Graph-Augmented Co-Attention Model for Socio-Sequential Recommendation

Bin Wu, Xiangnan He, *Senior Member, IEEE*, Le Wu, *Member, IEEE*, Xue Zhang, and Yangdong Ye

***Abstract*—A sequential recommendation has become a hot research topic, which seeks to predict the next interesting item for each user based on his action sequence. While previous methods have made many efforts to capture the dynamics of sequential patterns, we contend that they still suffer from two inherent limitations: 1) they fail to model item transition patterns in an efficient and time-sensitive manner and 2) they are unaware of the importance of dynamically capturing social influence, resulting in suboptimal performance. We introduce a new concept dubbed *socio-sequential recommendation*, where the challenge mainly lies in dynamically modeling social influences and capturing item-to-item transition patterns in a time-sensitive manner. In light of this, we contribute a novel solution named GCARec (short for graph-augmented co-attention model), which takes into account the joint effect of dynamic sequential patterns and dynamic social influences. GCARec decomposes socio-sequential recommendation workflow into two steps. First, we adopt a light graph embedding module to model long-term user preference. Then, we propose a time-sensitive attention mechanism and a social-aware attention mechanism to capture dynamic patterns at sequential-level and social-level, respectively. Extensive experiments have been conducted on eight real-world datasets from different scenarios, demonstrating the superiority of GCARec against several state-of-the-art methods. The codes and datasets have been released at: https://github.com/wubinzzu/GCARec.**

***Index Terms*—Attention mechanisms, graph convolutional networks, sequential recommendation, social influence.**

I. Introduction

In THE era of information overload, the personalized recommendation has been an essential component in various commercial applications, which could push personalized information for customers and increase great profits for content providers. Learning users' preferences toward items from historical interaction records is the core of a personalized recommendation. Traditional recommender systems usually model and understand user behaviors in a static manner, ignoring the fact that user interests generally shift over time. In recent years, due to the high practicability, sequential recommendation greatly attracts academia and industry concerns. Many efforts toward this task have been devoted to developing various sequential models, such as Markov chains (MCs) techniques [1], [2], recurrent neural networks (RNNs) [3], [4], convolution neural networks (CNNs) [5], [6], and self-attention mechanisms [7], [8]. Nonetheless, the existing sequential recommenders mainly capture item transition patterns within individual sequence, which is not expressive enough to model short-term user interest especially for cold-start users.

With the popularity of social networks, lots of platforms like Douban and Epinions encourage users to enjoy online entertainment and exchange experiences with their friends. According to social influence and homophily theories [9], [10], social relations play a crucial role in recommending suitable items for each user. This refers to *social recommendation* [11], [12], [13], which exploits social relations to alleviate two key challenges of a modern recommendation: 1) *data sparsity* and 2) *cold start*. Recent years have witnessed the rapid evolution of this direction, spanning from the early factorization-based models [11], [14], to neural social recommenders [15], [16], and the recent graph neural networks (GNNs)-based methods [17], [18]. Despite their promising results, existing social recommendations forgo the sequential information of user behaviors, which may lead to suboptimal performance.

To our knowledge, very limited efforts have been devoted to leverage social relationships for improving the sequential recommendation performance. A recent one is DGRec [19] which models dynamic sequential patterns with a long short-term memory unit and captures static social influences with an attention-based GNN. Another attempt [20] adopts an attentive RNN to model dynamic user interests and an attention mechanism to capture social influences. Despite their effectiveness, these two approaches still suffer from the following shortages.

1) *Inefficient and Ineffective to Capture Sequential Patterns:* At each time step, such RNN-based recommenders rely heavily on the hidden state of the last step and the current state. In this case, they have difficulties learning nonadjacent transition patterns and utilizing parallel computation within a sequence. Furthermore, an underlying assumption made by them is to regard a user's action records as an ordered list, neglecting the time intervals between his last actions and the target

Manuscript received 3 June 2022; revised 31 December 2022; accepted 22 January 2023. Date of publication 21 February 2023; date of current version 16 June 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62102369 and Grant 61772475, and in part by the Postdoc Project of Henan Province under Grant 202102016. This article was recommended by Associate Editor J. Lu. *(Corresponding author: Yangdong Ye.)*

Bin Wu, Xue Zhang, and Yangdong Ye are with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China (e-mail: wubin7019088@gmail.com; zxue20191028@gmail.com; yeyd@zzu.edu.cn).

Xiangnan He is with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: xiangnanhe@gmail.com).

Le Wu is with the School of Computer and Information, Hefei University of Technology, Hefei 230009, Anhui, China (e-mail: lewu.ustc@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2023.3242308.

Digital Object Identifier 10.1109/TSMC.2023.3242308

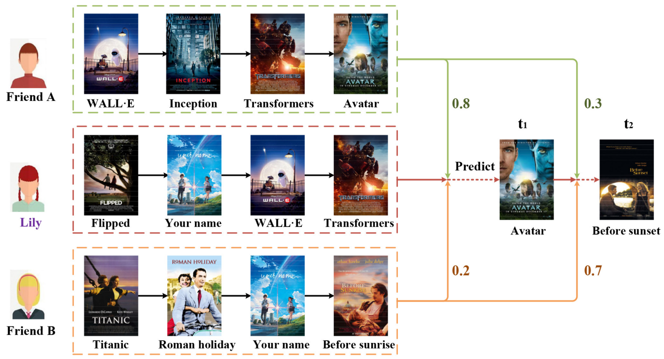


Fig. 1. Example to illustrate the dynamic social influence of the user's friends at different time steps.

item. Intuitively, the closer an item that the user has visited is to his current time, the more important this item is for inferring the next action.

2) *Neglecting the Importance of Dynamically Capturing Social Influence:* They fail to explicitly explore the dynamic social influences among different users, which may result in the suboptimal performance. To illustrate this limitation, Fig. 1 displays a toy example in the movie scenario. At time step t_1, Lily is looking for a science fiction movie. Two of her friends: A and B, have disparate tastes. Considering this context, Lily may be more influenced by friend A, e.g., decides to watch the movie "Avatar." Meanwhile, at time step t_2 (e.g., a Saturday night), she wants to see a romance movie with her boyfriend. In this case, it makes sense that friend B may have a higher influence strength for Lily as she is generally interested in this topic.

Toward this end, we introduce a new concept, i.e., *socio-sequential recommendation*. Herein socio refers to social space and sequential refers to temporal space. A system uses socio-sequential reasoning to solve the personalized recommendation problem by simulating how user interests are evolving with social influences and temporal factors. Furthermore, we contribute a novel recommender called graph-augmented co-attention model (GCARec) for the new problem. GCARec decomposes socio-sequential recommendation workflow into two steps. First, we adopt a light graph embedding module to model long-term user preference. Then, according to transition relationships amongst items and social relationships amongst users, we develop a co-attention module to capture dynamic sequential patterns and dynamic social influences. To be specific, a time-sensitive attention mechanism is devised to capture the complicated item transition patterns. On top of this, we further develop a social-aware attention mechanism to measure the significance of each friend *w.r.t.* the current user tastes. To sum up, the contributions of our work are four-fold.

1) We introduce a new concept: socio-sequential recommendation. To the best of our knowledge, this article is the first to highlight the importance of simultaneously exploring dynamic sequential patterns and dynamic social influences.
2) We contribute a novel solution dubbed GCARec, which seamlessly integrates social network information and time-sensitive sequential information into a unified framework.
3) We develop a co-attention module, which could select the informative ones from the recent items and differentiate which friends are more significant for inferring the next action.
4) Extensive experiments are conducted to evaluate our solution under a wide spectrum of recommendation scenarios, demonstrating that GCARec achieves substantial improvements over several competitive methods. Further studies verify the explainability of our model benefits brought by the co-attention module.

The remainder of this work is organized as follows. Section II reviews several highly relevant previous studies. In Section III, we formulate our problem to be solved in this article and present a graph-augmented co-attention model. Afterward, we conduct experiments to justify the superiority of GCARec on four real-world datasets in Section IV. Finally, Section V presents the conclusion and discusses future work.

II. Related Work

Our work is highly relevant to three active research topics: 1) general recommendation; 2) social recommendation; and 3) sequential recommendation.

A. General Recommendation

At the heart of a modern recommendation is to learn the representations (*also known as* embeddings) of users and items from historical behaviors [21], [22]. From the perspective of representation learning, previous studies on general recommendation can be roughly categorized into: *instance-level* methods that regard each user–item action record as an isolated data instance, and *graph-level* methods that reorganize users' historical behaviors as a bipartite graph. As one of the former methods, matrix factorization (MF) [23], [24] projects each user (item) into an embedding and employs a dot product as the interaction function. Several follow-on studies [21], [25], [26] replace an inner product with more expressive interaction functions for learning better representations. For instance, the convolutional collaborative filtering method [25] replaced an inner product in MF with an outer product and employed a CNN to capture the user–item interaction relationships; the factorized metric learning framework [26] models user preferences and captures the user–item proximity in a joint metric space. As one of the graph-level methods, the neural graph collaborative filtering (NGCF) framework [27] explicitly exploited high-order connectivity to refine the user and item embedding learning. Subsequently, He et al. [22] proved that the nonlinear feature transformation in NGCF had a negative effect on the recommendation performance and devised a light graph convolution network.

Our solution differs from the above methods in two points. On the one hand, they model user behaviors in a static manner and fail to capture short-term user interest, which may result in the suboptimal performance. On the other hand, these works forgo the social influences among different users, which is insufficient to obtain high-quality user representations especially for inactive users. We seek to solve the two defects with a co-attention module, which explores the joint effect of dynamic sequential patterns and dynamic social influences.

B. Social Recommendation

Living in the social communities, people usually seek suggestions from their friends before making a decision. As a matter of fact, some works have converged that user preference is impacted by his neighbors with homophily and social influence theories [9], [10]. In recent years, the social recommendation has already become a popular research direction by exploiting social influence among users. To implement this idea, factorization-based works [13], [14], [28] represent social connections among users as a co-occurrence matrix, which is decomposed as well as user preference matrix; regularization-based approaches [11], [12], [15] guide the user embedding learning by utilizing a graph Laplacian regularization on the social network; sampler-enhanced approaches [29], [30] modify the fundamental assumption of the Bayesian personalized ranking (BPR) framework by modeling social relationships.

Recently, inspired by graph representation learning, a trend in social recommendation augments the user embedding learning with GNNs. Specifically, Wu et al. [17] formulated the neural influence diffusion framework, which adopts two graph convolution layers to simulate the social influence propagation process. Later on, Liu et al. [31] proposed a high-order social recommender to exploit the "word-of-mouth" influence propagation in the social media. Recently, Wu et al. [18] formalized an efficient adaptive graph convolutional network (EAGCN), which jointly explores high-order neighbors reflected in two interaction graphs. Although great progress has been obtained, few works have explicitly explored the dynamic social influences, which is the major concern of this article.

C. Sequential Recommendation

A sequential recommendation has greatly attracted academia and industry concerns due to its practicability in various commercial applications [32], [33]. The early pioneer work for sequential recommendation is the FPMC method [1], which combines MF to capture long-term user preference and personalized MCs to capture sequential effects. Since it was introduced, FPMC has witnessed several evolutions, including translation-oriented recommender (TransRec) [2], attentive translation model (ATM) [34], multiorder attentive neural method [35], and self-attention-based sequential recommender (SASRec) [7], among others. Borrowing from sequence modeling in natural language processing, RNN-based methods [3], [4] have also been introduced to capture sequential user behaviors. For instance, Hidasi et al. [3] adopted the gated recurrent unit to predict the probability of an item being the next one. Furthermore, several CNN-based methods have shown competitive performance in dense datasets. In particular, Tang and Wang [5] developed a method called Caser that departs from the RNN module and captures sequential dynamics by adopting horizontal and vertical convolutional filters. Subsequently, Yan et al. [6] extended the Caser architecture by a 2-D convolutional network to learn local features and aggregated interactions among nonadjacent items in a feedforward manner. These methods only utilize sequential patterns without considering the social influences among different users, and thus they are insufficient for socio-sequential recommendation. Currently, very limited attempts exist in combining social relationships with a sequential recommendation to date. As an early attempt, Sun et al. [20] proposed the attentive recurrent social recommender, which employs an attentive RNN to model dynamic user preference and a standard attentive operation to capture social influences. The solution that is closest to this article is DGRec [19], which introduces a dynamic GNN for session-based social recommendation; it models dynamic sequential patterns with an RNN and captures static social influences with an attention-based GNN.

Despite effectiveness, we contend that the aforementioned methods still suffer from three defects: 1) they usually capture long-term user preference with a simple and crude embedding, which may be not sufficient to yield satisfactory long-term user representations; 2) such methods regard a user's action sequence as an ordered list, neglecting the time between his last actions and the target item; and 3) they capture social influence in a static way, which is suboptimal and nonextendable in real-world scenarios.

III. Problem Formulation and Proposed Model

In this section, we contribute a new model named GCARec, which explores the dynamics of sequential patterns and social influences for socio-sequential recommendation. First, we provide the formalization of our problem to be solved in this work. Then, we elaborate on how GCARec models long-term user preference and short-term user interest. As illustrated in Fig. 2, there are four parts in our architecture: 1) embedding initialization; 2) linear graph convolution module; 3) co-attention module; and 4) model prediction. Finally, we provide the model size and discuss the time complexity.

A. Problem Formulation

In this work, all vectors and matrices are denoted by bold letters (e.g., $\boldsymbol{e}$ and $\boldsymbol{W}$). We use uppercase calligraphic letters to denote sets (e.g., $\mathcal{U}$). For ease of presentation, several main symbols and the corresponding explanations are put into Table I.

Typically, we have the user set $\mathcal{U}$ and the item set $\mathcal{I}$. Given the target user $u \in \mathcal{U}$, we represent his sequential action records as $\mathcal{S}^u = \{\mathcal{S}_1^u, \ldots, \mathcal{S}_{t-1}^u, \ldots, \mathcal{S}_{|\mathcal{S}^u|}^u\}$, where $\mathcal{S}_{t-1}^u \in \mathcal{I}$ and $|\mathcal{S}^u|$ denotes the number of user $u's$ actions. Apart from his action sequence $\mathcal{S}^u$, we have social relationships $\mathcal{F}^u = \{(u, v)|u, v \in \mathcal{U}\}$, where the pair (u, v) indicates that there is a social connection from user u to user v.

Modeling and understanding the complicated interaction relationships between users and items lie at the core of a personalized recommendation. Specifically, modeling user-to-item relationships could answer the question "What kind of item would the target user like?"; capturing item-to-item transition patterns helps find out which type of shoes match the

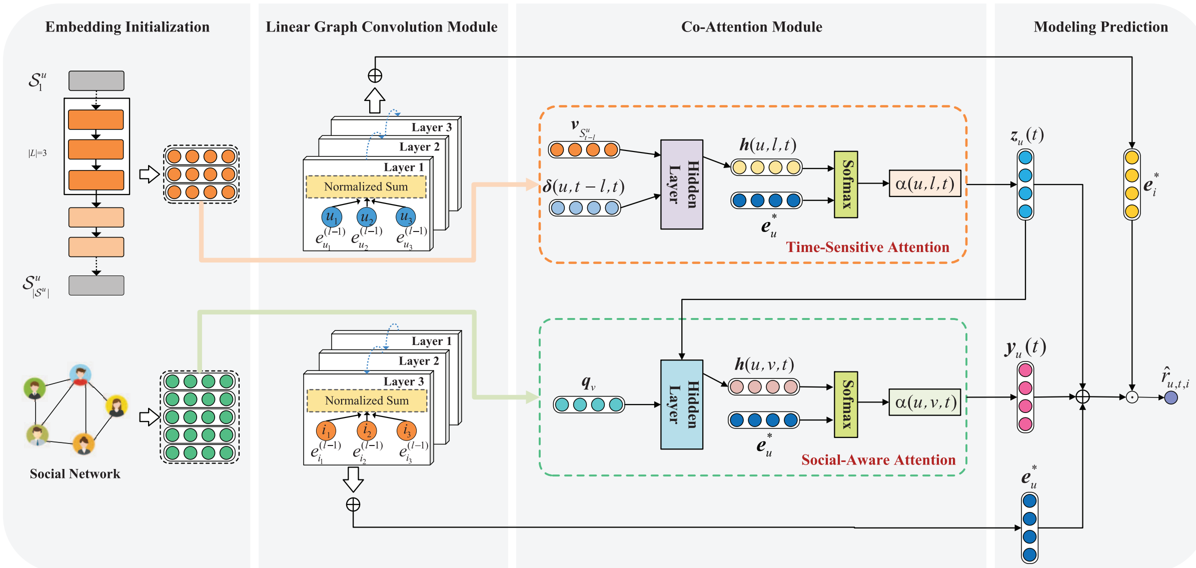


Fig. 2. Illustration of our GCARec framework.

TABLE I
NOTATIONS

Symbol	Explanation
$\mathcal{U}$,$\mathcal{I}$	User set and item set
u,j,t	User $u \in \mathcal{U}$, item $j \in \mathcal{I}$, a specific time step
$\mathcal{S}^u$	The set of items that are interacted by user u
$\mathcal{F}^u$	The set of friends that user u has connected with
$\mathcal{S}^u_{t-1}$	An item that is interacted by user u at time step $t-1$
$\boldsymbol{E}$	Initialized embeddings associated with all users and items
$\boldsymbol{V}$	Sequential embeddings associated with items
$\boldsymbol{e}_u$	An embedding associated with user u
$\boldsymbol{e}_i$	An embedding associated with item i
D	The dimensionality of an embedding
K	The number of graph convolution layers
L	The length of Markov chains
$\hat{r}_{u,t,i}$	Likelihood that user u will visit item i at time step t

skirts just bought; and understanding user-to-user interactions helps explain why the user favors a certain item at a specific time stamp. In this work, we concentrate on the item recommendation problem, where we observe whether a user visited an item. Formally, our task can be formalized as follows.

Definition 1 (Socio-Sequential Recommendation):

Input: User set $\mathcal{U}$, item set $\mathcal{I}$, each user $u \in \mathcal{U}$ who has an action sequence $\mathcal{S}^u = \{\mathcal{S}^u_1, \mathcal{S}^u_2, \ldots, \mathcal{S}^u_{|\mathcal{S}^u|}\}$, and a friend set $\mathcal{F}^u \subseteq \mathcal{U}$.

Output: Derive a top-N list of items from $\mathcal{I}/\mathcal{S}^u$ that user u has not visited before but may interact with them in the future.

In Fig. 3, we provide a visual comparison of socio-sequential recommendation and alternative recommendation tasks.

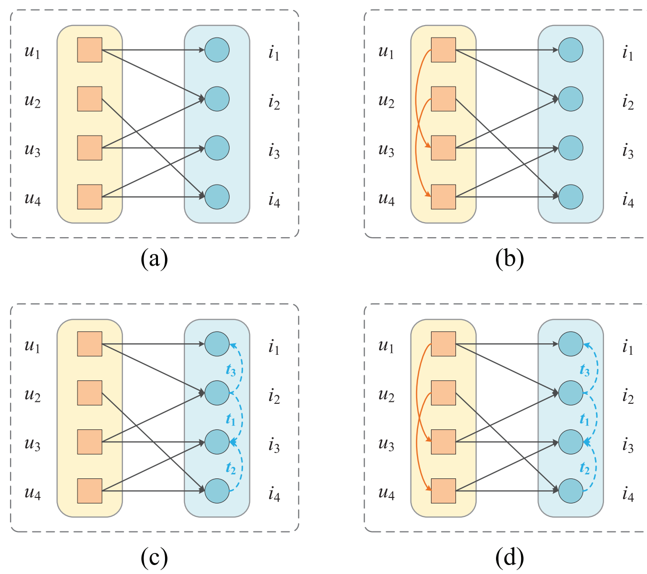


Fig. 3. Visual comparison of general, social, sequential, and socio-sequential recommendations.

B. Embedding Initialization

Similar to mainstream sequential recommendations [7], [36], [37], we apply one-hot encoding for each user/item and encode them with a dense vector. In particular, given a user u and an item i, this module produces their embeddings as $\boldsymbol{e}_u^{(0)} \in \mathbb{R}^D$ and $\boldsymbol{e}_i^{(0)} \in \mathbb{R}^D$, where D represents the dimensionality of an embedding. Note that the superscript "(k)" denotes a layer index, indicating the embeddings are generated from the kth graph convolution layer; the superscript "(0)" is the initialized embeddings. Following DGRec [19] and hierarchical gating networks (HGNs) [37], we additionally encode item i with an embedding $\boldsymbol{v}_i \in \mathbb{R}^D$ to capture sequential patterns, and each user with a vector $\boldsymbol{q}_u \in \mathbb{R}^D$ to capture social influence. As such, we could get three embedding matrices for all items and users

$$\begin{aligned}\boldsymbol{E} &= \left[\boldsymbol{e}_{u_1}^{(0)}, \ldots, \boldsymbol{e}_{u_{|\mathcal{U}|}}^{(0)}, \boldsymbol{e}_{i_1}^{(0)}, \ldots, \boldsymbol{e}_{i_{|\mathcal{I}|}}^{(0)}\right] \\ \boldsymbol{Q} &= \left[\boldsymbol{q}_{u_1}, \ldots, \boldsymbol{q}_{u_{|\mathcal{U}|}}\right], \boldsymbol{V} = \left[\boldsymbol{v}_{i_1}, \ldots, \boldsymbol{v}_{i_{|\mathcal{I}|}}\right] \qquad (1)\end{aligned}$$

where $[\cdot]$ denotes the concatenation operation. In the following sections, we present how to refine the above embeddings via other modules.

C. Linear Graph Convolution Module

In previous sequential recommenders, such as FPMC [1], SASRec [7], and HGN [37], the initialized user embeddings [i.e., $\boldsymbol{e}_u^{(0)}$] are directly utilized to capture long-term user preferences. Despite the simplicity, we contend that such a manner is not sufficient to achieve satisfactory recommendation results. The key reason is that these methods only explore the observed direct interaction relationships and fail to capture high-order proximity between users and items. In real-world applications, the main obstacle of a modern recommender system lies in the data sparsity problem. Without explicitly exploiting the collaborative signals, it is nontrivial to get high-quality users' representations especially for inactive users. Luckily, recent developments about GNN-based methods [22], [38], [39] shine some light on alleviating this issue. Inspired by LightGCN [22], we refine the initialized embedding matrix $\boldsymbol{E}$ by a light graph convolution module. To be specific, such module includes two parts: 1) neighbor aggregation, which learns each node embedding by linearly aggregating the features of its neighbors and 2) layer combination, which gathers the more representative information from multihop neighbors and helps mitigate the oversmoothing problem. In what follows, we elaborate these two ingredients.

1) Neighbor Aggregation: To explicitly exploit collaborative signals between users and items, GNN-based recommenders usually gather features from their neighbors by performing a neighbor aggregation operation. Taking a user node as an example, the kth layer of a graph propagation module can be abstracted as

$$\boldsymbol{e}_u^{(k)} = f_{\text{propagate}}\left(\boldsymbol{e}_u^{(k-1)}, \left\{\boldsymbol{e}_i^{(k-1)} | i \in \mathcal{S}_u\right\}\right) \tag{2}$$

where $f_{\text{propagate}}$ is a propagation function. Most of the GNN-based methods can be achieved under this architecture by adopting different aggregators, such as weighted summation [40], bi-interaction operation [27], and the self-attention mechanism [41]. Recently, several studies [22], [42] have pointed out that the two common operations in $f_{\text{propagate}}$—nonlinear activation and feature transformation—have no positive effect for recommender systems. In light of this, we apply the simplest operation—linear aggregator. Formally, we rewrite (2) as follows:

$$\begin{aligned}\boldsymbol{e}_u^{(k)} &= \sum_{i \in \mathcal{S}_u} \frac{1}{\sqrt{|\mathcal{S}_u||\mathcal{N}_i|}} \boldsymbol{e}_i^{(k-1)} \\ \boldsymbol{e}_i^{(k)} &= \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i||\mathcal{S}_u|}} \boldsymbol{e}_u^{(k-1)}\end{aligned} \tag{3}$$

where $|\mathcal{N}_i|$ denotes the users that have interacted with item i.

2) Layer Combination: After being propagated through the K times, we obtain two embedding sets for user u and item i, namely $\{\boldsymbol{e}_u^{(0)}, \ldots, \boldsymbol{e}_u^{(K)}\}$ and $\{\boldsymbol{e}_i^{(0)}, \ldots, \boldsymbol{e}_i^{(K)}\}$. To avoid oversmoothing, we average these embeddings to attain the final representations

$$\boldsymbol{e}_u^* = \frac{1}{K+1} \sum\nolimits_{k=0}^{K} \boldsymbol{e}_u^{(k)}; \quad \boldsymbol{e}_i^* = \frac{1}{K+1} \sum\nolimits_{k=0}^{K} \boldsymbol{e}_i^{(k)}. \tag{4}$$

D. Co-Attention Module

As the long-term user preference modeling has been discussed in Section III-C, how to capture short-term user interest is a key for improving the recommendation performance. Toward this goal, we explore dynamic patterns from two levels, i.e., sequential and social. As described in the introduction, the former aims to select the informative items that could better reflect the user's current taste. Continuing the earlier example, at time step t_1, Lily may pay more attention on the movies "WALL·E" and "Transformers" than the movie "Flipped." Moreover, the closer an item that the user has visited is to his current time, the more important this item is to predict the next action. On the other hand, the latter aims to capture dynamic social influences among different users. For instance, at time step t_1, friend A may have a higher influence strength for Lily; at other time steps (e.g., t_2), this case is quite the opposite. In light of this, we design a co-attention module to model short-term user interest, including the time-sensitive attention mechanism that captures dynamic sequential patterns and the social-aware attention mechanism that captures dynamic social influences. More details are explained as follows.

1) Time-Sensitive Attention: For each user, we extract successive L items as input and the next item i as the predictive target, where $\{\mathcal{S}_{t-L}^u, \ldots, \mathcal{S}_{t-1}^u\}$ forms the L-order MCs. Formally, the user u's taste at time step t could be abstracted as

$$z_u(t) = f_{\text{agg}}\left(\mathcal{S}_{t-L}^u, \ldots, \mathcal{S}_{t-1}^u\right) \tag{5}$$

where f_{agg} is an aggregation function, which aims to integrate the L-order MCs into a vector. Careful readers may realize that the maximum or average operation can be directly applied. Unfortunately, these two strategies fail to differentiate the importance of each candidate item for inferring the next action. For instance, the user u's watch history is $\mathcal{S}^u = \{(i_1, \textit{Fed}\ 26\text{th}), (i_2, \textit{Fed}\ 26\text{th}), (i_3, \textit{Fed}\ 26\text{th})\}$; and the user v's watch history is $\mathcal{S}^v = \{(i_1, \textit{Fed}\ 16\text{th}), (i_2, \textit{Fed}\ 20\text{th}), (i_3, \textit{Fed}\ 22\text{nd})\}$. Although the interactions of the two users have the same sequential order, it is reasonable to transmit more information from $\mathcal{S}^u$ to u's short-term interest than from $\mathcal{S}^v$ to v's. That is because user u watched these movies within one day, while user v produced these actions in one week. Nevertheless, most of the recent recommenders (e.g., HGN [37], SASRec [7], and DGRec [19]) regard the two cases as the same because they only focus on the sequential order (i.e., relative position). To tackle this problem, we devise a time-sensitive attention mechanism, which is a personalized and time interval-aware manner, to determine the contribution ratio of each item from L-order MCs. Given the current timestamp T_t^u of user u, we introduce a time interval feature as follows:

$$\boldsymbol{\delta}(u, t-l, t) = \phi\left(\boldsymbol{w}_\delta \log\left(T_t^u - T_{t-l}^u\right) + \boldsymbol{b}_\delta\right) \tag{6}$$

where $\boldsymbol{w}_\delta, \boldsymbol{b}_\delta \in \mathbb{R}^D$ and $\phi(\cdot)$ is the *tanh* function. The time interval feature $\boldsymbol{\delta}(u, t-l, t)$ encodes the absolute temporal distance between time step $t-l$ and time step t. On top of this, the attention vector is formalized as follows:

$$\boldsymbol{h}(u, l, t) = \phi\Big(\boldsymbol{v}_{\mathcal{S}^u_{t-l}} \boldsymbol{W}_1 + \boldsymbol{\delta}(u, t-l, t)\boldsymbol{W}_2 + \boldsymbol{b}\Big) \quad (7)$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{D\times D}$ and $\boldsymbol{b} \in \mathbb{R}^D$ are the learnable parameters to distill useful information. To reduce the model parameters and achieve the personalized schema, we employ the embedding $\boldsymbol{e}^*_u$ as the context vector. Hereafter, we obtain the weight value $\alpha(u, l, t)$ as the normalized affinity between $\boldsymbol{h}(u, l, t)$ and $\boldsymbol{e}^*_u$ by applying the softmax function

$$\alpha(u, l, t) = \frac{\exp\big(\boldsymbol{e}^*_u \boldsymbol{h}^\top(u, l, t)\big)}{\sum_{g=1}^{L} \exp\big(\boldsymbol{e}^*_u \boldsymbol{h}^\top(u, g, t)\big)}. \quad (8)$$

After obtaining the attention weights, we can rewrite (5) as follows:

$$\boldsymbol{z}_u(t) = \sum_{l=1}^{L} \alpha(u, l, t) \cdot \boldsymbol{v}_{\mathcal{S}^u_{t-l}}. \quad (9)$$

2) Social-Aware Attention: Successfully capturing social influence is a key for a socio-sequential recommendation, since dynamically modeling social relations among different users could help explain why the user favors a certain item at a specific timestamp. Nevertheless, most of the recent works either do not exploit social relations (e.g., ATM [34], SR-GNN [43], and SASRec [7]) or capture social influence in a time-insensitive manner (e.g., DiffNet [17] and EAGCN [18]). Indeed, it is nontrivial to do it well. Let us continue the earlier example in Fig. 1. When Lily considers whether she watches the movie "Avata," she might put more attention weight on friend A; this is because friend A has just seen the movie and also watched lots of other science fiction movies like "Inception" and "Transformer." At time step t_2, the influence of friend A is relatively low. The reason may be that friend A watched very few romance movies in her consumed records. Therefore, friend A has less contribution on the target user decision making. As such, we settle the above challenge by borrowing the idea from attention mechanism [7], [34], a natural way that explores user's varying attentions on his/her friends. In this regard, we devise a social-aware attention mechanism to distinguish the significance of each friend for predicting the next action of a given user. The attention vector is formalized as follows:

$$\boldsymbol{h}(u, v, t) = \phi\big(\boldsymbol{z}_u(t)\boldsymbol{W}_3 + \boldsymbol{q}_v \boldsymbol{W}_4 + \boldsymbol{b}'\big) \quad (10)$$

where $\boldsymbol{W}_3, \boldsymbol{W}_4 \in \mathbb{R}^{D\times D}$ and $\boldsymbol{b}' \in \mathbb{R}^D$ are the learnable parameters. Same as (8), we also apply the embedding $\boldsymbol{e}^*_u$ as the context vector. Hereafter, we obtain the weight value $\alpha(u, v, t)$ as the normalized affinity between $\boldsymbol{h}(u, v, t)$ and $\boldsymbol{e}^*_u$ by adopting the softmax function

$$\alpha(u, v, t) = \frac{\exp\big(\boldsymbol{e}^*_u \boldsymbol{h}^\top(u, v, t)\big)}{\sum_{g\in\mathcal{F}^u} \exp\big(\boldsymbol{e}^*_u \boldsymbol{h}^\top(u, g, t)\big)}. \quad (11)$$

After obtaining the social-aware attention weights, we can compute the influence strength of the user's friends as follows:

$$\boldsymbol{y}_u(t) = \sum_{v\in\mathcal{F}^u} \alpha(u, v, t) \cdot \boldsymbol{q}_v. \quad (12)$$

E. Model Prediction and Training

1) Model Prediction: After performing the linear graph convolution module and the co-attention module, we get the long-term user representation $\boldsymbol{e}^*_u$ and sequential dynamics $\boldsymbol{z}_u(t)$ and social dynamics $\boldsymbol{y}_u(t)$. Finally, we can compute the user u's preference on item i

$$\hat{r}_{u,t,i} = \mu_i + \big(\boldsymbol{e}^*_u + \boldsymbol{z}_u(t) + \boldsymbol{y}_u(t)\big)\top \cdot \boldsymbol{e}^*_i \quad (13)$$

where μ_i captures the overall item popularity.

2) Model Training: Given user u, his friends $\mathcal{F}^u$, and the recent actions $\{\mathcal{S}^u_{t-L}, \ldots, \mathcal{S}^u_{t-1}\}$, we can compute the probability of item i being the next interacted one as follows:

$$Pr\big(i|u, \mathcal{S}^u_{t-1}, \ldots, \mathcal{S}^u_{t-L}\big) = \sigma\big(\widehat{r}_{u,t,i}\big) \quad (14)$$

where $\sigma(x) = 1/(1+e^{-x})$. Consequently, we train GCARec by optimizing the following objective:

$$\begin{aligned}\mathcal{L} = -\sum\nolimits_{u\in\mathcal{U}} \sum\nolimits_{t=L+1}^{|\mathcal{S}^u|} \log\big(\sigma\big(\widehat{r}_{u,t,i}\big)\big) \\ + \sum\nolimits_{j\in\mathcal{I}\setminus\mathcal{S}^u} \log\big(1-\sigma\big(\widehat{r}_{u,t,j}\big)\big) + \lambda\|\Theta\|^2\end{aligned} \quad (15)$$

where λ is a regularization hyperparameter, and Θ is the set of the model parameters. We adopt the uniform sampling strategy to select the negative instance j from unobserved items $\mathcal{I}\setminus\mathcal{S}^u$.

F. Model Analysis

1) Model Size: The model parameters of our proposed GCARec method stem from two portions: $\Theta_1 = \{\boldsymbol{E}, \boldsymbol{V}, \boldsymbol{Q}, \boldsymbol{\mu}\}$ and $\Theta_2 = \{\boldsymbol{w}_\sigma, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4, \boldsymbol{b}, \boldsymbol{b}'\}$. The former is proportional to the numbers of users and items [i.e., $2(|\mathcal{U}|+|\mathcal{I}|)D+|\mathcal{I}|$]. The latter is completely independent of the numbers of users and items (i.e., $4D^2+3D$). Analytically, the model size of GCARec is comparable to the state-of-the-art method HGN [i.e., $(|\mathcal{U}|+2|\mathcal{I}|+L+1)D+|\mathcal{I}|+2D^2$], and lighter than the pioneer model FPMC [i.e., $(|\mathcal{U}|+3|\mathcal{I}|)D$]. To be specific, on the Epinions dataset (12.7K users and 209.7K items), when D is set to 64, FPMC and HGN utilize 40.12M and 27.22M parameters respectively, while GCARec has 28.02M parameters.

2) Time Complexity: It mainly includes three parts. For the first module, obtaining the embeddings $\boldsymbol{e}^*_u$ and $\boldsymbol{e}^*_i$ has computation complexity $O(DK|\mathcal{R}^+|)$, where $|\mathcal{R}^+|$ represents the scale of user–item interactions. For the time-sensitive attention mechanism, the time cost of calculating $\boldsymbol{h}(u, l, t)$ is $O(D^2)$. As the denominator of the softmax function in (8) needs to traverse over L-order MCs, the time cost of computing an $\alpha(u, l, t)$ is $O(D^3L)$. For all users, the time overhead of this part is $O(|\mathcal{U}|D^3L)$. Analogously, for the social-aware attention mechanism, the time cost is $O(D^3|\mathcal{U}||\tilde{\mathcal{F}}|)$, where$|\tilde{\mathcal{F}}|$ denotes the average number of social relations received by a user. In a nutshell, the overall time complexity of GCARec is $O(D^3|\mathcal{U}|(L+|\tilde{\mathcal{F}}|)+DK|\mathcal{R}^+|)$.

TABLE II
STATISTICS OF EIGHT DATASETS

Dataset	#Users	#Items	#Actions	#Relations	Sparsity
Gowalla	5,303	32,361	357,144	80,081	99.79%
Ciao	6,596	107,320	274,065	114,893	99.96%
Yelp	9,776	18,984	442,325	638,479	99.76%
Epinions	13,012	214,793	562,208	352,209	99.97%
Dianping	28,281	10,904	1,125,746	256,570	99.64%
Movie	15,538	59,672	3,303,012	78,870	99.64%
Book	5,409	93,358	393,691	22,084	99.92%
Music	4,973	82,273	466,438	20,398	99.89%

IV. EXPERIMENTS

In this section, we perform experiments on eight public datasets to demonstrate the effectiveness and rationality of our solution. Through empirical evaluation, we intend to answer four questions.

RQ1 How does GCARec perform compared with several state-of-the-art recommendation approaches?

RQ2 How do different modules influence GCARec's results?

RQ3 How do different hyperparameter settings impact the recommendation performance?

RQ4 How does our co-attention module work?

A. Experimental Setup

1) Dataset Description: We perform experiments on a series of public datasets from different scenarios.

1) *Gowalla* [44] was collected from *gowalla.com*, involving lots of social relationships and check-ins of users at different locations from February 2009 to October 2010.
2) *Ciao* [45] was collected from a consumer review website, where users can provide product reviews and assign trust values to other users.
3) *Yelp* [44] was provided by the Yelp Challenge and includes an amount of restaurant businesses and ratings within different cities.
4) *Epinions* [29] was collected from a consumer review website, including an amount of online user ratings from January 2001 to November 2013.
5) *Dianping*[1] was crawled from a Chinese social network-based website Dianping, which contains an amount of consumed records of users in Shanghai and spans from April 2003 to November 2013.
6) *Douban* [19] was crawled from *www.douban.com*, where users could rate *movies*, *music*, and *books*.

For the above benchmarks, we transform user reviews as binary feedback and remove users that have less than 5 ratings and 2 social connections. Table II presents the detailed statistics of the filtered datasets.

2) Evaluation Metrics: Following previous studies [18], [22], and [27], we utilize the fold-out setting to evaluate the recommendation quality of competitors and our method. Specifically, we utilize the absolute timestamps to discriminate the sequence order of interactions and divide the historical sequence $\mathcal{S}^u$ into three portions: 1) the first 70% interactions as training data; 2) the next 10% interactions as the validation set; and 3) the remainder actions for testing. In our experiments, we evaluate all methods at two commonly ranking-oriented metrics: 1) *recall* and 2) *normalized discounted cumulative gain* (NDCG) [22], [46]. The detailed computations of Recall@N and NDCG@N are formalized as follows:

$$\text{Recall@N} = \frac{1}{|\mathcal{U}|}\sum_{u=1}^{|\mathcal{U}|}\frac{|l_{\text{rec}}^u \bigcap l_{\text{tes}}^u|}{|l_{\text{tes}}^u|}$$

$$\text{NDCG@N} = \frac{1}{|\mathcal{U}|}\sum_{u=1}^{|\mathcal{U}|}\frac{\sum_{a=1}^{\text{N}}\frac{2^{\delta u(a)}-1}{\log_2(a+1)}}{\sum_{a=1}^{\min\{\text{N},|l_{\text{tes}}^u|\}}\frac{1}{\log_2(a+1)}} \tag{16}$$

where l_{rec}^u is the top-N list and l_{tes}^u denotes the observed items that user u has visited in the test data. $\delta_u(a)$ is a binary indicator function that equals to 1 if the item at rank a is visited in the test data, 0 otherwise.

3) Comparing Recommendation-Related Methods: To justify the effectiveness of GCARec, we adopt eight representative methods as competitors, including general, social, and sequential recommendation methods.

1) *BPRMF:* It optimizes MF with the BPR loss function, which is a widely used pairwise recommender for the task of item ranking [47].
2) *LightGCN:* It is the recent GNN-based recommender [22], which adopts a light graph propagation module to refine the user and item embedding learning.
3) *EATNN:* This is a representative social-aware method [48], which presents a personalized manner to transfer the shared knowledge between the social domain and the item domain.
4) *EAGCN:* It is the recent state-of-the-art method for social recommendation [18], which employs a space-adaptive graph convolution module to jointly simulate the recursive diffusion process of user interest and social influence.
5) *FPMC:* It is the pioneering work for sequential recommendation [1], which unifies the strength of an MF model at modeling long-term user preference and the power of first-order MC at capturing item-to-item transitions.
6) *ATM:* It is an improved version of FPMC, which embeds all users and items into a transition space and designs a position-aware attention mechanism to explore the high-order MCs [34].
7) *HGN:* It is the recently introduced competitive solution for sequential recommendation, which embeds a hierarchical gating operation to model group-level sequential dynamics and adopts an item-level product module to discover the important items.
8) *DGRec:* This method is closest to our work [19], which presents a unified solution for session-based social recommendation; it first utilizes an RNN to model dynamic sequential patterns and then captures static social influences with a graph attention network.

4) Hyperparameter Settings: For BPRMF, FPMC, and ATM, we implemented them with a popular open-source

[1]https://lihui.info/data/list/

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT METHODS *w.r.t.* RECALL@10 AND NDCG@10. THE COLUMN "IMPROVE" STANDS FOR THE RELATIVE IMPROVEMENTS THAT GCAREC OBTAINS TO THE UNDERLINE RESULTS

Dataset	Metric@10	BPRMF	FPMC	DGRec	ATM	LightGCN	EATNN	EAGCN	HGN	GCARec	Improve
Gowalla	Recall	0.0788	0.0822	0.0833	0.0845	0.0844	0.0923	0.0971	0.0869	**0.1037**	6.79%
	NDCG	0.1194	0.1250	0.1341	0.1361	0.1330	0.1454	0.1496	0.1387	**0.1586**	6.01%
Ciao	Recall	0.0339	0.0352	0.0355	0.0359	0.0377	0.0391	0.0407	0.0372	**0.0426**	4.66%
	NDCG	0.0301	0.0318	0.0322	0.0327	0.0332	0.0339	0.0348	0.0336	**0.0362**	4.02%
Yelp	Recall	0.0351	0.0371	0.0385	0.0402	0.0455	0.0462	0.0475	0.0419	**0.0506**	6.52%
	NDCG	0.0346	0.0353	0.0358	0.0373	0.0455	0.0458	0.0477	0.0372	**0.0493**	3.35%
Epinions	Recall	0.0115	0.0125	0.0133	0.0148	0.0164	0.0161	0.0170	0.0154	**0.0189**	11.17%
	NDCG	0.0098	0.0106	0.0111	0.0122	0.0132	0.0134	0.0141	0.0139	**0.0157**	11.34%
Dianping	Recall	0.0294	0.0301	0.0319	0.0347	0.0320	0.0308	0.0353	0.0365	**0.0407**	11.50%
	NDCG	0.0252	0.0257	0.0272	0.0314	0.0268	0.0262	0.0279	0.0325	**0.0370**	13.84%
Movie	Recall	0.0209	0.0245	0.0259	0.0282	0.0227	0.0218	0.0232	0.0352	**0.0422**	20.17%
	NDCG	0.0388	0.0632	0.0673	0.0734	0.0501	0.0421	0.0434	0.0964	**0.1115**	15.66%
Book	Recall	0.0340	0.0344	0.0354	0.0390	0.0343	0.0341	0.0354	0.0468	**0.0513**	9.61%
	NDCG	0.0234	0.0270	0.0332	0.0294	0.0273	0.0315	0.0324	0.0437	**0.0457**	4.57%
Music	Recall	0.0274	0.0302	0.0310	0.0323	0.0319	0.0318	0.0342	0.0355	**0.0380**	7.04%
	NDCG	0.0290	0.0331	0.0352	0.0366	0.0341	0.0336	0.0374	0.0403	**0.0420**	4.21%

library NeuRec.[2] For LightGCN,[3] EATNN,[4] EAGCN,[5] HGN,[6] and DGRec,[7] we utilize the source codes released by the corresponding authors. For all methods, the learning rate is tuned in {0.0005, 0.001, 0.005, 0.01} and the regularization coefficient is chosen from $\{0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$. For LightGCN and EAGCN, we apply three embedding propagation layers as suggested by the authors. For EATNN, μ is chosen from {0.01, 0.03, 0.05, 0.07, 0.1, 0.3, 0.5}. For DGRec, the maximum length of the user action sequence is searched in {50, 100, 150, 200}. For HGN and ATM, we tune the hyperparameter L in {2, 3, 4, 5, 6, 7, 8, 9, 10}. For GCARec, we implement it with TensorFlow and optimize it with *Adam* optimizer. The effect of different hyperparameters L and K is investigated below.

B. Performance Comparison With State-of-the-Arts (RQ1)

1) Overall Comparison: We first provide the overall performance of GCARec and other methods in Table III. Moreover, we conduct the paired t-test between GCARec and the strongest results (highlighted in underline), indicating that the improvements are statistically stable and noncontingent (i.e., $p < 0.01$). From the table, we have seven findings.

1) BPRMF performs the worst in each of the eight recommendation scenarios. Despite the simplicity of one-hot encoding, it provides very limited information for user and item embedding learning. With the same interaction function and model parameters, LightGCN consistently surpasses BPRMF on eight datasets by a large margin. These results show that stacking multiple graph convolution layers could refine user and item representation learning.
2) EATNN generally achieves better results than BPRMF, indicating the benefits of explicitly exploiting social networks in the user embedding learning. Meanwhile, ECGCN further improves over EATNN on all datasets. Such improvements might be attributed by a dual-side graph propagation module, which could explicitly exploit collaborative signals and high-order social relations.
3) Compared with BPRMF, FPMC achieves significantly better results in all cases, demonstrating that capturing sequential patterns by first-order MC could improve the recommendation performance.
4) Compared to FPMC, the performance of DGRec indicates that leveraging social networks in a sequential recommender has a positive effect on inferring the next item. However, the empirical results show limited benefits *w.r.t.* Rcall@10 and NDCG@10; the key reason is that DGRec fails to dynamically explore social influence among users for recommender systems.
5) ATM achieves better performance than FPMC on all datasets. It makes sense that since ATM employs a position-aware attention strategy to model high-order MCs, while FPMC only adopts the latest one action to encode short-term dynamics in sequences. This conclusion is consistent with ATM's finding [34].
6) HGN consistently outperforms ATM in all cases. To be specific, HGN improves over ATM *w.r.t.* NDCG@10 by 3.50%, 47.13%, 48.63%, and 10.10% in Dianping, Movie, Book, and Music, respectively. This is because HGN captures both group-level and point-level sequential patterns by designing a hierarchical gating module, whereas ATM neglects the effect of group-level sequential patterns.
7) GCARec consistently yields the best performance under a wide spectrum of recommendation scenarios. For instance, GCARec outperforms the strongest results *w.r.t.* NDCG@10 by 11.34%, 13.84%, 15.66%, and 4.57% in Epinions, Dianping, Movie, and Book, respectively. We contend that such remarkable performance gains stem from three portions: a) by embedding a graph convolution module, GCARec could better capture long-term user preference, than other sequential recommenders that generate the long-term representation of

[2] https://github.com/wubinzzu/NeuRec
[3] https://github.com/kuandeng/LightGCN
[4] https://github.com/chenchongthu/EATNN
[5] https://github.com/wubinzzu/EAGCN
[6] https://github.com/allenjack/HGN
[7] https://github.com/DeepGraphLearning/RecommenderSystems

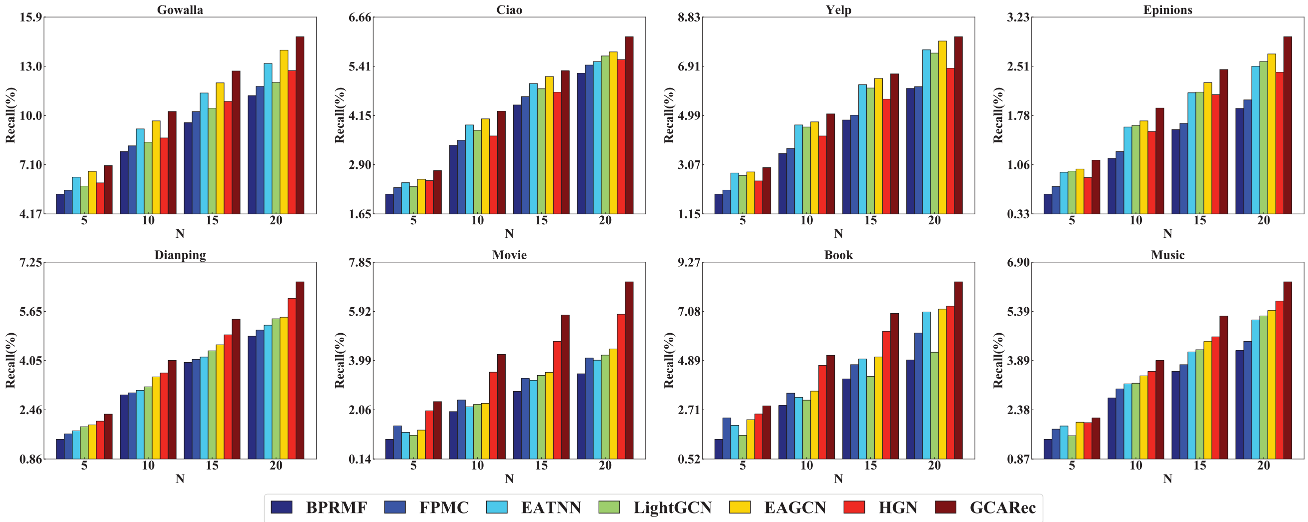


Fig. 4. Performance comparison on varying size of the top-N list.

a user only by an initial vector; b) it is necessary to explore the joint effect of dynamic sequential patterns and dynamic social influences; and c) our co-attention module could capture the varying importance of historical items and differentiate the contribution ratio of each friend at different times.

So far, Table III has presented the top-10 results. Nevertheless, we still cannot confirm that how GCARec and other recommenders perform under different lengths of the top-*N* list (e.g., [5, 10, 15, 20, 25]). Toward this end, we display the Recall@N of our solution versus other competitors in Fig. 4, and can observe.

1) GCARec and other methods present the same trend in terms of Recall@N on eight benchmarks. Precisely, with the increase of *N*, the Recall values gradually increase.
2) EATNN consistently achieves better results than BPRMF. It demonstrates that only utilizing user–item interaction information is not sufficient to establish informative embeddings of all users and items. Meanwhile, EAGCN further improves over EATNN by explicitly exploiting high-order connectivity reflected in both user–item graph and social graph.
3) FPMC achieves significant improvements over BPRMF in all cases. This again verifies the necessity of modeling sequential patterns for recommender systems. As the variant of FPMC, HGN obtains substantial performance gains by exploiting high-order MCs.
4) Our solution can always achieve the substantial improvement over other methods in all cases. This demonstrates that it is very important to dynamically model social influences and adaptively capture sequential patterns for providing a high-quality top-*N* recommendation list.

2) Performance Comparison Under Different Interaction Sparsity Levels: Except for the overall performance in Table III and Fig. 5, we also investigate the recommendation quality of GCARec and other competitors under different interaction sparsity levels. Following previous studies [18] and [27], we divide all users into four groups and ensure that the total interaction records over users in each group are the same. Here, we choose two sparser scenarios (i.e., Ciao and Epinions) and two denser datasets (i.e., Gowalla and Movie). Take the Ciao dataset as a case, the length per user sequence in each group is $\{\leq 38, \leq 92, \leq 260, \leq 1650\}$, and ≤ 92 denotes each user in this group has at least 38 actions and less than 92 actions. In Fig. 5, the background histograms present the user distribution of each dataset and the lines show the results *w.r.t.* NDCG@10 on each group. From the results, we have two key observations.

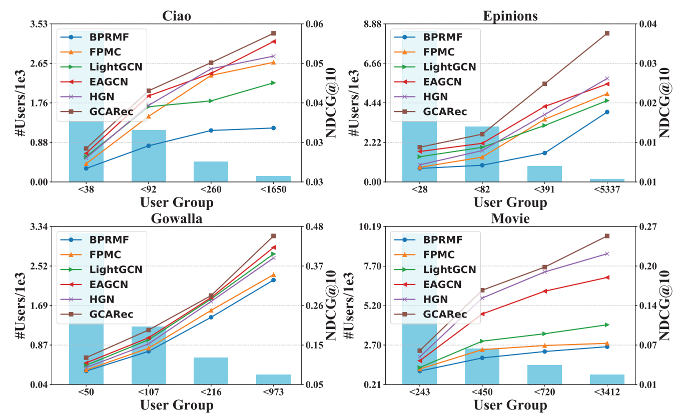


Fig. 5. Performance comparison under different user groups.

1) When users have much less actions, LightGCN consistently surpasses BPRMF on four datasets; meanwhile, EAGCN obtains further performance gains over LightGCN. It shows that exploring high-order collaborative signals and social relations is beneficial to alleviate the main obstacle of modern recommender systems—the data sparsity problem.
2) With the increase of user actions, the NDCG values of six recommenders increase since there are more training instances for user preference learning. Moreover, GCARec still obtains the best results. This indicates the robustness and superiority of GCARec under different interaction sparsity levels.

TABLE IV
ABLATION STUDY ON FOUR DATASETS *w.r.t.* NDCG@10

Dataset	BCE	BCE+G	BCE+T	BCE+C	GCARec
Gowalla	0.1208	0.1396	0.1408	0.1465	**0.1586**
Ciao	0.0315	0.0339	0.0335	0.0348	**0.0362**
Epinions	0.0102	0.0141	0.0133	0.0146	**0.0157**
Movie	0.0457	0.0513	0.0786	0.0972	**0.1115**

C. Ablation Study (RQ2)

To justify the rationality of our solution, we perform an ablation study in Table IV to evaluate the contribution of each part. To be specific, BCE stands for the MF model with binary cross-entropy loss. BCE+G replaces MF with a linear graph convolution module. On top of BCE, BCE+T only incorporates the time-sensitive attention mechanism to model item transition patterns, while BCE+C investigates the joint effect of dynamic sequential patterns and dynamic social influences by a co-attention module. Finally, we present GCARec to illustrate the necessity of unifying different modules. From Table IV, the main observations are as follows.

1) Without any modules, BCE achieves the worst performance. This shows that just modeling the direct user–item relationships is nontrivial to yield high-quality embeddings of users and items. Jointing analyzing Tables III and IV, BCE+G consistently surpasses BCE and LightGCN, showing the effectiveness of explicitly exploiting collaborative signals by the linear graph convolution module.
2) Compared with BCE, the results of BCE+T verify that capturing user's varying attentions on L-order MCs is beneficial for the recommendation performance. On top of BCE+T, BCE+C achieves further performance gains, demonstrating the importance of dynamically modeling social influence for inferring the next item.
3) As expected, GCARec achieves the best results, which shows that both modules are complementary for obtaining the satisfactory recommendation results. These results demonstrate the rationality of our solution.

D. Hyperparameter Sensitivity (RQ3)

In this section, we investigate the effect of two vital hyperparameters, i.e., the graph convolution layer number K that decides the representation capacity of long-term user preference and the Markov order L that plays a pivotal role in capturing sequential dynamics.

1) Effect of Layer Number K: By performing multiple graph convolution layers, GCARec could explicitly exploit topological structure information in the user–item interaction graph. As such, It is necessary to study the effect of different K values for our solution. Here, we choose $K \in \{1, 2, 3, 4, 5, 6\}$ to do experiments and illustrate the performance of GCARec in Fig. 6. We can observe the following.

1) With the increase of the layer number, GCARec obtains substantial improvements in all cases. For example, GCARec-2 significantly surpasses GCARec-1. We attribute these performance gains to the effective capturing of collaborative signals; second-order neighbor information in user–item graph is beneficial to refine node embedding learning, especially for inactive nodes.

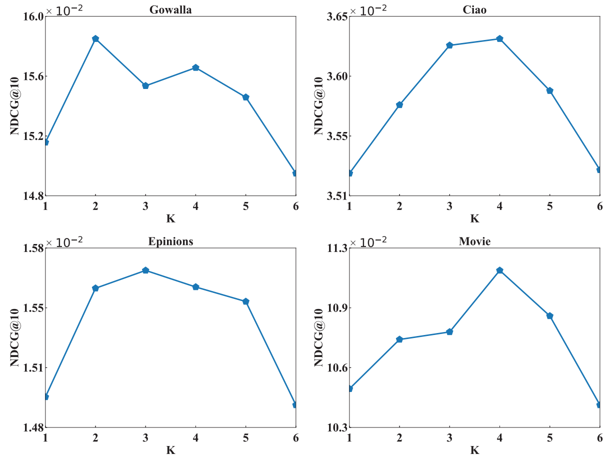


Fig. 6. NDCG@10 (y-axis) versus the layer number K (x-axis).

2) GCARec obtains the optimal results when K reaches a certain threshold. To be specific, when $K = 3$ on the Epinions dataset, our solution achieves the best results. When further stacking layers, the NDCG values downgrade quickly. This verifies that 3-order collaborative relations are sufficient for modeling long-term user preference on the Epinions dataset. Moreover, stacking more layers would introduce unnecessary neighbors for the target node in the embedding learning process, resulting in performance degradation. Other studies have also proved these findings [18], [22], [27], [40].

2) Effect of Markov Order L: In this section, we conduct GCARec with different Markov orders. The order of MCs determines the competence of capturing dynamic sequential patterns. In particular, we vary L from 1 to 7, while keeping the rest optimal hyperparameters fixed. In Fig. 7, we show the recommendation performance *w.r.t.* NDCG@10 and have the following observations.

1) Jointly analyzing Table III and Fig. 7, the GCARec with a first-order MC performs better than FPMC in all cases. This emphasizes the importance of integrating social factors into short-term user interest module for predicting the next action.
2) With the increase of L, the model performance gradually goes better. And when L goes beyond a certain threshold, the NDCG@10 values decrease with a further increase. This reveals that: a) a small order is sufficient for providing dynamic contextual information and b) a larger order may introduce some irrelevant items and even several negative impacts for capturing short-term dynamics.

E. Case Study (RQ5)

To clearly demonstrate how our solution works, several micro-level case studies are presented in Tables V and VI. We randomly sample two users (i.e., #328 and #796) from the douban Movie. For each user, they recently visited five items and had four friends in the social network. From the two tables, we have three main observations.

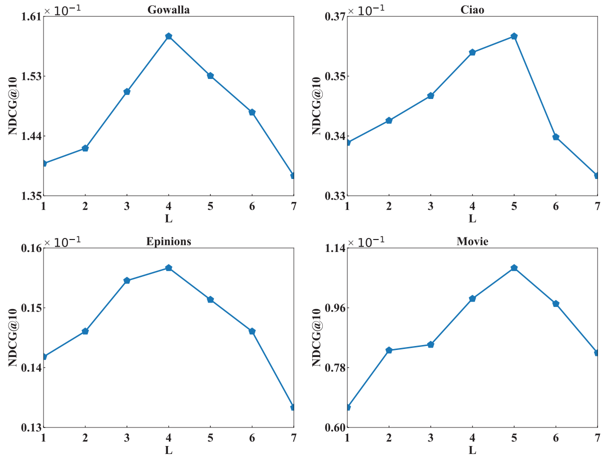


Fig. 7. NDCG@10 (y-axis) versus the Markov order L (x-axis).

TABLE V
CASE STUDIES OF THE TIME-SENSITIVE ATTENTION MECHANISM OF TWO SAMPLED USERS

UID \ Time	t-5	t-4	t-3	t-2	t-1
#796	#910	#505	#86	#1041	#3562
	0.052	0.135	0.112	0.205	**0.496**
#328	#125	#657	#86	#1041	#3328
	0.075	0.048	0.216	**0.372**	0.289

TABLE VI
CASE STUDIES OF THE SOCIAL-AWARE ATTENTION MECHANISM OF THE USER #328

Time \ FID	#128	#536	#1802	#2821
t_1	0.131	0.061	0.212	**0.596**
t_2	**0.488**	0.237	0.112	0.163

1) *For Different Target Users, the Attention Weights of the Same Items Vary Significantly:* For the same user, the more attentions appear in the end of his action sequence. This phenomenon justifies that the more relevant an item is to the user current taste, the closer its position is to the end of action sequence. Moreover, even though an item holds the same position in the L-order MCs (i.e., #86 and #1041), the contribution ratios show a marked difference between two users. To explain the rationality, we further analyze the timestamps of these actions. The user #328 watched these movies within two days, while the user #796 produced these actions in one month. These findings well validate our motivations (i.e., Fig. 1) in the introduction, presenting evidence that our solution is capable of uncovering meaningfully sequential patterns.
2) *For a Given User, the Influence Strengths of Different Friends Vary Significantly at a Specific Timestamp:* For instance, when predicting the user preference at time step t, GCARec assigns a higher attention weight on friend #2821 and a lower attention on friend #536. To analyze the reason, we investigate the action records of the target user #328 and his friend #2821. In recent days, such user is obsessed with the science fiction movies, and his friend #2821 also watched lots of movies on this topic like "Transformers" and "Avatar." In this case, it is reasonable to put a higher attention on friend #2821.
3) *For a Given User, the Influence Strengths of the Same Friend Vary Significantly at Different Time Steps:* For instance, when predicting the user preference at the timestamp t_1, the influence strength of friend #128 is considerably lower. At another timestamp, this case is very opposite. According to the douban Movie dataset, at the timestamp t_2, the user watched the romance movie "Before sunrise" a moment ago. In the observed interactions, friend #128 also watched a series of romance movies, including "Before sunset" and "Before sunrise." Therefore, the attention weight on his friend #128 is relatively higher. These findings well justify our argument in the introduction, presenting evidence that the social-aware attention mechanism could capture and infer the detailed variance of social influence.

V. CONCLUSION AND FUTURE WORK

In this article, we introduce a novel concept of socio-sequential recommendation, which highlights the importance of dynamically modeling social influences and adaptively capturing item transition patterns in a sequence. Toward this end, we develop an effective solution, named the graph-augmented co-attention model (GCARec), to explore the joint effect of dynamic social influences and dynamic sequential patterns. In particular, our solution includes two key modules: 1) a linear graph convolution module that models long-term user preference and 2) a co-attention module that differentiates the contribution ratio of each item in the L-order MCs and each friend for predicting the next interesting item. Remarkably, such two modules are easily implemented, plug-and-play, and could be integrated into different social or sequential recommendations. Extensive experiments have been made under a wide spectrum of recommendation scenarios. The empirical results demonstrated that GCARec significantly surpasses over several state-of-the-art methods. Furthermore, we performed qualitative studies of time-sensitive and social-aware attention mechanisms, which provide good interpretability about why the user favors a certain item at a specific timestamp.

This work represents an initial attempt for the task of socio-sequential recommendation and opens up new research possibilities. Despite the great success of GCRec, there are still many interesting questions to be explored by the research community. For instance, in order to keep the model's simpleness, we adopt a sampling-based learning strategy to optimize GCRec's model parameters, whereas several studies [18], [49] have proven that the model performance is highly sensitive to the design of the negative sampler. In this case, it is unclear whether there is a nonsampling learning strategy to optimize GCRec's model parameters. Besides, going beyond the merely graph structure data, how to leverage nonstructural knowledge for improving the GCRec's performance is worth studying.

In the future, we would like to explore more efficient and effective learner, which could optimize GCRec in a non-sampling manner and potentially achieve better performance than a sampling-based learning strategy. Moreover, we also would like to explore textual reviews [50], [51] and visual knowledge [52], [53] to build a more reliable recommendation.

REFERENCES

[1] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing Personalized Markov chains for next-basket recommendation," in *Proc. WWW*, 2010, pp. 811–820.

[2] R. He, W. Kang, and J. McAuley, "Translation-based recommendation," in *Proc. RecSys*, 2017, pp. 161–169.

[3] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *Proc. ICLR*, 2016, pp. 1–10.

[4] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. CIKM*, 2017, pp. 1419–1428.

[5] J. Tang and K. Wang, "Personalized top-N sequential recommendation via convolutional sequence embedding," in *Proc. WSDM*, 2018, pp. 565–573.

[6] A. Yan, S. Cheng, W. Kang, M. Wan, and J. McAuley, "CosRec: 2D convolutional neural networks for sequential recommendation," in *Proc. CIKM*, 2019, pp. 2173–2176.

[7] W. Kang and J. J. McAuley, "Self-attentive sequential recommendation," in *Proc. ICDM*, 2018, pp. 197–206.

[8] J. Li, Y. Wang, and J. McAuley, "Time interval aware self-attention for sequential recommendation," in *Proc. WSDM*, 2020, pp. 322–330.

[9] P. V. Marsden and N. E. Friedkin, "Network studies of social influence," *Sociol. Methods Res.*, vol. 22, no. 1, pp. 127–151, 1993.

[10] L. S.-L. M. McPherson and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, 2001.

[11] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. RecSys*, 2010, pp. 135–142.

[12] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. WSDM*, 2011, pp. 287–296.

[13] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix factorization," in *Proc. CIKM*, 2008, pp. 931–940.

[14] B. Yang, Y. Lei, J. Liu, and W. Li, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.

[15] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, "Collaborative neural social recommendation," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 464–476, Jan. 2021.

[16] X. Wang, X. He, L. Nie, and T. Chua, "Item silk road: Recommending items from information domains to social users," in *Proc. SIGIR*, 2017, pp. 185–194.

[17] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *Proc. SIGIR*, 2019, pp. 235–244.

[18] B. Wu, L. Zhong, L. Yao, and Y. Ye, "EAGCN: An efficient adaptive graph convolutional network for item recommendation in social Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16386–16401, Sep. 2022.

[19] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. WSDM*, 2019, pp. 555–563.

[20] P. Sun, L. Wu, and M. Wang, "Attentive recurrent social recommendation," in *Proc. SIGIR*, 2018, pp. 185–194.

[21] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. WWW*, 2017, pp. 173–182.

[22] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. SIGIR*, 2020, pp. 639–648.

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

[24] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. ICDM*, 2008, pp. 263–272.

[25] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," in *Proc. IJCAI*, 2018, pp. 2227–2233.

[26] S. Zhang, L. Yao, B. Wu, X. Xu, X. Zhang, and L. Zhu, "Unraveling metric vector spaces with factorization for recommendation," *IEEE Trans. Ind. Informat.*, vol. 16, no. 2, pp. 732–742, Feb. 2020.

[27] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, "Neural graph collaborative filtering," in *Proc. SIGIR*, 2019, pp. 165–174.

[28] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel recommendation model regularized with user trust and item ratings," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1607–1620, Jul. 2016.

[29] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. CIKM*, 2014, pp. 261–270.

[30] J. Chen, C. Wang, S. Zhou, Q. Shi, Y. Feng, and C. Chen, "SamWalker: Social recommendation with informative sampling strategy," in *Proc. WWW*, 2019, pp. 228–239.

[31] Y. Liu, C. Liang, X. He, J. Peng, Z. Zheng, and J. Tang, "Modelling high-order social relations for item recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 9, pp. 4385–4397, Sep. 2022.

[32] H. Fang, G. Guo, D. Zhang, and Y. Shu, "Deep learning-based sequential recommender systems: Concepts, algorithms, and evaluations," in *Proc. ICWE*, 2019, pp. 574–577.

[33] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. A. Orgun, "Sequential recommender systems: Challenges, progress and prospects," in *Proc. IJCAI*, 2019, pp. 6332–6338.

[34] B. Wu, X. He, Z. Sun, L. Chen, and Y. Ye, "ATM: An attentive translation model for next-item recommendation," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1448–1459, Mar. 2020.

[35] L. Yu, C. Zhang, S. Liang, and X. Zhang, "Multi-order attentive ranking model for sequential recommendation," in *Proc. AAAI*, 2019, pp. 5709–5716.

[36] Z. Sun, B. Wu, Y. Chen, and Y. Ye, "Learning from the future: Light cone modeling for sequential recommendation," *IEEE Trans. Cybern.*, early access, Nov. 23, 2022, doi: 10.1109/TCYB.2022.3222259.

[37] C. Ma, P. Kang, and X. Liu, "Hierarchical gating networks for sequential recommendation," in *Proc. KDD*, 2019, pp. 825–833.

[38] W. Duan, J. Xuan, M. Qiao, and J. Lu, "Learning from the dark: Boosting graph convolutional neural networks with diverse negative samples," in *Proc. AAAI*, 2022, pp. 6550–6558.

[39] B. Wu, L. Zhong, H. Li, and Y. Ye, "Efficient complementary graph convolutional network without negative sampling for item recommendation," *Knowl.-Based Syst.*, vol. 256, Nov. 2022, Art. no. 109758.

[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017, pp. 1–14.

[41] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2018, pp. 1–12.

[42] F. Wu, A. H. de Souza, T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proc. ICML*, 2019, pp. 6861–6871.

[43] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. AAAI*, 2019, pp. 346–353.

[44] Y. Liu, T. Pham, G. Cong, and Q. Yuan, "An experimental evaluation of point-of-interest recommendation in location-based social networks," *Proc. VLDB Endowment*, vol. 10, no. 10, pp. 1010–1021, 2017.

[45] J. Tang, H. Gao, H. Liu, and A. D. Sarma, "eTrust: Understanding trust evolution in an online world," in *Proc. SIGKDD*, 2012, pp. 253–261.

[46] B. Wu, X. He, Q. Zhang, M. Wang, and Y. Ye, "GCRec: Graph-augmented capsule network for next-item recommendation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 25, 2022, doi: 10.1109/TNNLS.2022.3164982.

[47] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. UAI*, 2009, pp. 452–461.

[48] C. Chen et al., "An efficient adaptive transfer neural network for social-aware recommendation," in *Proc. SIGIR*, 2019, pp. 225–234.

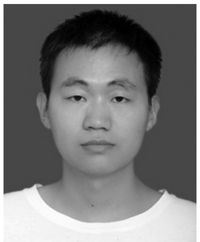
[49] C. Chen, W. Ma, M. Zhang, C. Wang, Y. Liu, and S. Ma, "Revisiting negative sampling vs. non-sampling in implicit recommendation," *ACM Trans. Inf. Syst.*, to be published.

[50] Z. Cheng, Y. Ding, X. He, L. Zhu, X. Song, and M. S. Kankanhalli, "A^3NCF: An adaptive aspect attention model for rating prediction," in *Proc. IJCAI*, 2018, pp. 3748–3754.

[51] C. Ma, P. Kang, B. Wu, Q. Wang, and X. Liu, "Gated attentive-autoencoder for content-aware recommendation," in *Proc. WSDM*, 2019, pp. 519–527.

[52] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, and Z. Qin, "Aesthetic-based clothing recommendation," in *Proc. WWW*, 2018, pp. 649–658.

[53] B. Wu, X. He, Y. Chen, L. Nie, K. Zheng, and Y. Ye, "Modeling product's visual and functional characteristics for recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 3, pp. 1330–1343, Mar. 2022.



Bin Wu received the B.S. degree in computer science and technology, the M.S. degree in computer software and theory, and the Ph.D. degree in software engineering from Zhengzhou University, Zhengzhou, China, in 2013, 2016, and 2020, respectively.

He is currently a Research Associate Professor with the School of Computer and Artificial Intelligence, Zhengzhou University. He has over 20 publications appeared in referred journals and conferences, including IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, International Conference on Web Search and Data Mining, *Information Sciences*, and IEEE INTERNET OF THINGS JOURNAL. His current research interests include graph representation learning, recommender systems, social network analysis, and multimedia.

Xiangnan He (Senior Member, IEEE) received the Ph.D. degree in computer science from the National University of Singapore (NUS), Singapore, in 2016.

He was a Postdoctoral Research Fellow with NUS, until 2018. He is currently a Professor with the School of Data Science, University of Science and Technology of China, Hefei, China. He has over 100 publications appeared in several top conferences, including ACM SIGIR Conference on Research and Development in Information Retrieval, ACM SIGKDD Conference on Knowledge Discovery and Data Mining, and journals, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and IEEE TRANSACTIONS ON MULTIMEDIA. His current research interests include information retrieval, data mining, and multimedia analytics.

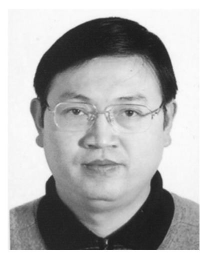
Le Wu (Member, IEEE) received the Ph.D. degree in computer application technology from the University of Science and Technology of China, Hefei, China, in 2015.

She is currently a Professor with the Hefei University of Technology, Hefei. She has published more than 50 papers in referred journals and conferences. Her general area of research interests is data mining, recommender systems, and social network analysis.

Dr. Wu is the recipient of the Best of SDM 2015 Award, the Distinguished Dissertation Award from the China Association for Artificial Intelligence 2017, and the Youth Talent Promotion Project from the China Association for Science and Technology.

Xue Zhang received the B.S. degree in electronics information science and technology from Henan University, Kaifeng, China, in 2012. She is currently pursuing the master's degree in software engineering with the School of Information Engineering, Zhengzhou University, Zhengzhou, China.

Her current research interests include recommender systems, graph learning, and information retrieval.



Yangdong Ye received the Ph.D. degree in traffic information engineering and control from the China Academy of Railway Sciences, Beijing, China, in 2002.

From 2003 to 2004, he was a Senior Visiting Scholar with Deakin University, Geelong, VIC, Australia. He is currently a Professor with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. He has over 60 publications in peer-reviewed prestigious journals and conferences, such as IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTION ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON CYBERNETICS, *ACM Transactions on Knowledge Discovery From Data*, *Neural Networks*, *Information Fusion*, *Information Sciences*, *Pattern Recognition*, IEEE Conference on Computer Vision and Pattern Recognition, International Joint Conference on Artificial Intelligence, and ACM International Conference on Multimedia. He has wide research interests, including machine learning, pattern recognition, knowledge engineering, and intelligent system.