of Chongqing Education Commission of China [grant number KJZD-M202201102]; the Federal Ministry of Education and Research [Grant No. 01IS21086].

Appendix A. Model parameters

In our experiments, we choose the best hyperparameters for baseline models by searching the set of hyperparameters on the validation set. It is worth noting that we will directly utilize the best hyperparameter settings for baseline models when the same datasets and evaluation settings are utilized in existing literature. In Table 6, we present the best hyperparameter setting for baseline models used in our experiments.

Table 6
The best hyperparameter settings for baseline models used in our experiments. The hyperparameters with * indicate that we search the best hyperparameters on the validation set.

Model	Paramater Name	Diginetica	Yoochoose1/64	Yoochoose1/4	Retailrocket
GRU4REC	Batch size	32	32	32	32*
	Embedding size	100	100	100	100*
	Learning rate	2e−1	2e−1	2e−1	2e−1*
NARM	Batch size	512	512	512	512*
	Embedding size	50	50	50	50*
	Learning rate	1e−3	1e−3	1e−3	1e−3*
STAMP	Batch size	512	512	512	256*
	Embedding size	100	100	100	256*
	Learning rate	3e−3	3e−3	3e−3	3e−3*
CSRM	Batch size	512	512	512	256*
	Embedding size	150	150	150	150*
	Memory size	512	512	512	256*
	Memory dim	100	100	100	100*
	Learning rate	5e−4	5e−4	5e−4	5e−4*
DSAN	Batch size	512	512*	512*	256*
	Embedding size	100	100*	100*	256*
	Normalize weight(w_k)	20	20*	20*	20*
	Dropout rate	0.5	0.2*	0.2*	0.2*
	Learning rate	1e−3	1e−3*	1e−3*	1e−3*
SR-GNN	Batch size	100	100	100	256*
	Embedding size	100	100	100	256*
	Number of GNN layers	1	1	1	1*
	Learning rate	1e−3	1e−3	1e−3	1e−3*
TAGNN	Batch size	100	100	100*	100*
	Embedding size	100	100	100*	256*
	Number of GNN layers	1	1	1*	1*
	Learning rate	1e−3	1e−3	1e−3*	1e−3*
COTREC	Batch size	100	256*	256*	256*
	Embedding size	100	256*	256*	256*
	Number of GNN layers	2	2*	2*	2*
	Learning rate	1e−3	1e−3*	1e−3*	1e−3*
	Self-supervise loss weight(β)	5e−3	5e−3*	5e−3*	5e−3*
	Divergence constraint loss weight(α)	5e−3	5e−3*	5e−3*	5e−3*
GCE-GNN	Batch size	100	256*	256*	256*
	Embedding size	100	256*	256*	256*
	Number of GNN layers	1	1*	1*	1*
	Dropout rate	0.2	0*	0*	0.2*
	Learning rate	1e−3	1e−3*	1e−3*	1e−3*

References

[1] J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, in: arXiv preprint arXiv:1607.06450 (2016).
[2] W. Chen, F. Cai, H. Chen, M. de Rijke, A dynamic co-attention network for session-based recommendation, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1461–1470.
[3] W. Chen, C. Fei, H. Chen, M.D. Rijke, Joint neural collaborative filtering for recommender systems, ACM Trans. Inf. Syst. (2019) 39:1–39:30.
[4] C. Feng, C. Shi, C. Liu, Q. Zhang, S. Hao, X. Jiang, Context-aware item attraction model for session-based recommendation, Expert Syst. Appl. 176 (2021) 114834.
[5] Q. Guo, X. Qiu, P. Liu, Y. Shao, X. Xue, Z. Zhang, Star-transformer, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019, pp. 1315–1325.
[6] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: Proceedings of the 43nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 639–648.
[7] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. (2004) 5–53.
[8] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based recommendations with recurrent neural networks, in: Proceedings of the 4th International Conference on Learning Representations, 2016.
[9] W. Kang, J. McAuley, Self-attentive sequential recommendation, in: Proceedings of the 2018 IEEE International Conference on Data Mining, 2018, pp. 197–206.
[10] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings of the 5th International Conference on Learning Representations, 2017.
[11] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, J. Ma, Neural attentive session-based recommendation, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1419–1428.
[12] Y. Li, D. Tarlow, M. Brockschmidt, R. Zemel, Gated graph sequence neural networks, in: Proceedings of the 4th International Conference on Learning Representations, 2016.
[13] Q. Liu, Y. Zeng, R. Mokhosi, H. Zhang, STAMP: short-term attention/memory priority model for session-based recommendation, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1831–1839.

[14] A. Luo, P. Zhao, Y. Liu, F. Zhuang, D. Wang, J. Xu, J. Fang, V.S. Sheng, Collaborative self-attention network for session-based recommendation, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, 2020, pp. 2591–2597.
[15] Z. Pan, F. Cai, W. Chen, H. Chen, M. de Rijke, Star graph neural networks for session-based recommendation, in: Proceedings of the 29th ACM International Conference on Information and Knowledge Management, 2020, pp. 1195–1204.
[16] Z. Pan, F. Cai, Y. Ling, M. de Rijke, Rethinking item importance in session-based recommendation, in: Proceedings of the 43nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 25–30.
[17] R. Qiu, J. Li, Z. Huang, H. Yin, 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, 2019, pp. 579–588.
[18] C. Raffel, D.P.W. Ellis, Feed-forward networks with attention can solve some long-term memory problems, CoRR abs/1512.08756 (2015).
[19] P. Ren, J. Li, Z. Chen, Z. Ren, J. Ma, M. de Rijke, Repeatnet: A repeat aware neural recommendation machine for session-based recommendation, in: Proceedings of The Thirty-Third AAAI Conference on Artificial Intelligence, 2019.
[20] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized markov chains for next-basket recommendation, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 811–820.
[21] B.M. Sarwar, G. Karypis, J.A. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the Tenth International World Wide Web Conference, 2001, pp. 285–295.
[22] A. See, P.J. Liu, C.D. Manning, Get to the point: Summarization with pointer-generator networks, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 2017, pp. 1073–1083.
[23] G. Shani, D. Heckerman, R.I. Brafman, An mdp-based recommender system, J. Mach. Learn. Res. (2005) 1265–1295.
[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. (2014) 1929–1958.
[25] R.K. Srivastava, K. Greff, J. Schmidhuber, Highway networks, in: arXiv preprint arXiv:1505.00387 (2015).
[26] Y.K. Tan, X. Xu, Y. Liu, Improved recurrent neural networks for session-based recommendations, in: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, 2016, pp. 17–22.
[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, Attention is all you need, in: Proceedings of the 31st Conference on Neural Information Processing Systems, 2017.
[28] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: Proceedings of the 6th International Conference on Learning Representations, 2018.
[29] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, M. de Rijke, A collaborative session-based recommendation approach with parallel memory modules, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 345–354.
[30] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, X. Cheng, Learning hierarchical representation model for nextbasket recommendation, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 403–412.
[31] S. Wang, L. Hu, Y. Wang, Q.Z. Sheng, M.A. Orgun, L. Cao, Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, pp. 3771–3777.
[32] Z. Wang, W. Wei, G. Cong, X.L. Li, X.L. Mao, M. Qiu, Global context enhanced graph neural networks for session-based recommendation, in: Proceedings of the 43nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 169–178.
[33] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, in: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, 2019, pp. 346–353.
[34] X. Xia, H. Yin, J. Yu, Y. Shao, L. Cui, Self-supervised graph co-training for session-based recommendation, in: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, 2021, pp. 2180–2190.
[35] C. Xu, P. Zhao, Y. Liu, V.S. Sheng, J. Xu, F. Zhuang, J. Fang, X. Zhou, Graph contextualized self-attention network for session-based recommendation, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, pp. 3940–3946.
[36] K. Xu, J. Ba, R. Kiros, K. Cho, A.C. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 2048–2057.
[37] F. Yu, Y. Zhu, Q. Liu, S. Wu, L. Wang, T. Tan, Tagnn: Target attentive graph neural networks for session-based recommendation, in: Proceedings of the 43nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1921–1924.
[38] J. Yuan, Z. Song, M. Sun, X. Wang, W.X. Zhao, Dual sparse attention network for session-based recommendation, in: Proceedings of the 35th AAAI Conference on Artificial Intelligence, 2021, pp. 4635–4643.
[39] J. Zhang, C. Ma, X. Mu, P. Zhao, C. Zhong, A. Ruhan, Recurrent convolutional neural network for session-based recommendation, Neurocomputing 437 (2021) 157–167.