Graph Co-Attentive Session-based Recommendation

ZHIQIANG PAN, FEI CAI, WANYU CHEN, and HONGHUI CHEN, National University of Defense Technology, China

Session-based recommendation aims to generate recommendations merely based on the ongoing session, which is a challenging task. Previous methods mainly focus on modeling the sequential signals or the transition relations between items in the current session using RNNs or GNNs to identify user's intent for recommendation. Such models generally ignore the dynamic connections between the local and global item transition patterns, although the global information is taken into consideration by exploiting the global-level pair-wise item transitions. Moreover, existing methods that mainly adopt the cross-entropy loss with softmax generally face a serious over-fitting problem, harming the recommendation accuracy. Thus, in this article, we propose a Graph Co-Attentive Recommendation Machine (GCARM) for session-based recommendation. In detail, we first design a Graph Co-Attention Network (GCAT) to consider the dynamic correlations between the local and global neighbors of each node during the information propagation. Then, the item-level dynamic connections between the output of the local and global graphs are modeled to generate the final item representations. After that, we produce the prediction scores and design a Max Cross-Entropy (MCE) loss to prevent over-fitting. Extensive experiments are conducted on three benchmark datasets, i.e., Diginetica, Gowalla, and Yoochoose. The experimental results show that GCARM can achieve the state-of-the-art performance in terms of Recall and MRR, especially on boosting the ranking of the target item.

CCS Concepts: • **Information systems → Recommender systems;**

Additional Key Words and Phrases: Session-based recommendation, graph co-attention networks, max cross-entropy

ACM Reference format:
Zhiqiang Pan, Fei Cai, Wanyu Chen, and Honghui Chen. 2021. Graph Co-Attentive Session-based Recommendation. *ACM Trans. Inf. Syst.* 40, 4, Article 67 (November 2021), 31 pages.
https://doi.org/10.1145/3486711

1 INTRODUCTION

Recommender systems play an important role in helping people filter out unrelated information and satisfying their personalized needs, which is an effective method to deal with the information explosion problem [1, 68]. Most general recommender systems predict users' preference based on

This work was partially supported by the National Natural Science Foundation of China under No. 61702526 and the Postgraduate Scientific Research Innovation Project of Hunan Province under No. CX20200055.
Authors' address: Z. Pan, F. Cai (corresponding author), W. Chen, and H. Chen, Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, 410073, China; emails: {panzhiqiang, caifei, wanyuchen, chenhonghui}@nudt.edu.cn.

1046-8188/2021/11-ART67 $15.00
https://doi.org/10.1145/3486711

their long-term historical interactions with items [14, 15]. For example, collaborative filtering (CF) utilizes the user-item interaction matrix to make recommendations [7, 53]. However, for situations where we only have users' current interactions (e.g., clicks on items within 24 hours), the general recommenders, e.g., CF-based methods, are not applicable. Thus, session-based recommendation (SBR) is proposed to capture user preference and make recommendations based on the ongoing session [17].

Considering that items in a session are organized chronologically, many sequential methods are proposed to model the user's dynamic interest migration [16, 17, 52]. For example, Rendle et al. [38] adopt the Markov Chains (MC) to capture the sequential signals between adjacent items; and Hidasi et al. [17] propose GRU4REC, which first adopts the Gated Recurrent Units (GRUs) to model the item sequence. In addition, the attention mechanism is also applied in SBR for distinguishing the items according to their different importances. For instance, Li et al. [23] apply an attention mechanism to emphasize a user's main intent based on GRU4REC. Recently, Graph Neural Networks (GNNs) are introduced into SBR to model the complex transitions among items. For instance, Wu et al. [57] and Qiu et al. [33] propose to transform each session into an individual graph for modeling the local pair-wise transition relations between items in the current session, and Qiu et al. [32] take all sessions in the training set into consideration to establish the global connections for modeling the global-level transitions between items. Furthermore, Wang et al. [55] propose to simultaneously consider the personalized and generalized behavior patterns that are respectively reflected in the local and global graphs for recommendation.

However, simply modeling the sequential orders or distinguishing the item importance [33, 57] is not enough to capture the complicated user behavior patterns in SBR. In addition, existing GNN-based methods either consider the local item transitions in each session [29, 33, 57] or model the global connections reflected in all sessions [32]. Such local and global item transitions are even simultaneously considered to select the global information [55]. The aforementioned approaches, on the one hand, fail to take the dynamic correlations between the local and global graphs into consideration. Some accidentally clicked items in the local graph and unrelated neighbors from the global graph may introduce the bias that leads to an inaccurate user preference. For instance, as shown in Figure 1, the interacted item *Clothes* and the introduced neighbor *MacBook Pro* both introduce the noise when modeling the user's main intent. On the other hand, the widely adopted cross-entropy loss with softmax in SBR regards all items except the target item as the negative samples for model optimization. It is unreasonable, since the user does not interact with some items may be due to that the recommender systems fail to expose them to the user rather than the user does not like them. Thus, simply decreasing the scores of all those items without interactions may result in an issue of model over-fitting, limiting the model's generalization ability.

To deal with the aforementioned issues, we propose the Graph Co-Attentive Recommendation Machine (GCARM) for session-based recommendation. Specifically, given the training set, we first establish the global connections of items (i.e., the global-level transitions) from all sessions. Then, given an ongoing session, we design a Graph Co-Attention Network (GCAT) to generate the accurate representations of the contained items. In detail, we transform the session into a local graph and extract the related correlations from the global connections to form a global graph. After that, we introduce the co-attention mechanism into the information propagation process and the item representation combination stage to eliminate the bias in the local and global graphs. Next, we attentively combine the item representations in current session to generate the user preference, which is utilized to produce the prediction scores. Finally, we propose a Max Cross-Entropy (MCE) loss that selects the items with the largest prediction scores (except the target item) as the negative samples for comparison in the model optimization to avoid over-fitting. Extensive experiments are conducted on three publicly available datasets, i.e., Diginetica, Gowalla, and Yoochoose. The exper-

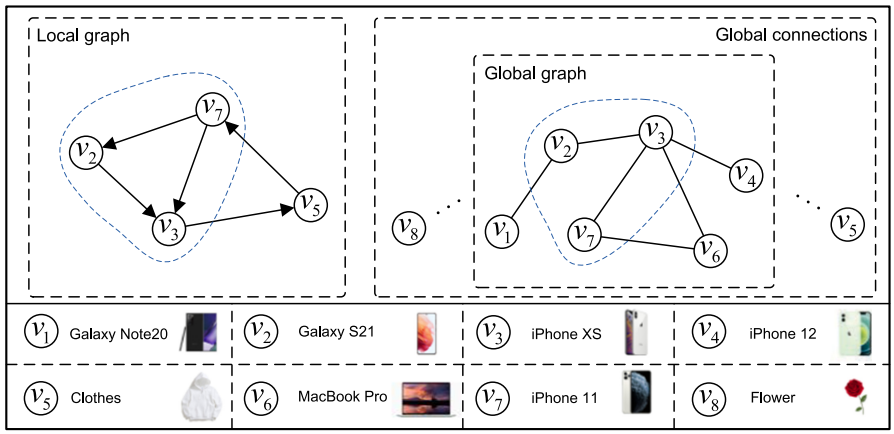


Fig. 1. An example of the local and global graphs for an ongoing session $S = \{v_7, v_2, v_3, v_5, v_7, v_3\}$. The arrows in the local graph (left part) indicate the item sequential orders, where *Clothes* is an unrelated item to user's main purpose (emphasized by the dotted blue curves). The global-level item transitions are presented in the global connections (right part), where *Clothes* and *Flower* are uncorrelated to the other items in the global connections. Moreover, the global graph is a subgraph extracted from the global connections, where *MacBook Pro* is an unrelated global neighbor.

imental results illustrate that our proposed GCARM can achieve the state-of-the-art performance in terms of both Recall and MRR.

The main contributions in this article can be summarized as follows:

(1) To the best of our knowledge, we are the first to consider the dynamic connections between the local and global transition relations among items when combining personalized and generalized behavior patterns for session-based recommendation;
(2) We propose a Graph Co-Attention Network (GCAT) that introduces a co-attention mechanism into the information propagation and item representation combination processes to obtain the accurate item representations;
(3) We design an Max Cross-Entropy (MCE) loss to deal with the over-fitting problem in model optimization for SBR and provide a theoretical analysis to prove its effectiveness;
(4) We conduct comprehensive experiments on three publicly available datasets to examine the performance of our proposal against the state-of-the-art baselines. The experimental results show that GCARM can beat the baselines in terms of both Recall and MRR.

2 RELATED WORK

In this section, we first review the related work about SBR from two aspects, i.e., the individual SBR that merely relies on the current session to make recommendations and the collaborative SBR that incorporates information from other sessions for modeling the user preference. Then, we summarize the previous work on negative sampling in recommender systems considering that the designed MCE loss can be regarded as a negative sampling strategy.

2.1 Individual SBR

As for the individual SBR recommenders, the user preference is usually generated by considering the sequential order [16, 17, 38], the importance of different items [21, 26, 31], or taking the pairwise item transitions into consideration [29, 33, 57] in the ongoing session.

2.1.1 Sequential Methods. Traditional methods like the Markov Chains are adopted to model the sequential relations between adjacent items [38, 41, 50]. Neural models such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have also been widely applied to the sequential signal modeling in SBR [3, 34, 42, 63]. For example, Hidasi et al. [17] propose GRU4REC to capture the dynamic interest in the ongoing session using the GRUs. Li et al. [23] design a hybrid session encoder that utilizes the GRUs to model the sequential signals and emphasizes user's main intent by an attention mechanism. Moreover, Hidasi and Karatzoglou [16] propose to solve the gradient vanishing problem by improving the loss functions, and Bogina and Kuflik [2] propose to take the dwell time between the interactions into consideration for boosting the recommendations. In addition, Song et al. [42] propose a hierarchical memory network that utilizes the densely connected CNNs to simultaneously capture user's item-level and feature-level preferences.

2.1.2 Attention-based Methods. The item importance plays an important role in capturing the user intent by distinguishing different items [21, 45, 62, 67]. Specifically, the attention mechanism can be applied together with other methods such as RNNs [6, 23, 49] or utilized individually [26, 31]. For instance, Liu et al. [26] propose to explicitly consider the recent interest together with the long-term preference for user intent detection using the memory networks [44, 56], where the attention mechanism is applied to assign different importance scores for items. Moreover, Pan et al. [31] propose to generate the importance score of each item by comparing it with the other items in current session, thus to avoid the bias brought by the unrelated items. In addition, Ren et al. [35] introduce a repeat-explore mechanism that utilizes the attention mechanism to consider the repeat consumption behaviors of users.

2.1.3 GNN-based Methods. To take the complex transition relations between items in current session into consideration, GNN-based methods are proposed [5, 29, 33, 57]. For instance, Wu et al. [57] propose SR-GNN, which transforms each session into a session graph and adopts the gated graph neural network (GGNN) for item representation learning. Yu et al. [64] improve SR-GNN by introducing a target-aware attention to adaptively detect user's interest. Moreover, Xu et al. [59] take the long-range dependencies between items into consideration in the session aggregation by the self-attention mechanism [46]. Pan et al. [29] exploit the long-distance information between items in the information propagation stage by a star graph neural network (SGNN). In addition, Chen and Wong [5] focus on solving the information loss in GNNs for SBR by preserving the edge order and introducing the shortcut connections. Furthermore, Qiu et al. [33] design a weighted graph attention network (WGAT) for attentively computing the information flow in propagation.

However, the Markov Chains, RNN and CNN based methods mainly model the user preference by relying on the sequential signals; the attention-based approaches focus on the item importance. Such approaches have been proved not effective to fully consider the complicated transition relationship between items [33, 57]. For the individual GNN-based methods, merely limited information contained in current session graph can be utilized for user preference generation, which may be disturbed by the contained unrelated items [29]. Differently, in our proposed GCARM, we can incorporate the related generalized behavior patterns for enhancing the item representation learning in current session by the mean of extracting the global graph from the global connections for the current session.

2.2 Collaborative SBR

Since merely limited information can be used for detecting the user intent in current session for SBR, many collaborative methods are proposed to incorporate related information from other sessions for user preference modeling based on RNNs [18, 30, 49], SAT [27], and GNNs [32, 55].

The external information in the collaborative models for SBR can be introduced mainly by two ways, i.e., incorporating the neighbor sessions and exploiting the global item transitions.

2.2.1 Neighbor Incorporation-based Methods. In this field, neighbor sessions are first selected and then combined with the current session to generate the final user preference, where the neighbor detection can rely on the k-nearest neighbor (KNN) method [11, 18, 25, 40], the memory networks [49], or the user intent guidance [30]. Moreover, the representation of the current session and each neighbor session can be generated by the RNNs [18, 30, 49], the self-attention mechanism (SAT) [27], and so on. Specifically, Jannach and Ludewig [18] propose to linearly combine the KNN approach with the RNN-based encoder GRU4REC and significantly improve the performance of GRU4REC. Moreover, Wang et al. [49] design a parallel memory module that consists of an inner and an outer memory encoder for modeling the current session and exploiting the collaborative information, respectively. In addition, to improve the accuracy of the neighbor session selection, Pan et al. [30] propose to introduce the user intent into the neighbor detection.

2.2.2 Global Transitions-aware Methods. However, collaborative information can also be introduced by incorporating the global-level item transitions from other sessions into the graph construction for the current session [32, 55]. Specifically, the global item transitions contained in all sessions are established for representing the generalized user behavior patterns, which are then introduced into the personalized preference modeling in the ongoing session. For example, Qiu et al. [32] propose to establish the global connections reflected in all sessions and extract the related part to the ongoing session for the information propagation and graph pooling to generate the user preference. Furthermore, Wang et al. [55] propose to separately construct the local and global graphs for each session and utilize user's current interest to guide the information propagation in the global graph. After that, the item vectors in two graphs are combined as the final item representations.

However, considering the inaccuracy of the neighbor detection, the neighbor incorporation-based methods easily introduce the bias, as the incorporated neighbors may be unrelated [30], limiting the recommendation performance. As for the global transitions-aware methods, the existing models merely rely on the global connections between items [32] or simply extract the related information from the global connections based on the current session [55], failing to consider the dynamic connections between the local and global graphs. In addition, such approaches are also easily disturbed by the bias brought by the unrelated items in both local and global graphs. Thus, in this article, we design GCAT which introduces a co-attention mechanism into the information propagation and item representation combination to eliminate the bias by considering the dynamic connections between local and global graphs.

2.3 Negative Sampling for Recommendation

Negative sampling is widely applied in the recommender systems to solve the one-class problem [28] in scenarios with only implicit feedback such as clicks and purchases, which aims to select a subset of the unobserved items as negative samples for model optimization. Static samplers adopt a fixed probability distribution on items to obtain the negative samples, which will not be changed in the training phase and are not adaptive to different users [4, 37]. For example, Rendle et al. [37] obtain the negative samples using the random distribution on items, and Chen et al. [4] further propose a popularity-biased negative sampling (PNS) approach to select the negative samples according to the item popularity. In addition, the generative adversarial network (GAN) [12] is adopted to obtain the adversarial negative samples [48, 51]. For instance, Wang et al. [48] improve the quality of negative samples by introducing a generator to play a minimax game with the recommender. Recently, GNN-based samplers are proposed by exploiting the graph

structure [54, 61]. For example, Wang et al. [54] propose to select high-quality negative samples by exploring multi-hop paths on the knowledge graph using the reinforcement learning (RL) method. Moreover, the hard negative sampling methods such as DNS [69], Adaptive Oversampling [36], and LambdaFM [65] are proposed, which select the unobserved items with the highest scores in prediction as the negative samples in the Bayesian Personalized Ranking (BPR) loss for learning the model parameters. In addition, the one most closely related to our work is the softmax-based samplers [9, 17]. For instance, Covington et al. [9] select the negative samples corrected by an importance weighting [19], which are then inputted together with the ground truth into the softmax function, and Hidasi et al. [17] propose to directly adopt the target item of the other training samples in the current mini-batch as the negative samples.

However, most sampling methods are generally designed for the BPR loss, neglecting the importance of improving the quality of negative samples in the cross-entropy loss with softmax. Moreover, the softmax-based samplers fail to provide enough gradient for increasing the target item score due to the selected low-quality negative samples. Compared with existing samplers, our proposed MCE loss can not only obtain the high-quality negative samples for model optimization, but effectively prevent the over-fitting problem by avoiding taking items with low prediction scores for comparison.

3 PRELIMINARIES

In this section, we first define the task of the session-based recommendation. Then, we introduce the attentive information propagation in Graph Attention Network (GAT).

3.1 Problem Statement

Assuming the item set is $V = \{v_1, v_2, \ldots, v_{|V|}\}$, where v_i is an item and $|V|$ is the size of the item set. Given an ongoing session denoted as $S = \{v_1, v_2, \ldots, v_n\}$, the goal of session-based recommendation is to predict the user's interaction at next timestamp, i.e., v_{n+1}. Specifically, given a user's interacted items in a session, the recommender first obtains the user's preference and then generates the prediction scores on all candidate items in V. After that, items with the K largest scores will constitute the final recommendation list to the user.

We list the main notations used in this article in Table 1.

3.2 Attentive Information Propagation

Given a node x_i where its neighbors are included in a set denoted as $N(x_i)$, the graph neural network (GNN) updates the representation of x_i by propagating information from its neighbors $N(x_i)$ to the node x_i. Here, we adopt the GAT [47] as an example, since it can dynamically determine the weights of different neighbors in the information propagation. Other GNNs such as GGNN [24] and GCN [22] can also be adopted to replace GAT in our framework.

Specifically, we first utilize a self-attention mechanism [46] to determine the importance of different neighbors in the information propagation as follows:

$$e_{ij} = \text{LeakyRelu}\left(\mathbf{W}_{att}^{\mathsf{T}}[\mathbf{W}_q\mathbf{x}_i; \mathbf{W}_k\mathbf{x}_j]\right), \tag{1}$$

where $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ are the d-dimensional node vectors of x_i and x_j, respectively, and $x_j \in \{x_i, N(x_i)\}$. Here, we add a self connection to propagate information from node x_i for preserving the specific characteristics of x_i. $[;]$ indicates the concatenation operation, $\mathbf{W}_q, \mathbf{W}_k \in \mathbb{R}^{d\times d}$ and $\mathbf{W}_{att} \in \mathbb{R}^{2d}$ are the trainable parameters, and LeakyRelu indicates the non-linear activation function.

Table 1. Main Notations Used in This Article

Notation	Description
U	all sessions in the training set
V	the item set consisting of all candidate items
$S = \{v_1, v_2, \ldots, v_n\}$	a session containing n chronological items
$x_i, N(x_i)$	a node and its corresponding neighbors in the graph neural network (GNN)
e_{ij}, α_{ij}	the importance of neighbor x_j for node x_i before and after normalization
$G = (V, E)$	the global connections established by the item transitions in all sessions
$G_l = (V_l, E_l)$	the constructed local graph for session S
$G_g = \{V_g, E_g\}$	the extracted global graph for session S from G
V_s, V_S^g	the unique items in session S and the incorporated global neighbors
m	the number of the unique items in session S
K	the length of the recommendation list for the user
N	the number of the global neighbors of each item in G
H	the hop number for introducing multi hops of neighbors
M	the number of incorporated global neighbors in G_g for each item in S
$N^l(x_i), N^g(x_i)$	the neighbor of node x_i in the local and global graphs
$\tilde{\mathbf{C}}^i$	the generated neighbor-level co-attention matrix for node x_i
$\mathbf{X}^c_{N^l(x_i)}, \mathbf{X}^c_{N^g(x_i)}$	the co-attentive representation of the local and global neighbors of x_i
$\tilde{\mathbf{x}}_i^{IL}, \tilde{\mathbf{x}}_i^{IG}$	the individual local and global representations of x_i
$\tilde{\mathbf{x}}_i^{CL}, \tilde{\mathbf{x}}_i^{CG}$	the neighbor-level co-attentive local and global representations of x_i
$\tilde{\mathbf{x}}_i^{L}, \tilde{\mathbf{x}}_i^{G}$	the combined local and global representation of node x_i in GCAT
$\hat{\mathbf{C}}$	the item-level co-attention matrix
$\hat{\mathbf{x}}_i^{L}, \hat{\mathbf{x}}_i^{G}$	the item-level co-attentive local and global representations of x_i
$\mathbf{Z} = \{\mathbf{z}_i, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$	the combined representation of items in session S produced by GCAT
$\psi^{IL}, \psi^{IG}, \psi^{CL}, \psi^{CG}$	the trainable parameters in the attentive information propagation
$\mathbf{W}_{\tilde{c}}, \mathbf{W}_{\hat{c}}$	the trainable parameters in the neighbor- and item-level co-attentions
$\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$	the trainable parameters in the item representation combination
$\mathbf{W}_4, \mathbf{W}_5, \mathbf{W}_6, \mathbf{W}_7$	the trainable parameters in the user preference generation
$\mathbf{s}_r, \mathbf{s}_l, \mathbf{s}_h$	the recent, long-term, and hybrid preference in the ongoing session
ϵ_i, β_i	the importance of v_i in session aggregation before and after normalization
$\tilde{\mathbf{y}}_i$	the initial prediction score of item v_i
v^+	the target item of session S
$V \backslash v^+$	all the candidate items except the target item
V^-	the sampled negative items after max sampling in MCE
$\hat{\mathbf{y}}_i$	the prediction score of item v_i after normalization

Then, we normalize the importance scores using a softmax layer:

$$\alpha_{ij} = \text{Softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{x_k \in \{x_i, N(x_i)\}} \exp(e_{ik})}. \tag{2}$$

After that, we linearly combine the node vectors of the neighbors weighted by the generated attention scores to obtain the updated representation of node x_i:

$$\tilde{\mathbf{x}}_i = \text{Update}(\mathbf{x}_i, \mathbf{X}_{N(x_i)}) = \sigma\left(\sum_{x_j \in \{x_i, N(x_i)\}} \alpha_{ij} \mathbf{W}_v \mathbf{x}_j\right), \tag{3}$$

where $\mathbf{x}_i$ is the vector of node x_i and $\mathbf{x}_j \in \mathbf{X}_{\{x_i, N(x_i)\}}$ is the representation of the neighbor $x_j \in \{x_i, N(x_i)\}$. $\tilde{\mathbf{x}}_i$ is the updated vector of x_i, $\mathbf{W}_v \in \mathbb{R}^{d \times d}$ is the learnable parameters, and σ denotes the sigmoid function. In summary, the learnable parameters in the attentive information

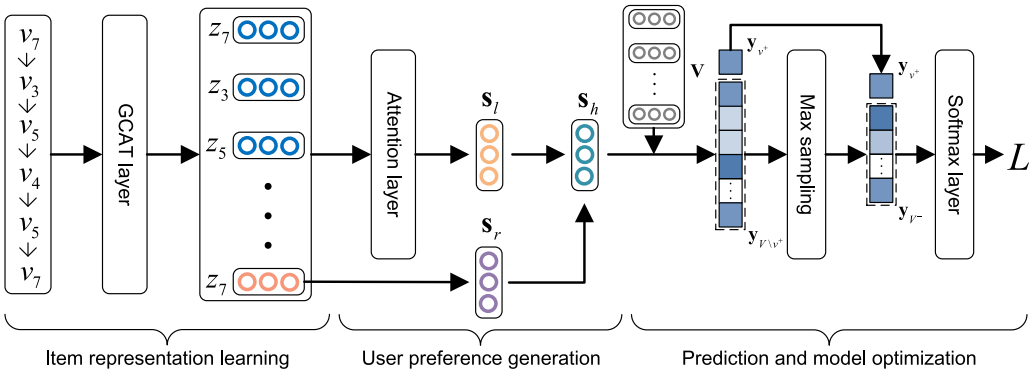


Fig. 2. The workflow of GCARM.

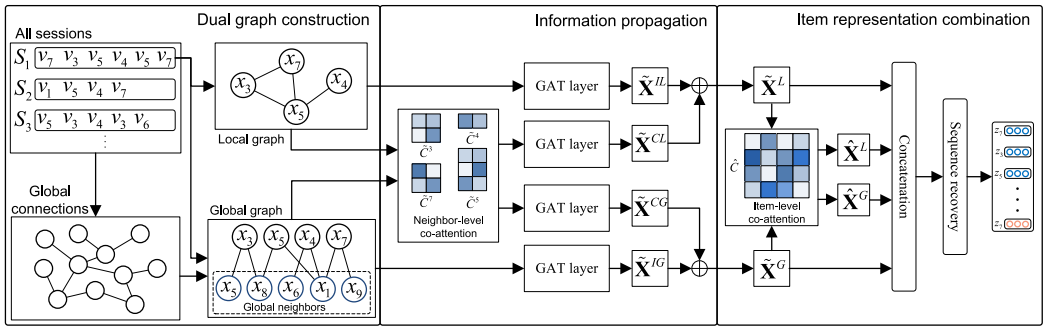


Fig. 3. The workflow of GCAT.

propagation process are marked as $\psi = \{\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v, \mathbf{W}_{att}\}$, which vary in different information propagation channels detailed in Section 4.1.2 in our proposal.

4 APPROACH

We detail our proposed Graph Co-Attentive Recommendation Machine (GCARM) in this section, which contains three main components, i.e., the graph co-attentive item representation learning, the user preference generation, the prediction and model optimization. The workflow of GCARM is shown in Figure 2. Specifically, given an ongoing session, we first learn the item representations through the proposed GCAT; then, we generate the hybrid user preference by an attention mechanism; finally, we make predictions and optimize the model using the designed MCE loss.

4.1 Item Representation Learning

To learn accurate item representations, we design a Graph Co-Attention Network (GCAT) to consider both the local and global transitions between items in the corresponding local and global graphs, as well as the dynamic connections between them. The workflow of the GCAT is plotted in Figure 3, which consists of three main components, i.e., the dual graph construction, the information propagation, and the item representation combination.

4.1.1 Dual Graph Construction. We construct the dual session graph to model the pair-wise transition relationship between items based on the global connections, where the dual graph

includes a local graph that indicates the personalized behavior patterns in the ongoing session and a global graph that reflects the generalized behavior patterns of all users.

Global connections. First, based on all given sessions, to incorporate the generalized user behavior patterns into consideration, we establish the global connections of all items in sessions in the training set. Let $G = \{V, E\}$ denote the graph that contains the global connections, where V indicates all items and E denotes all the transitions among items in the training set, respectively. An edge $(v_i, v_j) \in E$ indicates that items v_i and v_j are clicked adjacently in a session. Moreover, considering that some items may have many connected neighbors in G and some neighbors may be connected because of several accidental interactions, here, we conduct a max sampling according to the weights of edges to select the most related N neighbors as the final neighbors of each item. By the max sampling, the loosely connected neighbors of each item can be filtered to avoid introducing bias. In addition, multi-hops of global neighbors can be incorporated to enhance the information propagation process based on the global connections in G. Finally, the constructed global connections can represent the generalized behavior patterns of users reflected in all sessions.

Then, we construct the so-called dual graph, i.e., the local graph and the global graph. Here, we construct the undirected graphs for both local and global graphs, since additional trainable parameters need to be introduced for distinguishing different types of edges if a directed graph is involved [29, 55, 57], which decreases the training efficiency and easily leads the model to over-fitting.

Local graph. Given an ongoing session containing n items as $S = \{v_1, v_2, \ldots, v_n\}$, we construct an undirected local session graph to represent the item transitions in S as $G_l = (V_l, E_l)$, where $V_l = \{x_1, x_2, \ldots, x_m\}$[1] indicates the contained nodes and E_l denotes the edges. An edge $(x_i, x_j) \in E_l$ indicates that items x_i and x_j are adjacently clicked in the current session. Note that we only take the pair-wise co-occurrence of two items into consideration here, ignoring the weights of different edges in the graph reflected by the co-occurrence. Instead, we learn the weights of different neighbors in information propagation dynamically using GAT [47].

Global graph. After establishing the global connections, for each session S, we extract a subgraph $G_g = \{V_g, E_g\}$ from G as the global graph of S to incorporate the related generalized behavior patterns into the current user preference modeling. Here, $V_g = \{V_S, N_S^g\}$ contains a set of unique items V_S in session S and a pool of related global neighbors $N_S^g \in \mathbb{R}^{|V_S| \times M}$ extracted from the global connections in G, where M is the number of the incorporated global neighbors from the global graph for each node in V_S. Each edge $(x_i, x_j) \in E_g$ indicates that items x_i and x_j are connected in the global graph.

To clearly illustrate the global graph construction, we present the extraction procedure of the subgraph G_g from the global connections in G in Algorithm 1. Specifically, given a session S, we first get the unique items in S as $V_S = \{x_1, x_2, \ldots, x_m\}$ in line 1. Then for each item x_i, we first obtain its global neighbors of 1-hop, i.e., $N(x_i)$ in line 3, and meanwhile regard them as the latest hop neighbors of x_i in line 4. After that, we explore the multi-hop global neighbors from line 5 to 8. Specifically, we utilize the neighbors of the latest hop as the index to obtain the neighbors of the next hop in line 6, which are combined with the current global neighbors as the extended neighbors of x_i in line 7. Next, we combine V_S with the neighbors N_S^g as the nodes in G_g in line 10, and extract the related edges from the global connections G in line 11, i.e., connect each item

[1]Note that the node number m in G_l may be smaller than the item number n in S, since some items may be repeatedly clicked in the session [33, 35].

ALGORITHM 1: Procedure of Global Graph Extraction

Input: $S = \{v_1, v_2, \ldots, v_n\}$: The sequential items in an ongoing session;
$G = \{V, E\}$: The global connections constructed from all sessions;
H: The hop number for incorporating the global neighbors in G;
Output: $G_g = \{V_g, E_g\}$: The extracted global graph for session S;

$V_S = \{x_1, x_2, \ldots, x_m\} \leftarrow \text{Unique}(\{v_1, v_2, \ldots, v_n\})$;
for i in range(m) **do**
  $N^g(x_i) = N(x_i)$;
  $N^g_{last}(x_i) = N(x_i)$;
  for h in range(H-1) **do**
    $N^g_{last}(x_i) = N(N^g_{last}(x_i))$;
    $N^g(x_i) = N^g(x_i) + N^g_{last}(x_i)$;
  end for
end for
$V_g = V_s + N^g_S$;
$E_g = \text{Extract}(E, V_s, N^g_S)$;
return $G_g = \{V_g, E_g\}$.

$v_i \in V_S$ with its corresponding global neighbors $N^g(x_i) \in N^g_S$ to form the edges in G_g. Finally, we return the extracted global graph as $G_g = \{V_g, E_g\}$.

4.1.2 Information Propagation in GCAT. After constructing the local graph and global graph for the current session, we perform information propagation to learn the accurate representations of items. On the one hand, we individually propagate information on the local and global graphs. On the other hand, we take the dynamic connections between the local and global neighbors of each unique item in session S into consideration through a co-attentive information propagation.

First, we embed each node x_i in the local and global graphs to obtain its initial representation $\mathbf{x}_i$ as follows:

$$\mathbf{x}_i = \text{Embedding}(x_i), \tag{4}$$

where Embedding denotes the embedding layer, $\mathbf{x}_i \in \mathbb{R}^d$ is the embedding of x_i, and d is the embedding dimension.

Propagation on local graph. After transforming the user interactions in the ongoing session into a local graph $G_l = \{V_l, E_l\}$, we propagate information on G_l to model the local transition relationship between different items. Specifically, given a node $x_i \in V_l$ and its neighbors $N^l(x_i)$ in the local graph G_l, we can update the representation of x_i as:

$$\tilde{\mathbf{x}}^{IL}_i = \text{Update}^{IL}\left(\mathbf{x}_i, \mathbf{X}_{N^l(x_i)}\right), \tag{5}$$

where $\tilde{\mathbf{x}}^{IL}_i \in \mathbb{R}^d$ indicates the *individual local* representation of node x_i and Update^{IL} is the function described in Equation (3) that indicates the information propagation operation.

Propagation on global graph. To explore the generalized user behavior patterns for enhancing the personalized user preference modeling, we conduct information propagation on the global graph $G_g = \{V_g, E_g\}$ to obtain the updated global-level representation of each item in S. Specifically, for each unique item x_i with its neighbors $N^g(x_i)$ in the global graph G_g, we update the representation of x_i as follows:

$$\tilde{\mathbf{x}}^{IG}_i = \text{Update}^{IG}\left(\mathbf{x}_i, \mathbf{X}_{N^g(x_i)}\right), \tag{6}$$

where $\tilde{\mathbf{x}}_i^{IG} \in \mathbb{R}^d$ indicates the *individual global* representation of node x_i and Update^{IG} is the information propagation function. Here, the Update functions with different superscripts have different learnable parameters.

Co-attentive propagation. Through the information propagation on the local and global graphs, we can obtain the individual representations of the nodes in the corresponding graphs, which can reflect the personalized and the generalized behavior patterns, respectively. However, simply combining them like Reference [55] cannot fully take the advantage of the contained information in both representations. Specifically, some unrelated items that are interacted accidentally in the local graph and the neighbors that do not conform to the current user intent in the global graph will both introduce the noise in the user preference modeling. Thus, we propose to conduct a co-attentive propagation to take the dynamic correlations between the local and global neighbors of each item in session S into consideration.

Specifically, for each unique item x_i in the current session, we first obtain its neighbors in the local graph and the global graph, denoted as $N^l(x_i) = \{x_1, x_2, \ldots, x_{N_l}\}$ and $N^g(x_i) = \{x_1, x_2, \ldots, x_{N_g}\}$, respectively, where N_l and N_g indicate the number of neighbors in $N^l(x_i)$ and $N^g(x_i)$. Then, we calculate the neighbor-level co-attention matrix between the local neighbors $N^l(x_i)$ and the global neighbors $N^g(x_i)$ as follows:

$$\tilde{\mathbf{C}}^i = \tanh\left(\mathbf{X}_{N^l(x_i)}{}^{\top} \mathbf{W}_{\tilde{c}} \mathbf{X}_{N^g(x_i)}\right), \tag{7}$$

where $\mathbf{X}_{N^l(x_i)} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_l}\}$ and $\mathbf{X}_{N^g(x_i)} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_g}\}$ are the embeddings of the neighbors in $N^l(x_i)$ and $N^g(x_i)$, respectively, and $\mathbf{W}_{\tilde{c}} \in \mathbb{R}^{d \times d}$ is the trainable parameters. $\tilde{\mathbf{C}}^i \in \mathbb{R}^{N_l \times N_g}$ contains the neighbor-level co-attention weights. The j-row in $\tilde{\mathbf{C}}^i$, i.e., $\tilde{\mathbf{C}}^i_j \in \mathbb{R}^{N_g}$, indicates the similarities of the neighbor x_j of node x_i in the local graph to all neighbors of x_i in the global graph. Correspondingly, the k-column in $\tilde{\mathbf{C}}^i$, i.e., $\tilde{\mathbf{C}}^{i\top}_k \in \mathbb{R}^{N_l}$ denotes the similarities of the neighbor x_k of node x_i in the global graph to all neighbors of x_i in the local graph.

Then, we incorporate the neighbor-level global information from the global graph to the local graph and perform the information propagation to update the representation of node x_i in the local graph as follows:

$$\begin{aligned} \tilde{\mathbf{x}}_i^{CL} &= \text{Update}^{CL}\left(\mathbf{x}_i, \mathbf{X}^c_{N^l(x_i)}\right), \\ \mathbf{X}^c_{N^l(x_i)} &= \text{Softmax}(\tilde{\mathbf{C}}^i)\mathbf{X}_{N^g(x_i)}, \end{aligned} \tag{8}$$

where $\tilde{\mathbf{x}}_i^{CL} \in \mathbb{R}^d$ is the *co-attentive local* representation of node x_i.

Similarly, we input the neighbor-level local information extracted from the local graph to the global graph to obtain the updated representation of node x_i in the global graph as follows:

$$\begin{aligned} \tilde{\mathbf{x}}_i^{CG} &= \text{Update}^{CG}\left(\mathbf{x}_i, \mathbf{X}^c_{N^g(x_i)}\right), \\ \mathbf{X}^c_{N^g(x_i)} &= \text{Softmax}(\tilde{\mathbf{C}}^{i\top})\mathbf{X}_{N^l(x_i)}, \end{aligned} \tag{9}$$

where $\tilde{\mathbf{x}}_i^{CG} \in \mathbb{R}^d$ is the *co-attentive global* representation of x_i.

Note that, here, we adopt different trainable parameters for propagating different source of information to update the item representations, i.e., ψ^{IL}, ψ^{IG}, ψ^{CL}, and ψ^{CG}, which correspond to the functions Update^{IL}, Update^{IG}, Update^{CL}, and Update^{CG}, respectively.

4.1.3 Item Representation Combination. After propagating information on GCAT, we combine the *individual local* and *co-attentive local* representations of node x_i as the final local representation

of x_i, which reflects the personalized user behavior patterns in the ongoing session as follows:

$$\tilde{\mathbf{x}}_i^L = \mathbf{W}_1 \left[\tilde{\mathbf{x}}_i^{IL}; \tilde{\mathbf{x}}_i^{CL}\right], \tag{10}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times 2d}$ is the trainable parameters.

Similarly, we combine the *individual global* and *co-attentive global* representations of node x_i as the final global representation of x_i to represent the generalized user behavior patterns of all users:

$$\tilde{\mathbf{x}}_i^G = \mathbf{W}_2 \left[\tilde{\mathbf{x}}_i^{IG}; \tilde{\mathbf{x}}_i^{CG}\right], \tag{11}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d \times 2d}$ indicates the learnable parameters.

After obtaining the representations of the unique items in session S in the local and global graphs, i.e., $\tilde{\mathbf{X}}^L = \{\tilde{\mathbf{x}}_1^L, \tilde{\mathbf{x}}_2^L, \ldots, \tilde{\mathbf{x}}_m^L\}$ and $\tilde{\mathbf{X}}^G = \{\tilde{\mathbf{x}}_1^G, \tilde{\mathbf{x}}_2^G, \ldots, \tilde{\mathbf{x}}_m^G\}$, to consider the item-level connections between the unique items in S in the local and global graphs, we introduce the item-level co-attention mechanism. Specifically, we calculate the co-attention matrix between $\tilde{\mathbf{X}}^L$ and $\tilde{\mathbf{X}}^G$ as:

$$\hat{\mathbf{C}} = \tanh\left(\tilde{\mathbf{X}}^{L^\top} \mathbf{W}_{\hat{c}} \tilde{\mathbf{X}}^G\right), \tag{12}$$

where $\mathbf{W}_{\hat{c}} \in \mathbb{R}^{d \times d}$ is the learnable parameters, and $\hat{\mathbf{C}} \in \mathbb{R}^{m \times m}$ contains the item-level co-attention weights. Moreover, the i-row in $\hat{\mathbf{C}}$, i.e., $\hat{\mathbf{C}}_i \in \mathbb{R}^m$, indicates the similarities of item x_i in the local graph to the items in the global graph (excluding the global neighbors). Correspondingly, the j-column in $\hat{\mathbf{C}}$, i.e., $\hat{\mathbf{C}}_j^\top \in \mathbb{R}^m$, denotes the similarities of the item x_j in the global graph to all items in the local graph.

Then, we incorporate the item-level global information to the local graph according to the co-attention scores:

$$\hat{\mathbf{x}}_i^L = \text{Softmax}(\hat{\mathbf{C}}_i)\tilde{\mathbf{X}}^G, \tag{13}$$

and vice versa:

$$\hat{\mathbf{x}}_i^G = \text{Softmax}\left(\hat{\mathbf{C}}_i^\top\right)\tilde{\mathbf{X}}^L. \tag{14}$$

After that, for each unique item x_i in session S, we combine the output of the local and global graphs, i.e., $\tilde{\mathbf{x}}_i^L$ and $\tilde{\mathbf{x}}_i^G$, as well as the *item-level co-attentive* item representations $\hat{\mathbf{x}}_i^L$ in the local graph and $\hat{\mathbf{x}}_i^G$ in the global graph, to generate the final representation of item x_i:

$$\mathbf{z}_i = \mathbf{W}_3 \left[\tilde{\mathbf{x}}_i^L; \tilde{\mathbf{x}}_i^G; \hat{\mathbf{x}}_i^L; \hat{\mathbf{x}}_i^G\right], \tag{15}$$

where $\mathbf{z}_i \in \mathbb{R}^d$ denotes the final representation of x_i outputted by GCAT, and $\mathbf{W}_3 \in \mathbb{R}^{d \times 4d}$ is the trainable parameters that transform the concatenated vector from $\mathbb{R}^{4d}$ to $\mathbb{R}^d$.

Finally, after obtaining the representations of the unique items in session S, i.e., $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\}$, we recover the item sequence from the graph accordingly and obtain the representations of items in session as $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$, where n is the item number in session S.

We detail the procedure of GCAT in Algorithm 2. Specifically, given a session S and the global connections G, we first construct the local and global graphs in lines 1 and 2, respectively, then we obtain the unique items in S in line 3. After that, for each unique item, we update its representation from line 4 to 13. Specifically, we first embed each node in line 5 and then generate the individual local and global representations of node x_i in lines 6 and 7, respectively. After that, we obtain the neighbor-level co-attentive local and global representations of node x_i in lines 9 and 10 based on the co-attention matrix generated in line 8. Then, we produce the final local and global node vectors of x_i in lines 11 and 12, respectively. After that, we calculate the item-level co-attention matrix between the items in the local and global graphs in line 14. Then for each item, we generate the item-level co-attentive local and global item representations in line 16, which are combined

ALGORITHM 2: Procedure of GCAT

Input: $S = \{v_1, v_2, \ldots, v_n\}$: The sequential items in an ongoing session;
$G = \{V, E\}$: The global connections constructed from all sessions;
Output: $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$: Representation of the items in the session;

1: $G_l = \{V_l, E_l\} \leftarrow$ LocalGraphConstruct(S);
2: $G_g = \{V_g, E_g\} \leftarrow$ GlobalGraphConstruct(S, G);
3: $\{x_1, x_2, \ldots, x_m\} \leftarrow$ Unique($\{v_1, v_2, \ldots, v_n\}$);
4: **for** i in range(m) **do**
5: $\mathbf{x}_i$ = Embedding(x_i) $\quad \forall x_i \in \{V_l, V_g\}$ based on Equation (4);
6: $\tilde{\mathbf{x}}_i^{IL} = \text{Update}^{IL}(\mathbf{x}_i, \mathbf{X}_{N^l(x_i)})$ based on Equation (5);
7: $\tilde{\mathbf{x}}_i^{IG} = \text{Update}^{IG}(\mathbf{x}_i, \mathbf{X}_{N^g(x_i)})$ based on Equation (6);
8: $\tilde{\mathbf{C}}^i = \text{NeighborCoATT}(\mathbf{X}_{N^l(x_i)}, \mathbf{X}_{N^g(x_i)}))$ based on Equation (7);
9: $\tilde{\mathbf{x}}_i^{CL} = \text{Update}^{CL}(\mathbf{x}_i, \tilde{\mathbf{C}}^i, \mathbf{X}_{N^g(x_i)})$ based on Equation (8);
10: $\tilde{\mathbf{x}}_i^{CG} = \text{Update}^{CG}(\mathbf{x}_i, \tilde{\mathbf{C}}^i, \mathbf{X}_{N^l(x_i)})$ based on Equation (9);
11: $\tilde{\mathbf{x}}_i^{L} = \text{Concatenate}(\tilde{\mathbf{x}}_i^{IL}, \tilde{\mathbf{x}}_i^{CL})$ based on Equation (10);
12: $\tilde{\mathbf{x}}_i^{G} = \text{Concatenate}(\tilde{\mathbf{x}}_i^{IG}, \tilde{\mathbf{x}}_i^{CG})$ based on Equation (11);
13: **end for**
14: $\hat{\mathbf{C}} = \text{ItemCoATT}(\tilde{\mathbf{X}}^L, \tilde{\mathbf{X}}^G)$ based on Equation (12);
15: **for** i in range(m) **do**
16: $\hat{\mathbf{x}}_i^L, \hat{\mathbf{x}}_i^G = \text{LinearCombine}(\hat{\mathbf{C}}, \tilde{\mathbf{X}}^L, \tilde{\mathbf{X}}^G)$ based on Equation (13) and Equation (14);
17: $\mathbf{z}_i = \text{Concatenate}(\tilde{\mathbf{x}}_i^L, \tilde{\mathbf{x}}_i^G, \hat{\mathbf{x}}_i^L, \hat{\mathbf{x}}_i^L)$ based on Equation (15);
18: **end for**
19: $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\} \leftarrow \text{SequenceRecovery}(\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\})$;
20: **return** $\mathbf{Z}$.

with the individual representations as the final node vector of x_i in line 17. Finally, we recover the graph into the item sequence in line 19, which is returned as the output of GCAT.

4.2 User Preference Generation

After obtaining the representations of items in the ongoing session, following References [5, 29, 57], both user's short- and long-term interests in the current session are taken into consideration for generating the final preference. Since it has been proved that the last interacted item in the ongoing session can represent user's instant interest [5, 29, 57], we adopt the representation of the last item, i.e., $\mathbf{z}_n$, as user's short-term interest:

$$\mathbf{s}_r = \mathbf{z}_n, \tag{16}$$

where $\mathbf{s}_r \in \mathbb{R}^d$ indicates user's recent interest.

Besides, the attention mechanism is applied to selectively combine the item representations in the ongoing session as user's long-term interest as follows:

$$\begin{aligned} \mathbf{s}_l &= \sum_{i=1}^{n} \beta_i \mathbf{z}_i, \\ \beta_i &= \text{Softmax}(\epsilon_i), \\ \epsilon_i &= \mathbf{W}_4^\top \sigma(\mathbf{W}_5 \mathbf{z}_i + \mathbf{W}_6 \mathbf{s}_r + \mathbf{b}), \end{aligned} \tag{17}$$

where $\mathbf{s}_l \in \mathbb{R}^d$ denotes user's long-term interest, $\mathbf{W}_4 \in \mathbb{R}^d$, $\mathbf{W}_5, \mathbf{W}_6 \in \mathbb{R}^{d \times d}$ are trainable parameters, and $\mathbf{b} \in \mathbb{R}^d$ is the bias.

After that, we combine the short- and long-term interests as the final hybrid user preference $\mathbf{s}_h \in \mathbb{R}^d$ by a concatenation:

$$\mathbf{s}_h = \mathbf{W}_7[\mathbf{s}_l; \mathbf{s}_r], \tag{18}$$

where $[\cdot]$ denotes the concatenation operation and $\mathbf{W}_7 \in \mathbb{R}^{d \times 2d}$ is the trainable parameters.

4.3 Prediction and Model Optimization

4.3.1 Max Cross-entropy. After generating user's preference, we produce the prediction scores by multiplying the representations of user preference and the candidate items as follows:

$$\tilde{\mathbf{y}}_i = \mathbf{s}_h^\top \mathbf{v}_i \quad \forall i = 1, 2, \ldots, |V|. \tag{19}$$

After obtaining the prediction scores, the state-of-the-art methods typically follow the training mechanism that first normalizes the scores through the softmax layer and then inputs them into the cross-entropy loss for optimization [23, 29, 55, 57]. However, such a training mechanism involves adopting all items (excluding the target item) as the negative samples will make the model over-fitting, as stated in Section 1. Thus, we propose a Max Cross-Entropy (MCE) loss for optimization to prevent the over-fitting problem. Specifically, after obtaining the prediction scores at each iteration, we select the items from the set excluding the target item, i.e., $V \backslash v^+$, with the highest scores at a ratio of γ as the negative samples V^-:

$$V^- = \text{MaxSampling}(V \backslash v^+, \gamma, \tilde{\mathbf{y}}), \tag{20}$$

where $\tilde{\mathbf{y}}$ is the prediction scores on all candidate items, "\" indicates the set subtraction operation, and v^+ is the target item.

Then, we combine the sampled items V^- and the target item v^+ as $\tilde{V}$, and normalize the scores of items in $\tilde{V}$ using the softmax layer as follows:

$$\hat{\mathbf{y}}_i = \text{Softmax}(\tilde{\mathbf{y}}_i), \tag{21}$$

where $\tilde{\mathbf{y}}_i$ is the prediction score of item $v_i \in \tilde{V}$.

After normalization, the optimization loss is defined as the cross-entropy of the ground-truth and the prediction scores $\hat{\mathbf{y}}$:

$$L = -\sum_{i=1}^{|\tilde{V}|} \mathbf{y}_i \log(\hat{\mathbf{y}}_i) = -\log(\hat{\mathbf{y}}_{v^+}), \tag{22}$$

where $\mathbf{y}_i \in \mathbf{y}$ reflects the one-hot encoding vector of the ground truth in $\tilde{V}$, and $\hat{\mathbf{y}}_{v^+}$ is the normalized score of the target item. In addition, the **Back-Propagation Through Time (BPTT)** algorithm [39] is adopted to train GCARM.

We show the training process of our proposed GCARM in Algorithm 3. Specifically, given the training set, we first establish the global connections from all sessions in line 1; then, for each session, we learn the representations of the contained items in line 5. After that, we generate user's hybrid preference from line 6 to 8, including modeling the recent and long-term interests in lines 6 and 7, respectively, which are concatenated as the final preference in line 8. Then, we make predictions in line 9 and conduct the max sampling to get the negative samples for comparison in line 10. Next, we normalize the scores of the target and negative items in line 11 and adopt the cross-entropy as the loss function in line 12. Finally, we utilize the back propagation to optimize the parameters in our proposal in each minibatch in line 15.

ALGORITHM 3: Training Process of GCARM

Input: U: all training sessions;
Epochs: the number of training interactions;
Batches: the number of minibatches in each epoch;
Output: $\mathbf{V}$: embeddings of all items in V;
ψ^{IL}, ψ^{IG}, ψ^{CL}, ψ^{CG}: trainable parameters in the attentive information propagation;
$\mathbf{W}_{\tilde{c}}, \mathbf{W}_{\hat{c}}$: the trainable parameters in the neighbor- and item-level co-attentions;
$\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$: trainable parameters in the item representation combination;
$\mathbf{W}_4, \mathbf{W}_5, \mathbf{W}_6, \mathbf{W}_7$: trainable parameters in the user preference generation;

$G = \{V, E\} \leftarrow$ Establish(U);
for epoch in range(Epochs) **do**
  for i in range(Batches) **do**
    for session S in U **do**
      $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\} \leftarrow$ use Equations (4)–(15) with S and G as inputs;
      $\mathbf{s}_r \leftarrow$ use Equation (16) with $\mathbf{Z}$ as inputs;
      $\mathbf{s}_l \leftarrow$ use Equation (17) with $\mathbf{Z}$ and $\mathbf{s}_r$ as inputs;
      $\mathbf{s}_h \leftarrow$ use Equation (18) with $\mathbf{s}_r$ and $\mathbf{s}_l$ as inputs;
      $\tilde{\mathbf{y}} \leftarrow$ use Equation (19) with $\mathbf{s}_h$ and $\mathbf{V}$ as inputs;
      $V^- \leftarrow$ use Equation (20) with $V \backslash v^+$ and $\tilde{\mathbf{y}}$ as inputs;
      $\hat{\mathbf{y}} \leftarrow$ use Equation (21) with $\tilde{\mathbf{y}}$, V^- and v^+ as inputs;
      $L \leftarrow$ use Equation (22) with $\mathbf{y}$ and $\hat{\mathbf{y}}$ as inputs;
    end for
  end for
  use back propagation to optimize the parameters;
end for
return The trainable parameters of GCARM and item embeddings.

4.3.2 Theoretical Analysis. In this section, we analyze the Max Cross-Entropy (MCE) loss to illustrate why it can be utilized for preventing the over-fitting problem and meanwhile achieving an effective optimization.

- First, we explain why adopting the sampled items in the item set (excluding the target item) for comparison can avoid over-fitting. As only positive interactions in SBR are observed [17, 57], we cannot certainly guarantee whether the user likes the other items or not. In the existing works, which adopt the cross entropy with softmax [5, 57] for optimization, the items with no interactions are typically all regarded as the negative samples for comparison in each iteration, which is unreasonable. Picking up the item with a relatively low prediction score as a negative sample will completely make it impossible to be recommended to the user in the validation and test sets, i.e., the model will be over-fitting on the training set.
- Next, we theoretically analyze why MCE can effectively optimize the model. Specifically, the normalization of the target item score through the softmax layer in Equation (21) can be formulated as:

$$\hat{\mathbf{y}}_{v^+} = \frac{exp(\tilde{\mathbf{y}}_{v^+})}{exp(\tilde{\mathbf{y}}_{v^+}) + \sum_{v_i \in V^-} exp(\tilde{\mathbf{y}}_i)}. \tag{23}$$

Then, combining Equation (23) with Equation (22), we can calculate the gradient of the loss L w.r.t. the target score as:

$$\frac{\partial L}{\partial \tilde{\mathbf{y}}_{v^+}} = \frac{exp(\tilde{\mathbf{y}}_{v^+})}{exp(\tilde{\mathbf{y}}_{v^+}) + \sum_{v_i \in V^-} exp(\tilde{\mathbf{y}}_i)} - 1. \tag{24}$$

Clearly, the gradient is a negative value, indicating that when the loss is optimized to decrease, the score of the target item will be increased. Moreover, the smaller the gradient is, the faster the target score is increased. From Equation (24), we can observe that increasing the value of $\sum_{v_i \in V^-} exp(\tilde{\mathbf{y}}_i)$ can decrease the gradient of the target score, thus the prediction score on the target item can be increased quickly. Thus, to prevent over-fitting and to provide enough gradient for increasing the target item score, we conduct the max sampling on the prediction scores to obtain the negative sample, as illustrated in Equation (20).

It is worth noting that adopting relatively more items with the largest scores may result in the over-fitting problem, and adopting relatively less items cannot provide enough gradient for increasing the target score. Hence, we refer to a parameter γ in Equation (20) to achieve a good performance. We will detail the impact of γ on the model performance in Section 6.4.

5 EXPERIMENTS

5.1 Research Questions

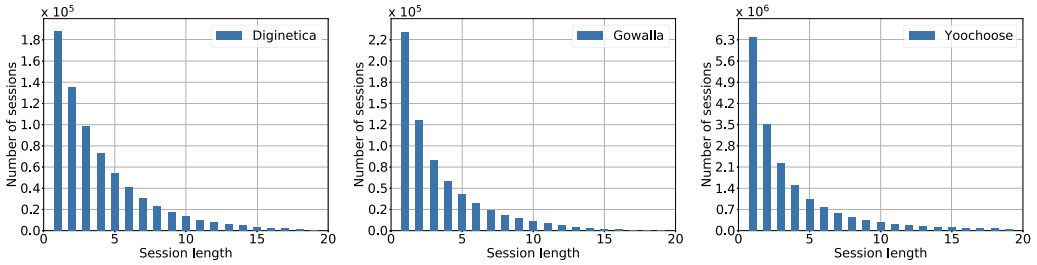
We examine the performance of our proposed GCARM by addressing the following seven research questions:

(RQ1) Can the proposed GCARM model achieve better performance than the state-of-the-art baselines for the session-based recommendation task?
(RQ2) How is the contribution of each component in GCAT to the performance of GCARM?
(RQ3) Can the Max Cross-Entropy (MCE) loss alleviate the over-fitting problem, and what is the impact of the parameter γ on the model performance?
(RQ4) How does GCARM perform with different number of global neighbors incorporated in the global graph?
(RQ5) How does GCARM perform on sessions with different lengths?
(RQ6) How is the performance of GCARM on a large-scale dataset?
(RQ7) How is the time and space complexity of GCARM compared with the competitive baselines?

5.2 Datasets and Evaluation Metrics

We adopt three benchmark datasets for evaluation, i.e., Diginetica,[2] Gowalla,[3] and Yoochoose.[4] Diginetica is an e-commerce dataset released by CIKM Cup 2016. Following References [5, 23, 57], we adopt the transaction data. Gowalla is a check-in dataset collected from February 2009 to October 2010. We keep the top 3,000 most popular locations for experiments [5, 13]. Yoochoose comes from the RecSys Challenge 2015, containing the click-streams of users. Items of each user in Diginetica as well as Yoochoose and the check-in records of each user in 24 hours in Gowalla are regarded as a session [5, 57]. Moreover, in all three datasets, items appearing less than five times are filtered, and sessions of length 1 are removed [5, 57]. Sessions in the last week in Diginetica and the last 20% of the sessions in Gowalla as well as Yoochoose are separated as the test set, respectively, and the remaining parts are utilized for training. In addition, we use the last 20% part separated from the training set as the validation set for tuning the hyper parameters

[2] http://cikm2016.cs.iupui.edu/cikm-cup.
[3] https://snap.stanford.edu/data/loc-gowalla.html.
[4] http://2015.recsyschallenge.com/challenge.html.



(a) Diginetica. **(b) Gowalla.** **(c) Yoochoose.**

Fig. 4. Distribution of sessions with various lengths in the training set of three datasets.

Table 2. Dataset Statistics

Statistics	Diginetica	Gowalla	Yoochoose
# clicks	981,620	1,122,788	28,812,618
# training sessions	716,835	675,561	18,235,054
# test sessions	60,194	155,332	3,044,742
# items	42,596	29,510	35,044
Average session length	4.80	3.85	3.82

of GCARM. The data statistics after processing are shown in Table 2. Moreover, we plot the distribution of sessions of different lengths (excluding the target item) in all three datasets in Figure 4, where we can find that most sessions are associated with less than five interactions, especially in Gowalla and Yoochoose. We choose Diginetica and Gowalla to evaluate the general performance of GCARM and the baselines for answering RQ1 to RQ5, and then adopt Yoochoose to examine the generalizability of GCARM on a large-scale dataset.

Following previous works [5, 57], Recall@K and MRR@K are adopted for evaluating the recommendation performance.

5.3 Model Summary

To examine the effectiveness of GCARM, we compare our proposal with the following competitive methods:

- **Item-KNN** [10] recommends similar candidates to items in the ongoing session by relying on a cosine similarity;
- **FPMC** [38] is a hybrid method where the Markov Chains are adopted for capturing the sequential signals; here, the user representation is omitted considering its unavailability in SBR.
- **NextItNet** [66] is a CNN-based method that increases the receptive fields using the dilated convolution rather than the lossy pooling operations;
- **NARM** [23] is an RNN-based approach that utilizes the GRUs to capture the sequential signals and adopts the attention mechanism to emphasize user's main intent;
- **FGNN** [33] designs a weighted graph attention network for computing the information flow in the session graph and generates the user preference by a Readout function;
- **SR-GNN** [57] proposes to transform each session into a graph and adopts the GGNNs to capture the pair-wise transition relations between items;
- **GC-SAN** [59] improves SR-GNN by adopting the self-attention mechanism for capturing the long-range dependencies between items;

- **SGNN-HN** [29] extends SR-GNN by introducing the star node for exploring the long-distance information in GNNs and alleviates the over-fitting problem using the highway networks [43];
- **LESSR** [5] proposes to handle the information loss problem in the GNN-based methods for SBR using the lossless edge-order preserving aggregation and the shortcut graph attention;
- **GCE-GNN** [55] designs a unified model that exploits the global- and session-level transition patterns between items in the global and session graphs, respectively.

5.4 Experimental Setup

Following References [23, 29, 57], we adopt a sequence splitting method to augment the training and test samples, i.e., generating sequences $([v_1], v_2), ([v_1, v_2], v_3), \ldots, ([v_1, \ldots, v_{n-1}], v_n)$ for session $S = \{v_1, v_2, \ldots, v_n\}$. Then, we model each augmented sequence and make recommendations individually. The hyper parameters of GCARM as well as the baselines are tuned on the validation set. Specifically, we apply a grid search to find the optimal hyper parameters of each model, where the embedding dimension is searched in $\{16, 32, 64, 128, 256\}$, the learning rate is tuned in $\{1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}\}$, and the L2 regularization is ranged in $\{0, 1e^{-6}, 1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}\}$. Moreover, for the GNN-based methods, we search the layer number of GNNs in $\{1, 2, 3, 4, 5\}$. In addition, the Adam optimizer is adopted for training the models and the batch size is set to 512. For our proposal GCARM, the global neighbor number N and the hop number H are both searched in $\{1, 2, 3\}$, and the ratio γ of negative samples in the MCE loss is tuned in $\{1, 1/2, 1/4, \ldots, 1/256\}$. In addition, the learning rate is initially set to $1e^{-3}$ and decays by 0.1 for every 3 epochs following References [29, 57], and the dimension of the item embedding is finally set to 128. All parameters are initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1.

To facilitate the reproducibility of our results in this article, we share the code and data used to run the experiments at https://bitbucket.org/nudtpanzq/gcarm/src/master/.

6 RESULTS AND DISCUSSION

6.1 Overall Performance

6.1.1 General Comparison. The results of our proposed GCARM and the baselines in terms of Recall@20 and MRR@20 are reported in Table 3. First, as shown in Table 3, the neural models, e.g., NARM and NextItNet, generally achieve a better performance than the traditional methods, i.e., Item-KNN and FPMC. In detail, Item-KNN outperforms FPMC for all cases on both datasets, which can be due to the fact that the Markov Chains in FPMC only capture the sequential dependencies between adjacent items, ignoring the relations among the whole item sequence.

In the neural models, the RNN-based method, i.e., NARM, shows better performance than the CNN-based model, i.e., NextItNet, which indicates that RNN outperforms CNN in modeling sequential signals in SBR. Moreover, GNN-based approaches achieve the state-of-the-art performance, indicating that modeling the pair-wise transitions between items in a sequence can help to improve the performance of SBR. In addition, we can find that, upon SR-GNN, capturing the long-term dependencies between items in a session can effectively improve the recommendation accuracy, e.g., GC-SAN and SGNN-HN. In particular, SGNN-HN performs best among the baselines in terms of Recall@20 on both datasets. Moreover, by solving the lossy session encoding problem, LESSR achieves the best performance in terms of MRR@20 on both Diginetica and Gowalla. As for the baseline GCE-GNN that also considers the global- as well as local-level item transitions, the performance is not satisfactory, especially in terms of MRR@20. This may be because that GCE-GNN cannot distinguish some unrelated items in both local and global graphs, which may introduce the

Table 3. Model Performance

Method	Diginetica		Gowalla	
	Recall@20	MRR@20	Recall@20	MRR@20
Item-KNN	39.51	11.22	38.60	16.66
FPMC	28.50	7.67	29.91	11.45
NextItNet	45.41	15.19	45.15	21.26
NARM	49.80	16.57	50.07	23.92
FGNN	50.03	17.01	50.06	24.12
SR-GNN	50.81	17.31	50.32	24.25
GC-SAN	50.90	17.63	50.68	24.67
SGNN-HN	<u>52.79</u>	17.36	<u>52.82</u>	24.65
LESSR	51.71	<u>18.15</u>	51.34	<u>25.49</u>
GCE-GNN	51.66	17.53	51.53	23.52
GCARM	**53.80**▲	**18.57**▲	**53.75**▲	**26.45**▲

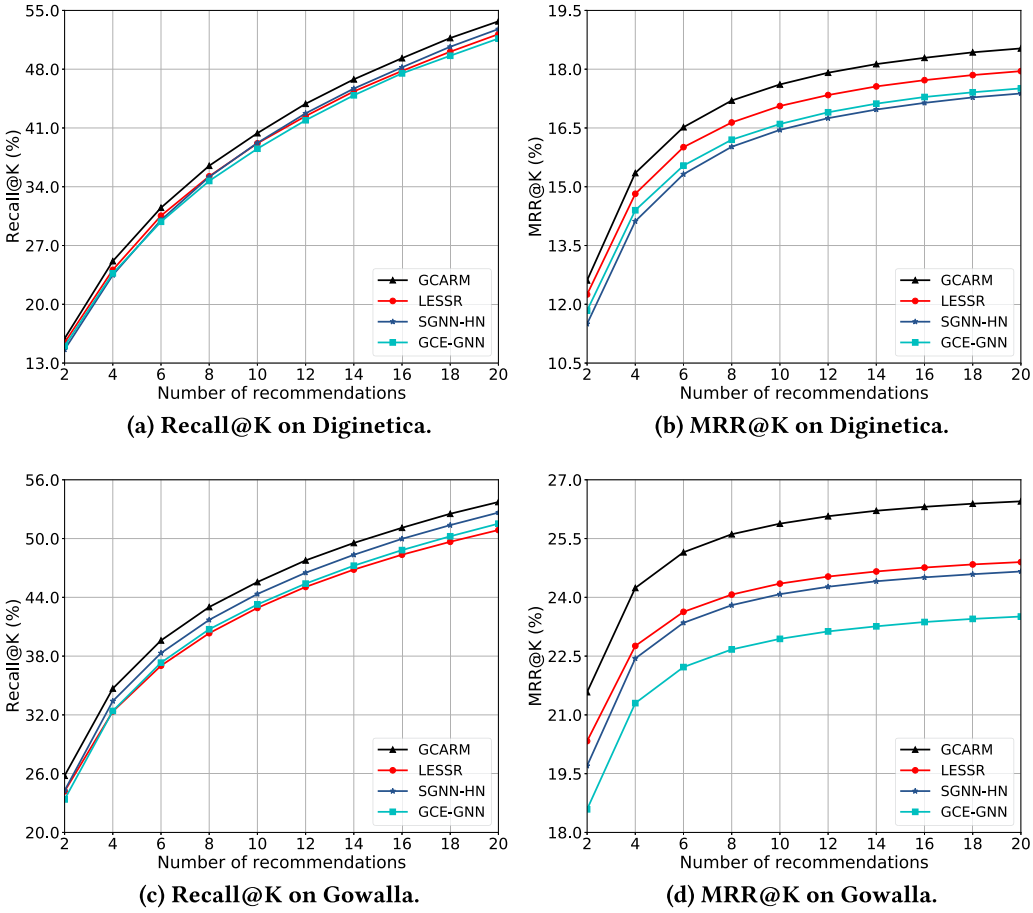
The results of the best-performing baseline and the best performer in each column are underlined and boldfaced, respectively. ▲ denotes a significant improvement of GCARM over the best baseline using a paired t-test ($p < 0.01$).

bias in user intent detection. Thus, in following experiments, we take the leading baselines, i.e., SGNN-HN, LESSR, and GCE-GNN, for further comparisons.

For our proposed GCARM, on both datasets, we can find that GCARM outperforms the baselines in terms of both metrics significantly, which are due to the following facts: First, we consider the dynamic connections between local and global graphs simultaneously in both the information propagation and the item representation processes, which can thus avoid the bias introduced by accidental clicks on items in session and the unrelated global neighbors. Second, our designed Max Cross-Entropy (MCE) loss can solve the over-fitting problem to some extent, which improves the generalization ability of our proposal, and thus increases the recommendation accuracy. Moreover, the improvements of GCARM over the best corresponding baselines in terms of Recall@20 and MRR@20 are 1.91% and 2.31% on the Diginetica dataset, and 1.76% and 3.77% on Gowalla, respectively. It is obvious that the improvements in terms of MRR@20 is higher than that of Recall@20 on both datasets, indicating that our proposal contributes more to ranking the target item at a high position than hitting them in the recommendation list.

6.1.2 Model Performance at Various Recommendations. To validate the effectiveness of GCARM at different recommendations, we compare the performance of GCARM with the baselines LESSR, SGNN-HN, and GCE-GNN when the recommendation number K changes from 2 to 20. The results are presented in Figure 5. As shown in Figure 5, we can observe that GCARM can always achieve the best performance for cases at different recommendations. Moreover, when the number of recommendations changes from 2 to 20, the performance of all models in terms of Recall@K and MRR@K on both datasets generally increases, since a recommendation list with a larger size is more likely to include the target item.

For Recall@K, from Figures 5(a) and 5(c), we can see that on both datasets, the performance of the baselines are similar when the recommendation number is small, and the gap between their performance becomes larger when recommendation number is increased. This is especially obvious on Diginetica. However, the performance improvements of GCARM over the baselines on both Diginetica and Gowalla are obvious with various recommendation numbers, especially for short recommendation lists. For instance, the improvements of GCARM against the best baseline



(a) Recall@K on Diginetica. **(b) MRR@K on Diginetica.**

(c) Recall@K on Gowalla. **(d) MRR@K on Gowalla.**

Fig. 5. Model performance at various recommendations on Diginetica and Gowalla.

SGNN-HN in terms of Recall@2 are 9.04% and 6.26% on Diginetica and Gowalla, respectively. For Recall@20, the corresponding improvements are 1.74% and 2.05%. This indicates that our proposed GCARM is more applicable to the scenarios where the number of recommendations is small due to a limited display interface [70], e.g., the recommendation on mobile phones. As for MRR@K, we can observe that the results are similar to that in terms of Recall@K on both datasets. The main difference is that the improvements in terms of MRR@K are more obvious than Recall@K at various number of recommendations, which is consistent with the findings in Section 6.1.1.

6.2 Ablation Study in GCAT

For RQ2, we perform an ablation study by comparing GCARM with its variants to analyze the effectiveness of each component in the proposed Graph Co-Attention Network (GCAT). Specifically, we produce six variants for comparison, including: (1) w/o GNN_{local}, which removes the whole local graph; (2) w/o GNN_{global}, which removes the whole global graph; (3) w/o GNN_{catt}, which removes the *co-attentive information propagation* in GCAT; (4) w/o GNN_{self}, which removes the *individual information propagation* in GCAT; (5) w/o IRC_{catt}, which removes the *co-attentive item*

Table 4. Ablation Study in GCAT

Method	Diginetica		Gowalla	
	Recall@20	MRR@20	Recall@20	MRR@20
w/o GNN_{local}	50.47	16.43	51.65	25.25
w/o GNN_{global}	53.16	18.16	53.04	25.08
w/o GNN_{catt}	53.29	18.01	53.28	25.96
w/o GNN_{self}	53.71	18.47	53.62	26.06
w/o IRC_{catt}	53.44	18.34	53.63	26.34
w/o IRC_{self}	53.24	18.42	53.34	25.40
GCARM	**53.80**	**18.57**	**53.75**	**26.45**

vectors in the item representation combination; and (6) w/o IRC_{self}, which removes the *individual item vectors* in the item representation combination. The results are shown in Table 4.

From Table 4, we can observe that removing each component of GCAT leads to a performance decrease, indicating that each component in GCAT contributes to the user preference modeling. Moreover, comparing w/o GNN_{catt} to w/o GNN_{local} and w/o GNN_{global}, we can observe that w/o GNN_{catt} generally outperforms w/o GNN_{local} and w/o GNN_{global}, indicating that combining the individual representations of items in the local and global graphs can improve the performance. However, w/o GNN_{catt} loses the competition to w/o GNN_{global} in terms of MRR@20 on Diginetica. We attribute the reason that simply concatenating the local and global item representations in the variant w/o GNN_{catt} may introduce unrelated information from the neighbors in the global graph. This is consistent with the finding that w/o GNN_{self} performs better than w/o GNN_{catt} for all cases on two datasets. By considering the dynamic connections between the local and global neighbors of each item in session, w/o GNN_{self} can effectively filter out the unrelated information in both local and global graphs and achieves a better performance than w/o GNN_{catt}.

Moreover, by comparing GCARM to w/o IRC_{catt} and w/o IRC_{self}, we can observe that both the individual and co-attentive item representations can contribute to generating an accurate item representation, which helps to improve the recommendation accuracy. On the one hand, the introduced co-attentive item representations can accurately distinguish the items with different importances in the local and global graphs, which helps to capture the user's intent in the current session. On the other hand, the individual item representations can preserve the personalized and generalized user behavior patterns, which can avoid the over-smoothing problem [60] in GNNs.

6.3 Impact of Global Neighbors

For RQ3, to investigate the impact of the number of global neighbors on the performance of GCARM, we apply a grid setting by ranging both the global neighbor number N and the hop number H in $\{1, 2, 3\}$, respectively, and we show the results in Table 5.

First, on the Diginetica dataset, we can find that the best performance of GCARM in terms of Recall@20 and MRR@20 are achieved with different parameters, i.e., the best Recall@20 is obtained by setting the neighbor number to 2 as well as the hop number to 1, and the best MRR@20 is returned by setting the neighbor number to 1 as well as the hop number to 3, respectively. In terms of Recall@20, we can see that increasing either the neighbor number or the hop number generally decreases the recommendation accuracy, which can be due to the noisy information introduced from the global neighbors. However, in terms of MRR@20, increasing the hop number consistently improves the performance when the neighbor number is fixed, while tuning the number of neighbors can not guarantee a good performance, which indicates that it is more reasonable

Table 5. Impact of Number of Global Neighbors

		Recall@20			MRR@20		
		H = 1	H = 2	H = 3	H = 1	H = 2	H = 3
Diginetica	N = 1	53.76	53.75	53.62	18.45	18.59	**18.67**
	N = 2	**53.80**	53.49	53.33	18.57	18.61	18.62
	N = 3	53.65	53.46	53.38	18.53	18.53	18.58
Gowalla	N = 1	53.71	53.56	53.59	26.39	26.39	26.38
	N = 2	53.75	53.53	53.39	26.46	26.45	26.44
	N = 3	**53.77**	53.47	53.33	**26.51**	26.44	26.33

to incorporate neighbors of multi hops than to increase the global neighbors of each item if we aim to rank the target item at a high position in a recommendation list.

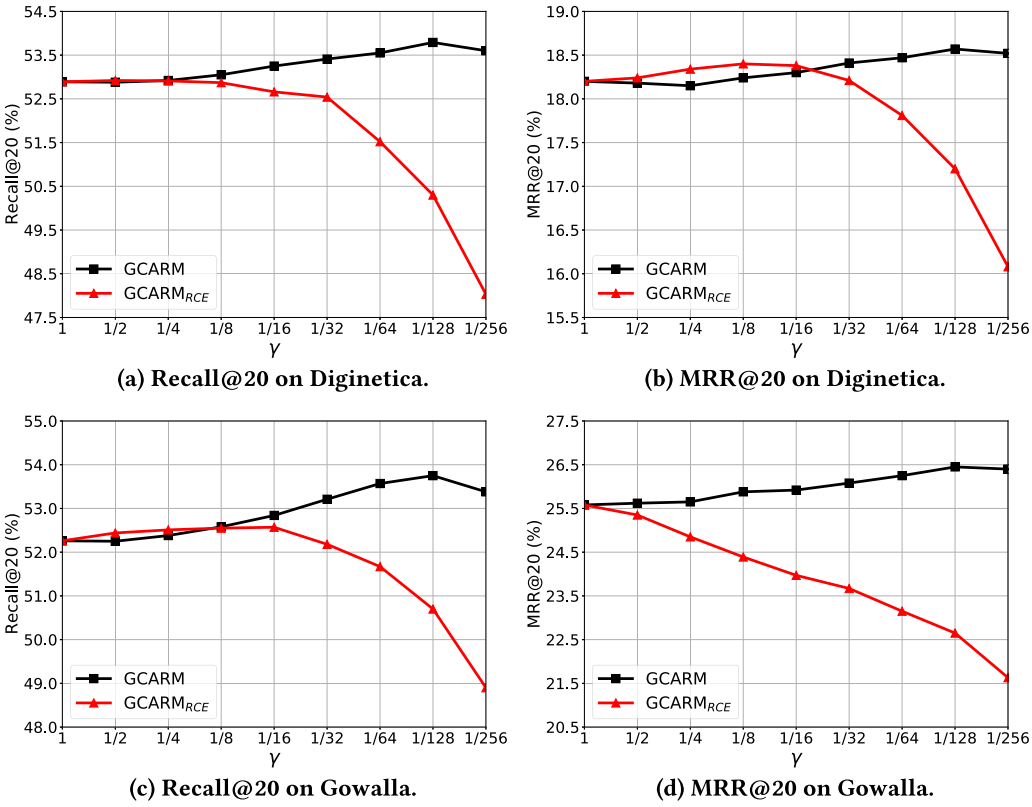
As for the Gowalla dataset, we can observe that the best performance in terms of both metrics are achieved with the neighbor number 3 and the hop number 1. Moreover, for the cases with a fixed number of neighbors, when the hop number is increased, the recommendation performance in terms of both Recall@20 and MRR@20 consistently goes down. It could be explained by the fact that the bias is easily introduced by increasing the hop number of incorporating the global neighbors in the check-in scenario.

6.4 Analysis on Max Cross-entropy

For RQ4, to examine the effectiveness of the designed Max Cross-Entropy (MCE) loss, we compare MCE with its variant Random Cross-Entropy (RCE), which replaces the max sampling strategy with the random sampling strategy in MCE. We denote the GCARM model that adopts the RCE loss as GCARM$_{RCE}$. We tune the parameter γ, which controls the ratio of negative samples in $\{1, 1/2, 1/4, \ldots, 1/256\}$ in both GCARM and GCARM$_{RCE}$ to see the impact of γ on the model performance. The results are presented in Figure 6.

On the Diginetica dataset, from Figures 6(a) and 6(b), we can see that when γ decreases, the performance of GCARM in terms of Recall@20 and MRR@20 is improved at first and achieves the peak value at $\gamma = 1/128$. After that, it begins to decrease. This means that adopting all items (excluding the target item) as the negative samples during training will hurt the recommendation accuracy, which is due to the over-fitting problem we describe in Sections 1 and 4.3.2. Moreover, we can observe that GCARM$_{RCE}$ performs stably in terms of Recall@20 with a large γ and then the performance consistently drops when γ decreases. For MRR@20, the performance first increases slightly from $\gamma = 1$ to 1/8 and then consistently decreases. This indicates that adopting the random sample strategy in GCARM$_{RCE}$ can also avoid the over-fitting problem to some extent and thus improves the recommendation performance. However, the random sampled items can not provide enough gradient for model optimization as analyzed in Section 4.3.2. Instead, in GCARM, we utilize the items with the highest prediction scores as the negative samples, which can provide enough gradient for training and avoid over-fitting at the same time. The results on Gowalla shown in Figures 6(c) and 6(d) are similar to those on Diginetica except that with γ decreasing, the performance of GCARM$_{RCE}$ in terms of MRR@20 consistently decreases. This indicates that the gradient shortage problem in the RCE loss is more serious on the Gowalla dataset than that on Diginetica. However, GCARM can still provide enough gradient by taking items with the largest prediction scores for comparison and shows large improvements over GCARM$_{RCE}$.

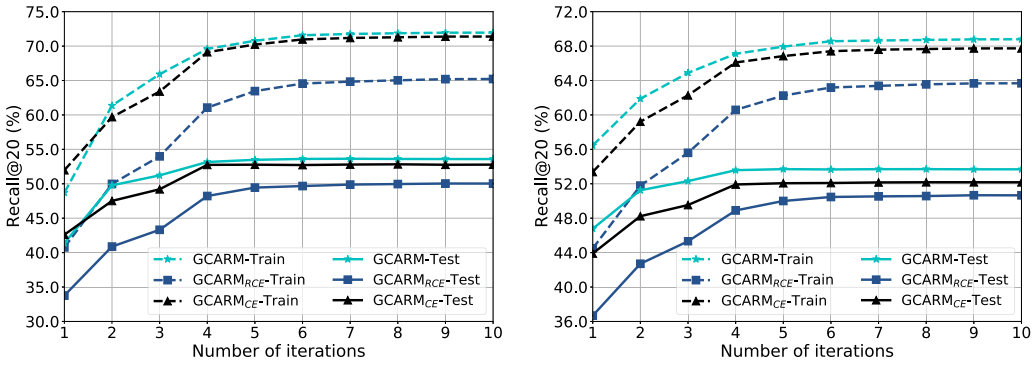
In addition, we also plot the performance curves on both the training and test sets of GCARM and GCARM$_{RCE}$ as well as the variant GCARM$_{CE}$, which adopts the original cross-entropy regarding all items except the target item as the negative items. We set the parameter γ in both GCARM



(a) Recall@20 on Diginetica. **(b) MRR@20 on Diginetica.**

(c) Recall@20 on Gowalla. **(d) MRR@20 on Gowalla.**

Fig. 6. Model performance of GCARM and GCARM$_{RCE}$ with different γ.



(a) Recall@20 on Diginetica. **(b) Recall@20 on Gowalla.**

Fig. 7. Recommendation performance with different numbers of iterations.

and GCARM$_{RCE}$ to 1/128 as an example and the results in terms of Recall@20 are plotted in Figure 7. Similar phenomena can be observed in terms of MRR@20. From Figure 7, we can see that both the training and test performance of involved methods on two datasets first increases and then keeps stable without an obvious decrease. This is due to us applying a learning rate decay

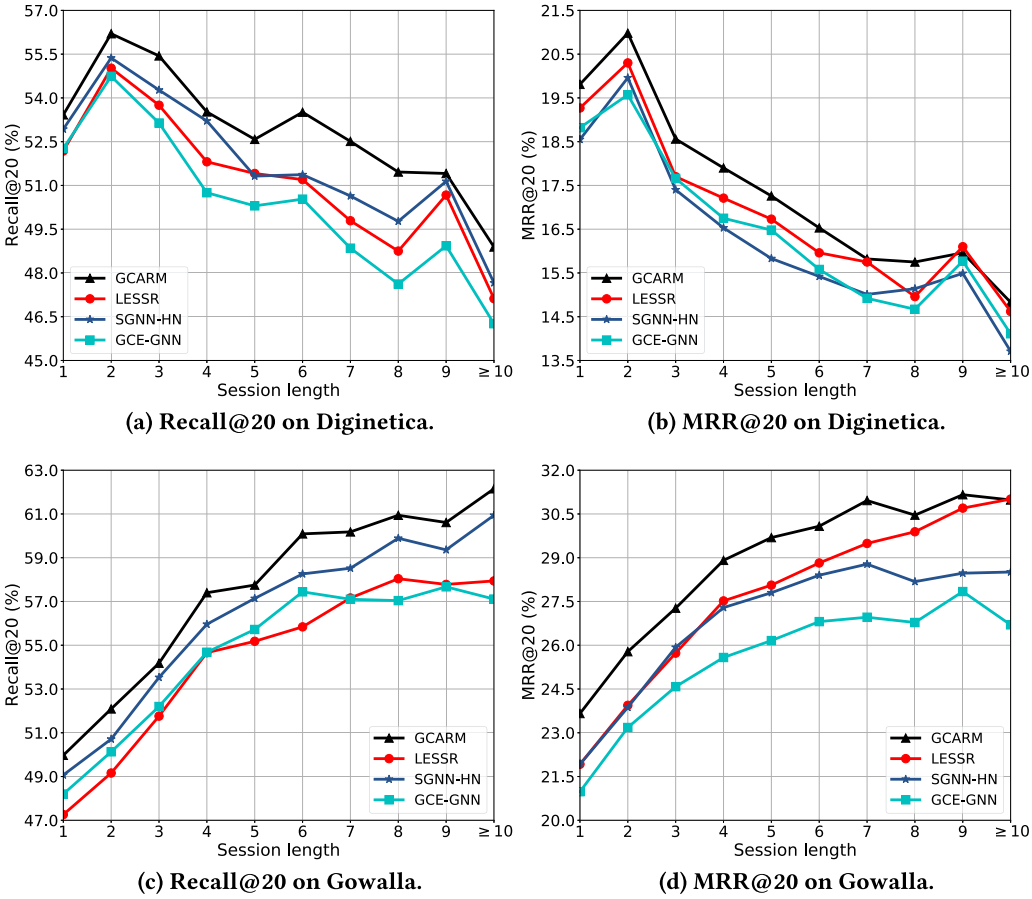
method in the Adam optimizer [29, 57], which is detailed in Section 5.4. We can investigate the extent of the over-fitting problem by comparing the achieved stable training and test performance. Specifically, a larger drop rate of the test performance against the training performance indicates a more serious over-fitting problem. Taking the results at the 10-th iteration on the Gowalla dataset as an example where the performance of all methods has achieved a stable status, the corresponding drop rates of GCARM and GCARM$_{RCE}$ are 21.99% and 20.46%, respectively, which are lower than that of GCARM$_{CE}$, i.e., 23.01%. This indicates that adopting the sampled items without interactions as the negative samples in GCARM and GCARM$_{RCE}$ can both alleviate the over-fitting problem, which is consistent with the theoretical analysis in Section 4.3.2. Moreover, we can see that the performance of GCARM$_{RCE}$ is obviously worse than GCARM and GCARM$_{CE}$, which is due to the fact that the randomly selected items cannot provide enough gradient for increasing the target item score, which is analyzed in Section 4.3.2. However, by using the max sampling to select the items with the highest prediction scores as the negative items, GCARM can solve the gradient shortage problem and achieve an effective model optimization.

6.5 Impact of the Session Length

As the majority of sessions are short (see Figure 4), it is meaningful to investigate the model performance on sessions with various number of interactions. We divide the sessions into groups according to their corresponding lengths, i.e., the number of interactions. Then, we evaluate the performance of GCARM and the competitive baselines, i.e., LESSR, SGNN-HN and GCE-GNN, on sessions with different lengths in both datasets. The results are shown in Figure 8. We can observe that GCARM can generally outperform the baselines in terms of both metrics with various lengths on both datasets, indicating that GCARM works well on detecting the user intent in sessions with different lengths.

In particular, on the Diginetica dataset, with the session length increasing, the performance of all models in terms of both metrics first increases from length 1 to 2, and then shows a continuous decreasing trend. This may be due to the fact that the increasing number of interacted items may bring the difficulty of user preference modeling, as the user intent may be diverse, especially in the e-commerce platform [8]. Moreover, we can see that the improvements of GCARM over the baselines in terms of both Recall@20 and MRR@20 are obvious when the session length is relatively short. For example, GCARM outperforms the best baseline SGNN-HN by 4.17% at length 6 in terms of Recall@20 and shows a 4.01% improvement above LESSR at length 4 in terms of MRR@20, while the corresponding improvements at length 10 are 2.58% and 1.44%, respectively. This indicates that comparing to the baselines, our proposed GCARM can effectively detect user's intent from limited interactions in an ongoing session on Diginetica.

On the Gowalla dataset, from Figures 8(c) and 8(d), we can find that the performance of all models in terms of both Recall@20 and MRR@20 shows an upward trend when the session length increases. This could be explained by the fact that the users are likely to focus on the similar places in the check-in scenario [58], thus a larger number of items contained in the session can provide relatively more information about the user's main purpose. In addition, comparing to the baseline GCE-GNN, which incorporates the global information, the improvements of GCARM over GCE-GNN in terms of both Recall@20 and MRR@20 keep increasing when the session length becomes large. This may be due to the fact that when the session contains relatively more items, the unrelated items in both local and global graphs in GCE-GNN may introduce more bias for user preference modeling. However, our proposed GCAT can solve the problem by the introduced neighbor-level and item-level co-attention mechanisms, thus can effectively detect user intent in the long sessions and make accurate recommendations.



(a) Recall@20 on Diginetica.

(b) MRR@20 on Diginetica.

(c) Recall@20 on Gowalla.

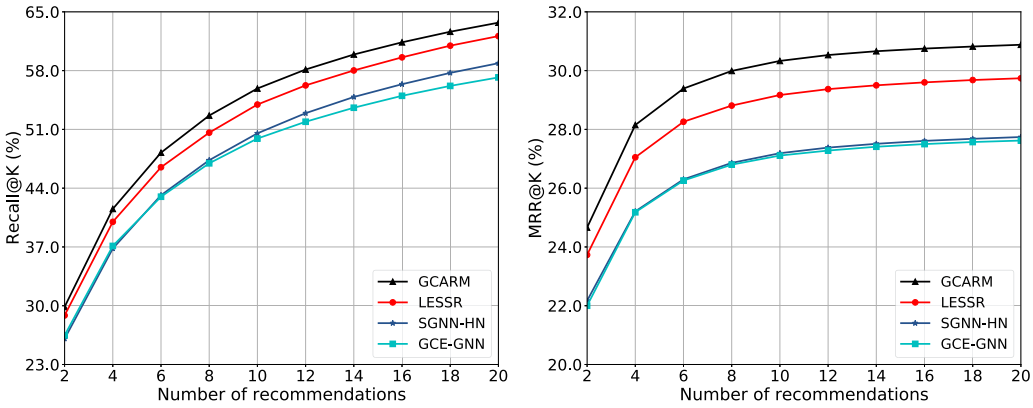
(d) MRR@20 on Gowalla.

Fig. 8. Model performance on sessions of different lengths.

6.6 Performance on a Large-scale Dataset

For RQ6, to validate the scalability of our proposal on large-scale datasets, we choose the Yoochoose dataset for experiments, which is much larger than Diginetica and Gowalla used in previous experiments, as shown in Table 2. Specifically, we compare the performance of GCARM and the competitive baselines in terms of Recall@K and MRR@K with K ranging from 2 to 20. The results are presented in Figure 9.

Clearly, from Figure 9, we can see that GCARM can outperform the baselines in all cases, indicating the effectiveness of our proposal on the large-scale datasets. Moreover, for the baselines, it is observed that LESSR shows a better performance than SGNN-HN and GCE-GNN, especially in terms of MRR@K. This may be due to the information loss problem being serious in the large-scale datasets. However, LESSR can effectively decrease the information loss and increase the recommendation accuracy. Moreover, the improvements of GCARM over the best baseline LESSR are consistently more obvious in terms of MRR@K than Recall@K. For instance, GCARM improves LESSR by 3.54% and 2.58% in terms of Recall@10 and Recall@20, respectively, while the respective improvements are 3.98% and 3.83% in terms of MRR@10 and MRR@20. This indicates that GCARM works especially well on ranking the target item at a right position, which is consistent with the findings obtained on smaller datasets (i.e., Diginetica and Gowalla) in Section 6.1.



(a) Recall@K on Diginetica. **(b) MRR@K on Diginetica.**

Fig. 9. Model performance at various recommendations on Yoochoose.

Table 6. Time and Space Complexity

Method	Time complexity	Space complexity
GCE-GNN	$O((\lvert E_l\rvert/2 + mM)d^2 + nd^2 + \lvert V\rvert d)$	$O(d + d^2 + Pd + \lvert V\rvert d)$
SGNN-HN	$O(Lm^2d^2 + nd^2 + \lvert V\rvert d)$	$O(d + d^2 + Pd + \lvert V\rvert d)$
LESSR	$O(L(\lvert E_l\rvert/2 + m^2)d^2 + nd^2 + \lvert V\rvert d)$	$O(Ld + Ld^2 + \lvert V\rvert d)$
GCARM	$O((\lvert E_l\rvert + mM)d^2 + nd^2 + \lvert V\rvert \log_2 \lvert V\rvert + \lvert V\rvert d)$	$O(d + d^2 + \lvert V\rvert d)$

6.7 Time and Space Complexity

To answer RQ7, we analyze the time complexity for modeling session $S = \{v_1, v_2, \ldots, v_n\}$ and the space complexity of our proposed GCARM as well as the state-of-the-art baselines, i.e., GCE-GNN, SGNN-HN, and LESSR. The results are provided in Table 6.

Here, in Table 6, m denotes the number of unique items in S, $|E_l|$ indicates the edge number in the local graph, M means the neighbor number of each item in the global graph, $|V|$ represents the number of all candidate items, L denotes the layer number, and P is the maximum session length in the dataset. Then the time complexity of GCARM is $O((|E_l| + mM)d^2 + nd^2 + |V| \log_2 |V| + |V|d)$, which mainly comes from the item representation learning module $O((|E_l| + mM)d^2)$, the user preference generation module $O(nd^2)$, and the prediction and optimization module $O(|V| \log_2 |V| + |V|d)$. For GCE-GNN, SGNN-HN, and LESSR, the time complexity is $O((|E_l|/2+mM)d^2+nd^2+|V|d)$, $O(Lm^2d^2+nd^2+|V|d)$, and $O(L(|E_l|/2+m^2)d^2+nd^2+|V|d)$, respectively. In real-world applications, $|V|$ is typically larger than other variables such as $|E_l|$, M, m, n, and d, and $d > \log_2 |V|$. Thus, the majority of the time complexity of four methods comes from the prediction module, i.e., $|V|d$. The others are relatively small and can be neglected. Thus, GCARM has a comparable time complexity to the baselines.

Moreover, as for the space complexity, from Table 6, we can observe that the space complexity of GCE-GNN, SGNN-HN, and LESSR is $O(d+d^2+Pd+|V|d)$, $O(d+d^2+Pd+|V|d)$ and $O(Ld+Ld^2+|V|d)$, respectively, which are larger than that of GCARM, i.e., $O(d + d^2 + |V|d)$. This is due to GCE-GNN and SGNN-HN introducing the trainable position embeddings with a space complexity of Pd to consider the sequential information in session. For LESSR, the parameters in multi-layer GNNs are not shared, introducing additional $(L - 1)(d + d^2)$ space complexity compared to GCARM. In general, GCARM has a slightly less space complexity than the competitive baselines.

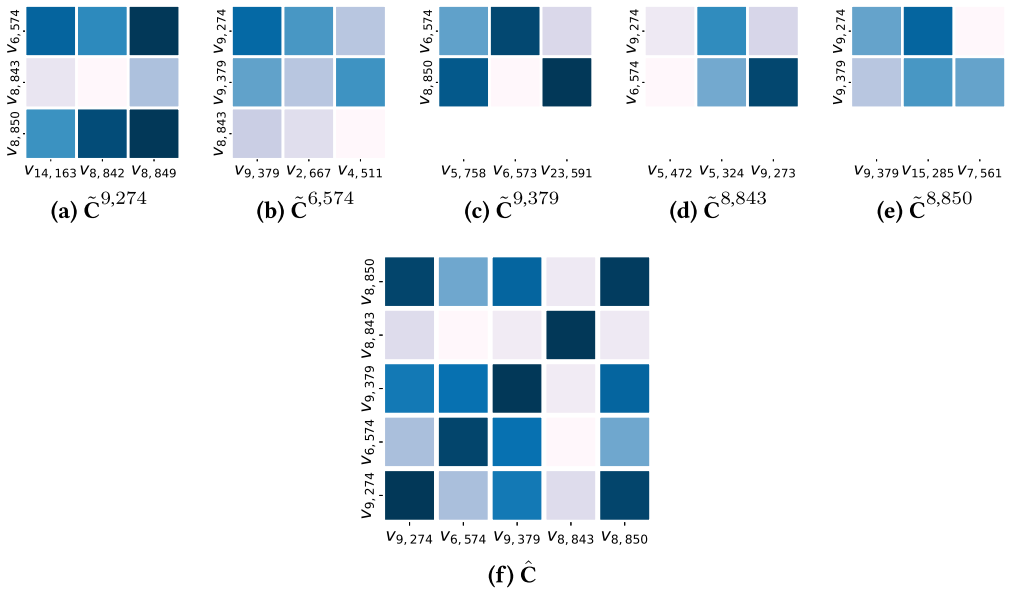


Fig. 10. Neighbor-level and item-level co-attention visualization in GCAT.

6.8 Case Study

To examine the effectiveness of GCAT in an intuitive way, we randomly sample a session from the Gowalla dataset and show the generated neighbor-level and item-level co-attention matrices in Figure 10. Specifically, for the sampled session $S = \{v_{9,274}, v_{6,574}, v_{9,379}, v_{6,574}, v_{8,843}, v_{9,274}, v_{8,850}, v_{9,379}\}$, we plot the neighbor-level co-attention scores obtained by Equation (7) in Figures 10(a)–10(e) corresponding to $\{v_{9,274}, v_{6,574}, v_{9,379}, v_{8,843}, v_{8,850}\}$ and the item-level co-attention matrix generated by Equation (12) in Figure 10(f), respectively. In Figures 10(a)–10(e), the y-axis items on the left of each co-attention matrix indicate the local neighbors and the x-axis items are the corresponding global neighbors. Similarly, the y-axis and x-axis items in Figure 10(f) denote the corresponding items in the local and global graphs, respectively. Moreover, the color depth indicates the similarity score of every two items. A darker color represents a higher similarity of two items.

Some interesting findings from Figure 10 can be observed.

- From Figures 10(a)–10(e), for most items in session, the local neighbors and global neighbors are different, which can respectively represent user's personalized and generalized behavior patterns;
- For the item $v_{8,843}$, the similarities of $v_{8,843}$ to other items in Figure 10(f) are small, indicating that $v_{8,843}$ may be an accidentally interacted item in S. Moreover, from Figures 10(a) and 10(b), item $v_{8,843}$ is also dissimilar to the incorporated global neighbors. Hence, we can filter out the unrelated item $v_{8,843}$ in current session by the neighbor-level and item-level co-attentions;
- From Figure 10(d), for the incorporated global neighbor $v_{5,472}$, the similarities of $v_{5,472}$ to the local items in $\tilde{\mathbf{C}}^{8,843}$ are generally small, indicating that the global graph may also contain unrelated information to the personalized interest. In GCAT, through the neighbor-level co-attention, we can filter out such unrelated neighbors.

In general, we can conclude that GCAT can simultaneously take the local and global item transitions into consideration, which enriches the information in current session and helps to generate

accurate item representations. Besides, the dynamic co-attention mechanisms in GCAT can help to eliminate the bias introduced by the unrelated items in both local and global graphs, which assists our model to accurately detect user's main intent in the ongoing session.

7 CONCLUSIONS AND FUTURE WORK

In this article, we propose a Graph Co-Attentive Recommendation Machine (GCARM) for session-based recommendation. GCARM applies a designed Graph Co-Attention Network (GCAT) to exploit the local and global pair-wise item transitions among items for user preference modeling in the ongoing session. Specifically, GCAT introduces a co-attention mechanism into the information propagation as well as the item representation combination processes to eliminate the noise brought by the unrelated items and thus generates accurate item representations. Moreover, to prevent the over-fitting problem in SBR, we propose a Max Cross-Entropy (MCE) loss, which takes the items with the largest prediction scores (excluding the target item) for comparison to prevent over-fitting. Extensive experiments conducted on three benchmark datasets demonstrate the effectiveness of GCARM in terms of Recall and MRR. Moreover, we find GCARM has a super ability to push the target item at an early position. As to future work, we plan to investigate the influence of user's multi-behaviors [20] for the session-based recommendation task. Moreover, we would like to design a strategy that can learn an adaptive γ to deal with different degrees of over-fitting problem in the training process. In addition, we are interested in exploring more sampling methods for selecting high-quality negative samples for model optimization, such as sampling the negative items with probability proportional to the prediction scores.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* 17, 6 (2005), 734–749.

[2] Veronika Bogina and Tsvi Kuflik. 2017. Incorporating dwell time in session-based recommendations with recurrent neural networks. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'17)*. ACM, 57–59.

[3] Yi Cao, Weifeng Zhang, Bo Song, and Congfu Xu. 2019. HiCAN: Hierarchical convolutional attention network for sequence modeling. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 1723–1732.

[4] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'17)*. ACM, 767–776.

[5] Tianwen Chen and Raymond Chi-Wing Wong. 2020. Handling information loss of graph neural networks for session-based recommendation. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'20)*. ACM, 1172–1180.

[6] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten de Rijke. 2019. A dynamic co-attention network for session-based recommendation. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 1461–1470.

[7] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten de Rijke. 2019. Joint neural collaborative filtering for recommender systems. *ACM Trans. Inf. Syst.* 37, 4 (2019), 39:1–39:30.

[8] Wanyu Chen, Pengjie Ren, Fei Cai, Fei Sun, and Maarten de Rijke. 2020. Improving end-to-end sequential recommendations with intent-aware diversification. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'20)*. ACM, 175–184.

[9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for YouTube recommendations. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'10)*. ACM, 191–198.

[10] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. 2010. The YouTube video recommendation system. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'10)*. ACM, 293–296.

[11] Diksha Garg, Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, and Gautam M. Shroff. 2019. Sequence and time aware neighborhood for session-based recommendations: STAN. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. ACM, 1069–1072.

[12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.

[13] Lei Guo, Hongzhi Yin, Qinyong Wang, Tong Chen, Alexander Zhou, and Nguyen Quoc Viet Hung. 2019. Streaming session-based recommendation. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'19)*. ACM, 1569–1577.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the International World Wide Web Conference (WWW'17)*. ACM, 173–182.

[15] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1 (2004), 5–53.

[16] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'18)*. ACM, 843–852.

[17] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR'16)*.

[18] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'17)*. ACM, 306–310.

[19] Sébastien Jean, KyungHyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'15)*. 1–10.

[20] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-behavior recommendation with graph convolutional networks. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20)*. ACM, 659–668.

[21] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'18)*. IEEE Computer Society, 197–206.

[22] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR'17)*.

[23] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'17)*. ACM, 1419–1428.

[24] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR'16)*.

[25] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput.* 7, 1 (2003), 76–80.

[26] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-term attention/memory priority model for session-based recommendation. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'18)*. ACM, 1831–1839.

[27] Anjing Luo, Pengpeng Zhao, Yanchi Liu, Fuzhen Zhuang, Deqing Wang, Jiajie Xu, Junhua Fang, and Victor S. Sheng. 2020. Collaborative self-attention network for session-based recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'20)*. 2591–2597.

[28] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'08)*. IEEE Computer Society, 502–511.

[29] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten de Rijke. 2020. Star graph neural networks for session-based recommendation. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'20)*. ACM, 1195–1204.

[30] Zhiqiang Pan, Fei Cai, Yanxiang Ling, and Maarten de Rijke. 2020. An intent-guided collaborative machine for session-based recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20)*. ACM, 1833–1836.

[31] Zhiqiang Pan, Fei Cai, Yanxiang Ling, and Maarten de Rijke. 2020. Rethinking item importance in session-based recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20)*. ACM, 1837–1840.

[32] Ruihong Qiu, Zi Huang, Jingjing Li, and Hongzhi Yin. 2020. Exploiting cross-session information for session-based recommendation with graph neural networks. *ACM Trans. Inf. Syst.* 38, 3 (2020), 22:1–22:23.

[33] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 579–588.

[34] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'17)*. ACM, 130–137.

[35] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. RepeatNet: A repeat aware neural recommendation machine for session-based recommendation. In *Proceedings of The AAAI Conference on Artificial Intelligence (AAAI'19)*. AAAI Press, 4806–4813.

[36] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM'14)*. ACM, 273–282.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI'09)*. 452–461.

[38] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the International World Wide Web Conference (WWW'10)*. ACM, 811–820.

[39] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 6088 (1986), 533–536.

[40] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the International World Wide Web Conference (WWW'01)*. ACM, 285–295.

[41] Guy Shani, David Heckerman, and Ronen I. Brafman. 2005. An MDP-based recommender system. *J. Mach. Learn. Res.* 6 (2005), 1265–1295.

[42] Bo Song, Yi Cao, Weifeng Zhang, and Congfu Xu. 2019. Session-based recommendation with hierarchical memory networks. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 2181–2184.

[43] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387* (2015).

[44] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS'15)*. 2440–2448.

[45] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 1441–1450.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS'17)*. 5998–6008.

[47] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the International Conference on Learning Representations (ICLR'18)*.

[48] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. ACM, 515–524.

[49] Meirui Wang, Pengjie Ren, Lei Mei, Zhumin Chen, Jun Ma, and Maarten de Rijke. 2019. A collaborative session-based recommendation approach with parallel memory modules. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. ACM, 345–354.

[50] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. ACM, 403–412.

[51] Qinyong Wang, Hongzhi Yin, Zhiting Hu, Defu Lian, Hao Wang, and Zi Huang. 2018. Neural memory streaming recommender networks with adversarial training. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'18)*. ACM, 2467–2475.

[52] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential recommender systems: Challenges, progress and prospects. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'19)*. 6332–6338.

[53] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. ACM, 165–174.

[54] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. 2020. Reinforced negative sampling over knowledge graph for recommendation. In *Proceedings of the Web Conference (WWW'20)*. ACM, 99–109.
[55] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xianling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20)*. ACM, 169–178.
[56] Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *Proceedings of the International Conference on Learning Representations (ICLR'15)*.
[57] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'19)*. AAAI Press, 346–353.
[58] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2019. Long- and short-term preference learning for next POI recommendation. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 2301–2304.
[59] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'19)*. 3940–3946.
[60] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the International Conference on Machine Learning (ICML'18)*. PMLR, 5449–5458.
[61] Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. 2020. Understanding negative sampling in graph representation learning. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'20)*. ACM, 1666–1676.
[62] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential recommender system based on hierarchical attention networks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'18)*. 3926–3932.
[63] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16)*. ACM, 729–732.
[64] Feng Yu, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: Target attentive graph neural networks for session-based recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20)*. ACM, 1921–1924.
[65] Fajie Yuan, Guibing Guo, Joemon M. Jose, Long Chen, Haitao Yu, and Weinan Zhang. 2016. LambdaFM: Learning optimal ranking with factorization machines using lambda surrogates. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'16)*. ACM, 227–236.
[66] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM'19)*. ACM, 582–590.
[67] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. 2018. Next item recommendation with self-attention. *arXiv preprint arXiv:1808.06414* (2018).
[68] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.* 52, 1 (2019), 5:1–5:38.
[69] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'13)*. ACM, 785–788.
[70] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'20)*. ACM, 1893–1902.

Received April 2021; revised July 2021; accepted September 2021