MGCN4REC: Multi-graph Convolutional Network for Next Basket Recommendation with Instant Interest

Yan Zhang, Bin Guo(✉), Qianru Wang, Yueqi Sun, and Zhiwen Yu

Northwestern Polytechnical University, Xi'an 710129, China
guob@nwpu.edu.cn

Abstract. Sequential patterns involved in users' historical behaviors have received extensive attention in recommendation system, which is important to represent item-level preferences. The existing works often combine the long- and short-term patterns to capture user's preferences. But the short-term preferences modeled by the recent behavior patterns cannot clearly indicate the users' instant interest. In this paper, we propose a sequential recommendation model MGCN4REC based on multi-graph to learn the representation of users and items and then model preferences and instant interests simultaneously. Firstly, this paper utilizes multi-graph convolutional network (MGCN) to learn users and items embeddings from multi-graph. Secondly, to aggregate preferences and instant interests, we use the attention mechanism to find the degrees of dependencies on these two features. Finally, this paper conducts experiments on real data sets of Amazon to evaluate the performance of MGCN4REC model, and the results show that our model outperforms the current state-of-the-art sequential recommendation methods over 15% on the metrics.

Keywords: Sequential recommendation · Multi-graph convolution network · Instant interest · Attention mechanism

1 Introduction

Recommendation systems strive to recommend items that users are willing to purchase or interact with in modern e-commerce applications, however, the large number of user interactions (such as browsing, clicking, collecting, shopping carts, purchasing) in e-commerce makes it a great challenge to identify the users' consumption patterns and interest preferences.

Existing recommendation systems model the user-item interactions mainly in two ways. The first one is traditional recommendation systems that focus on mining static associations from user-item interactions, which are represented by collaborative filtering models [3–5]. The second one is sequential recommendation systems [6, 7] that try to capture the patterns hide in the successive user's interaction behaviors, and the user's long-term preferences and short-term preferences can be dynamically represented.

Z. Yu et al. (Eds.): GPC 2020, LNCS 12398, pp. 171–185, 2020.
https://doi.org/10.1007/978-3-030-64243-3_14

Specifically, the user's long-term preferences reflect in the historical behaviors and the short-term preferences are determined by recent purchases items.

Although the existing sequential recommendation models have achieved promising results, two shortcomings are remained. Firstly, these methods directly use the item sequence to represent the relationship between items. However, they fail to take into account that different users pay attention to different aspects. Secondly, most of the existing models neglect users' instant interests. Specifically, instant interests refer to an instant and specific purchase demand and are more helpful than the short-term preferences. For example, we can learn from the Q&A function in Amazon's earphone information page, as shown in Fig. 1, user's search contents ("airplane use", "workouts") reflects the attention to different aspects when having the same instant purchase needs ("earphone"), which can't be captured in their historical behaviors and more accurate than short-term preferences.

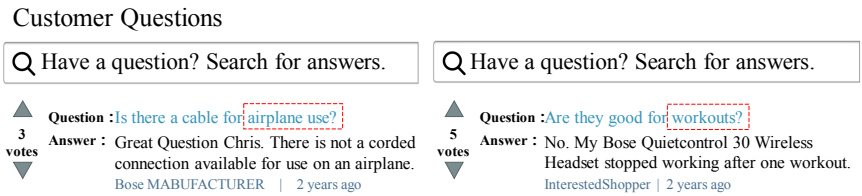

Fig. 1. Q&A in an item page of Amazon. User1 and user2 have different instant requirements when buying an earphone.

To address the above challenges, we propose a sequential recommendation model MGCN4REC. Firstly, inspired by the works in user and item representation learning [8–11], we model the users' historical interaction sequence as a multi-graph (user relationship graph, item relationship graph, and user item bipartite graph), and learn a more feasible representation of each user and item through graph convolution network. Secondly, we use a recurrent neural network to model users' preferences from historical interaction sequence and instant interests from users' question-and-answer data. Finally, we use attention mechanism to aggregate users' preferences and instant interests and make the recommendation.

In summary, the main contributions of this paper are as follows:

- This paper proposes a sequential recommendation model (MGCN4REC) based on preferences and instant interests. We leverage instant interests to better represent short-term preferences and uses the attention mechanism to aggregate preferences and instant interests.
- This paper firstly models the user-item interaction sequence as a multi-graph structure, and leverages a graph convolution network to learn the embedding representation of users and items in the same feature space.
- To evaluate our model, experiments are conducted on Amazon's real dataset and the results show that MGCN4REC outperforms than the state-of-art baselines over 15%.

2 Related Work

Among many Internet-based services [12] such as e-commerce, the recommendation system plays a vital role, and it has also attracted widespread attention in industry and academia. The related work in this paper mainly includes two aspects of representation learning and sequential recommendation system.

2.1 Representation Learning

For user and item representation learning, the recommendation system evolves according to the embedding between different entities in addition to three models: user-based recommendation based on user similarity, item-based recommendation based on user similarity of item interactions, and user extraction Interact with the characteristics of items and recommend model-based items with these characteristics. All three models inevitably need to embedding users and items to obtain the user-user, item-item, and user-item similarity. The user-based usage scenario is that the number of users is much larger than the number of items and few items appear, mainly focusing on the calculation of user-user similarity. On the contrary, the item-based usage scenario is that the number of items is much larger than the number of users and few new users appear, focusing on calculating items-items similarity. Model-based mainly focuses on calculating the correlation between users and items.

Barkan *et al*. [9] consider the user's historical interaction sequence as a natural sentence and they propose the item2vec, which maximizes the probability of the appearance of other items in the window under each central item to learn the vector representation of each item. Perozzi *et al*. [10] converts the historical interaction sequence of the user into a graph and they propose deepwalk, which utilize deep first search (DFS) on each node to obtain multiple item sequences and learn the vector representation of each item through item2vec. Different from DeepWalk, Grover *et al*. propose an item sequence sampling method by jointly utilizing broad first search (BFS) and deep first search (DFS), called Node2Vec, which allocates sampling probabilities according to the distance between items and the source item. After getting the sequence, they use Item2Vec to learn the representation of each item.

Most of the existing works focus on learning the similarity between items or learning the low-dimensional embedding of users and items separately. One challenge exists when directly combining the embedding of the user and his item sequence since they are in different vector space, making it impossible to directly use their inner product to get a recommendation list. Wu *et al*. [13] propose SR-GNN which constructs the user's interaction sequence of each session into a directed graph, and then utilizes a gated neural network to learn the representation of each node and users' current session. Grbovic *et al*. [14] propose list embedding which personalizes recommendations by leveraging Word2vec to model users and items into a sequence. However, due to the specificity of Airbnb data, the method is not robust to our work.

Therefore, this work intends to learn the embedding representation of each node from the user relationship graph, item relationship graph, and user-item interaction bipartite graph through graph convolutional neural network, so as to obtain the similarity between users and items.

2.2 Sequential Recommendation System

In recent years, researchers have begun to focus on various sequential recommendation scenarios, such as the next-basket recommendation [15, 16], session-based recommendation [17–19], and next item recommendation [7, 20].

The existing works of sequential recommender systems are mainly based on sequential pattern mining. Due to the huge success of deep neural networks in the past few years, sequential data modeling methods have made great progress in several applications such as NLP, social media, and recommender systems. For example, Wang *et al*. [16] make the next-time recommendation by hierarchical interactions, which captures the sequential patterns and the user's overall preferences through historical interaction sequences. Hidasi *et al*. [17] utilize recurrent neural networks (RNNs) to model the entire session and then build a session-based recommender system, which have significantly better performance than item-based methods. Tang *et al*. [19] utilize the horizontal convolution kernel and vertical convolution kernel in the convolutional neural network (CNN) to capture the user's long-term interest and short-term interest and make recommendations respectively. Li *et al*. [20] utilize the w-item2vec method to learn the vector of items, and then expresses the user's long-term and short-term preferences and makes recommendations. Although these models consider the user's sequence information, there are still shortcomings in terms of representing user long-term preferences and short-term preferences. In contrast, the method proposed in this paper combines the user's preferences with the instant interests to make recommendations. The preferences and instant interests are obtained from the user's historical interaction sequence with the item and the user's current question text.

In summary, the main work of this paper is to make a unified representation of users and items, and calculate their correlations. Based on this vector, model users' preferences and instant interests to perform sequential recommendations.

3 MGCN4REC Framework

3.1 Preliminaries

Let $U = \{u_1, u_2, \ldots, u_n\}$ denotes the set of users, $I = \{i_1, i_2, \ldots, i_m\}$ denotes the set of items, where n and m denote the number of users and items respectively. In addition, $E = \left\{e_1, e_2, \ldots, e_{(m+n)}\right\}$ denotes the embedding vector of all users and items, where e_u is the embedding vector of a user ($e_u \in R^d, u \in U$), and e_i is the embedding of the item i_i ($e_i \in R^d, i \in I$). For each user $u \in U$, we assume that his/her enquiry time is t_q. Its historical baskets before the question time t_q are represented as:

$$B^u_{<t_q} = \{\left[e^u, b^u_{t_1}\right], \left[e^u, b^u_{t_2}\right], \ldots, \left[e^u, b^u_{t_{q-1}}\right] | b_{t_j} \subset I\}$$

For each basket, there are usually more than one item in it ($|b_{t_j}|\rangle 1$), Therefore, we utilize the mean pooling operation to present the embedding of the shopping basket at each timestep:

$$b_{t_j} = \frac{1}{\left|b_{t_j}\right|} \sum_{k=1}^{\left|b_{t_j}\right|} e_{i_k},\ e_{i_k} \in E$$

where $\left|b_{t_j}\right|$ represents the number of items in the shopping basket at the timestep t_j, and e_{i_k} represents the embedding representation of the item i_k in the user's shopping basket. Similarly, the basket after t_q is donated as $b^u_{t_{q+1}}$.

To recommend items with preferences and instant interests, we formalize the problem as the following objective function:

$$L = \sum_u L_u(f(B^u_{<t_q}, s^u), b^u_{t_{q+1}}) \tag{1}$$

Formula (1) is used to indicate the difference between the recommended item and the actual purchased item at the next time t_{q+1}: where $f(\cdot)$ represents a function that predicts the probability of being recommended for other items from an aggregated preferences of traditional preferences and instant interests; $L_u(\cdot)$ represents the loss between the recommended items and the actual purchased items in this method. However, users may not immediately purchase the items they are interested in after asking questions: from the conclusion of Caser [19], it can be known that there is a skipping behavior in the user's interaction (the previous several sequence behaviors do not directly affect the purchase behavior at t + 1, but affect the behavior at t + 2 or even t + 3). Based on this, the aggregated preferences of traditional preferences and instant interests may affect the purchasing behavior at multiple moments after t_{q+1}. Therefore, we redefine the objective function as formula (2), where $B^u_{t>t_q} = U_{t>t_q} b^u_t$ represents a group of items to be purchased after time t_q.

$$L = \sum_u L_u(f(B^u_{<t_q}, s^u), B^u_{>t_{q+1}}) \tag{2}$$

In this section, we formally define next-basket recommendation problem, and present the MGCN4REC system in detail, as shown in Fig. 2. MGCN4REC is mainly composed of three modules: multi-graph-based joint embedding of users and items, preferences and instant interests modeling, attention-based preferences aggregating. Firstly MGCN4REC converts the user's historical interaction sequence into a graph, including a user relationship graph taking users as the nodes and the number of items that two nodes interact as the edge weights, an item relationship graph taking items as the nodes and the number of users that two nodes are purchased as the edge weights, the user-item bipartite graph taking users and items as the nodes and the interaction between user and item as edges. Secondly, the graph convolutional network(GCN) is used to learn the representation of each node from three types of graphs, which can obtain the similarity between the users and items. Afterwards, the preferences of users are obtained through a Bi-RNN, and the users' instant interests are analyzed from the user's question text at the current moment. Finally, the user's preferences and instant interests are aggregated based on the attention mechanism for accurate recommendations.

3.2 Multi-graph-Based User-Item Representation

At the first stage of GCN-Rec, the purpose of multi-graph-based user item vector representation learning is to generate a unified representation for each user and project by learning the similarity between them from a large number of user historical interactions.

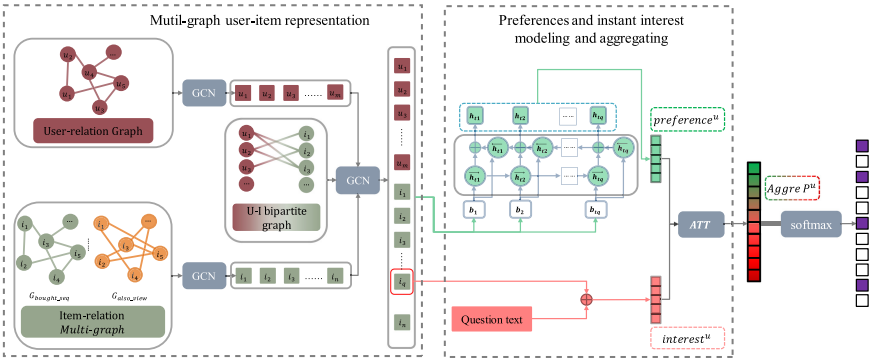

Fig. 2. Architecture of the MGCN4REC system.

In pervious works, sequential recommendations have always used one-hot coding or added an additional embedding layer to the deep learning architecture for items representation [17, 22]. However, for item sets on large e-commerce platforms, on the one hand, the one-hot coding network may cost expensive time, and it is often not well optimized due to the high sparsity [23]. On the other hand, the additional embedding layer will cause a certain loss of neural networks [17]. The item2vec mines similarities between items from user's historical interaction sequences, but it ignores the user interaction intensity for different items, which cannot effectively represent the item features with less interaction and user features. However, the graph neural networks (GNN) can learn from different types of graphs to the rich structural information and the relationships between different nodes, so MGCN4REC uses graph convolutional networks to abstract the representations of users and items.

Different types of heterogeneous graphs can present the characteristics of different dimensions of nodes. For example, we can learn the similarity between items from the shopping relationship graph, which represents those items may be purchased together. Furthermore, we could learn the competition among items from the attention item graph, meaning which item is selected from multiple items. Therefore, this paper uses the multi-graph to represent the relationship between items.

For the representation of items, we first get the sequence of the user's interaction history, $s_u = \left\{i_1^u, i_2^u, i_3^u, \ldots, i_n^u\right\}$(u denotes the index of the user), and the purchase relationship graph between items is represented from the historical interaction sequence, $G_{bought\_seq} = \left\{I, e_{bought\_seq}\right\}$(I denotes the item set and e_{bought_seq} denotes the edge set, $e_{seq} = \left\{\left(i_1^1, i_2^1\right), \left(i_2^1, i_3^1\right), \ldots \left(i_{(n-1)}^{|U|}, i_n^{|U|}\right)\right\}$, $\left(i_{(n-1)}^{|U|}, i_n^{|U|}\right)$ denotes that $i_{(n-1)}^{|U|}$ has the link with $i_n^{|U|}$). Second, we create a graph of the relationships between simultaneous purchases $G_{also-bought} = \left\{I, e_{also-bought}\right\}$, where $e_{also-bought}$ indicates that there is an edge between other purchased items at the same time and this item. Third, we create a graph between purchased items and simultaneous browsing items $G_{also-view} = \{I, e_{also-view}\}$, where $e_{also-view}$ indicates an edge between the other items browsed and the purchased item. Then calculate the adjacency matrix $A^{|I|*|I|}$ and degree matrix $D^{|I|*|I|}$ of each graph, using the graph convolutional neural network for each graph.

Specifically, the graph convolutional network aggregates the feature information of its neighbor nodes to update its own features for each node in the graph, and then nonlinearly transform the feature information to gain robust feature representations. The feature update function is as follows:

$$h_{v_i}^{l+1} = \sigma\left(\sum_j \frac{1}{c_{ij}} h_{v_j}^l W^l\right) \tag{3}$$

where j indicates the neighboring nodes of v_i, and c_{ij} is a normalization constant for the edge (v_i, v_j), originating from the symmetrically normalized adjacency matrix $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ in MGCN4REC.

Convert the update function to compute the nodes in the entire graph, so every neural network layer can then be described as a non-linear function:

$$H^{l+1} = f(H^l, A) \tag{4}$$

with $H^0 = X$ and $H^L = Z$. L is the number of layers, and X is the input features of nodes. In this paper, we use all aspects extracted from all reviews of each item to represent its initial features, and utilize each user's sentiment scores on all aspects to represent the user's initial feature. The specific propagation rule in convolutional layers $f(\cdot)$ can be written as:

$$f(H^l, A) = \sigma\left(\widehat{D}^{-\frac{1}{2}}\widehat{A}\widehat{D}^{-\frac{1}{2}}H^l W^l\right) \tag{5}$$

where W^l is a weight matrix for the neural network layer $layer_l$ and $\sigma(\cdot)$ is a non-linear activation function such as ReLU. $\hat{A} = A + E$, where E is the identity matrix and $\hat{D}$ is the diagonal degree matrix of $\hat{A}$, i.e. $\hat{D} = D + E$.

Since multi-graph can learn the features of nodes from different perspectives, the convolutional neural network is used to the graph G_{bought_seq} of the relationship between items purchased sequences, the graph $G_{also-bought}$ of the relationships between simultaneous purchases and the graph $G_{also-view}$ between purchased items and simultaneous browsing items to obtain the representation of each item:

$$Z_{item} = \left[Z_{seq}, Z_{also-bought}, Z_{also-view}\right]$$

For the representation of the user, repeat formulas (3)–(5), and will get the embedding for each user representing Z_{user}.

Take Z_{user} and Z_{item} as X, use the graph convolutional network for the bipartite graph of users and items interaction, and repeat formula (3)-(5) to obtain the embedding representation of all users and items:

$$Z = \left\{z_{u_1}, z_{u_2}, \ldots, z_{u_m}, z_{i_1}, z_{i_2}, .., z_{i_n}\right\}$$

3.3 User Preferences and Interests Modeling

The user's decision-making process is mainly affected by two factors: preferences and instant interests. In fact, the user's interaction process is a series of implicit feedback over time. Therefore, unlike traditional recommendation systems that explore user-item interaction from a static way, this paper uses serialized modeling to process the next basket recommendation problem. Specifically, the system designs two models of preferences and instant interests learning to distinguish the user's preferences from the instant interests, and then makes personalized recommendations by combining the user's preferences and instant interests.

Preferences Modeling
As the preferences of the user runs through the entire sequence of the user, they depend not only on the previous purchase sequence but also on the future purchase sequence. Inspired by the bidirectional RNN [24], this paper applies CLSTM to a bidirectional architecture Bi-CLSTM, making full use of the interaction sequence of the forward and backward directions to express the preferences of the user. Therefore, the system developed a deep neural network based on GRU [25] to model the stable preferences of users. After initialization, the hidden state h_j of each interaction is updated from the previous hidden state h_{j-1} at the jth time of the interaction:

$$\begin{aligned} i_j &= \delta(W_{vi}b_j + W_{hi}h_{j-1} + W_{ci}c_{j-1} + \widehat{b_i}), \\ f_j &= \delta(W_{vf}b_j + W_{hf}h_{j-1} + W_{cf}c_{j-1} + \widehat{b_f}), \\ c_j &= f_j c_{j-1} + i_j \tanh(W_{vc}b_j + W_{hc}h_{j-1} + \widehat{b_c}), \\ o_j &= \delta(W_{vo}b_j + W_{ho}h_{j-1} + W_{co}c_j + \widehat{b_o}), \\ h_j &= o_j \tanh(c_j) \end{aligned} \tag{6}$$

where i_j, f_j, and o_j correspond to the input gate, forget gate, and output gate of the GRU, b_j is the vector representation of the basket at the current time, c_j is the value of the GRU memory unit, $\hat{b}$ is the bias term, and h_j is the hidden status of timestep t_j.

At each time t_q, calculate the hidden state $\overrightarrow{h_{t_q}}$ of the forward RNN based on the previous hidden state $\overrightarrow{h_{t_q-1}}$ and the corresponding basket representation b_{t_q} of the current time; calculate the hidden state $\overleftarrow{h_{t_q}}$ of the forward RNN based on the back hidden state $\overleftarrow{h_{t_q+1}}$ and the corresponding basket representation b_{t_q}; therefore, the hidden state h_{t_q} of the bidirectional RNN at each moment t_q can be expressed by the hidden state of the forward RNN and the hidden state of the backward RNN:

$$h_{t_q} = concatenate(\overrightarrow{h_{t_q}}, \overleftarrow{h_{t_q}}) \tag{7}$$

Represent the preferences of user u through the average pooling layer as

$$preference^u = average(h_1, h_2, \ldots, h_{t_q}) \tag{8}$$

Instant Interests Modeling

As shown in the user question data of Fig. 1, the user's question can clearly reflect the user's instant interests and the characteristics most interested in the item in the recent period.

Instant interests are represented by the embedding vector of the item being questioned at the time t_q and the vector of the question text:

$$interest^u = [e_{t_q} | Score_{a_1}, Score_{a_2}, \ldots, Score_{a_m}], \quad e_{t_q} \in \mathrm{E} \tag{9}$$

where $Score_{a_i}$ is the emotional score of the $aspect_i$ in the user's question text.After obtaining the preferences and instant interests of user u at the question time t_q, by combining the stable preferences and dynamic instant interests, the aggregated preferences $P^u_{t_q}$ of user u at the question time t_q can be obtained, and then through the fully connected layer to make personalized recommendations.

3.4 Attention-Based Preferences Aggregating

If simply using the user aggregation preferences $P^u_{t_q}$ obtained in Sect. 3.3 for recommendation, it means that the degree of dependence on each user's preferences and instant interests is same. But in fact, the impact of preferences and instant interests on the final decision-making behavior of users is essentially different. Among them, instant interests determine the items that users want to buy, and preferences determine the details of item. And may be different for different users depend on preferences and instant interests, user A may be more in line with historical habits when shopping, while user B may prefer novel items. Therefore, the system introduces an attention mechanism [2] to aggregate the user's preferences and instant interests. The aggregated preferences after the introduction of the attention mechanism is expressed as follows:

$$Aggre\, P^u_{t_q} = \beta [preference^u, interest^u]^T \tag{10}$$

$$\beta^u_i = \frac{exp(Aggre\, p^u_i)}{\sum_{j=1}^{||preference|+|interest||} exp(Aggre\, p^u_j)} \tag{11}$$

Finally, the system uses a fully connected layer to find the relationship between the aggregation preferences $P^u_{t_q}$ and the target term, as shown in formula (12). And $\hat{y}^u$ represents the probability of user u interacting with the item after asking a question:

$$\hat{y}^u = sigmoid(W(Aggre\, p^u)^T + b) \tag{12}$$

3.5 Model Training

The Loss function of this system includes two parts: the first is the difference between the recommendation made by aggregated preferences and the actual purchase of items as shown in formula (2); the other part introduces an auxiliary loss function that is he error between the predicted preferences and the real preferences at each t + 1 moment in

preferences modeling. The user's preferences modeling is introduced in the Sect. 3.3.1. The GRU layer can obtain the set of items that may be purchased at time t + 1. The system reduces the error between the predicted set and the real set at each time, so that the recommendation can obtain better accuracy. The set of items that may be purchased at t + 1 can be expressed as:

$$\hat{b}_{t+1}^{u} = f_R(b_t^u, h_t) \tag{13}$$

where $f_R(\cdot)$ is a function based on the RNN structure, $h_t \in R^d$ is the hidden state at time t, and it is a dynamic representation of user preferences. In this article, $f_R(\cdot)$ selects GRU. At the same time, we introduce the error between the predicted value and the true value at each time:

$$L_R = \sum_{t=1}^{t_q-1} L_t(\hat{b}_t^u, b_t^u) \tag{14}$$

where $L_R(\cdot)$ is a loss function that measures the error between predicted preferences and true preferences. A cross-entropy loss function is used in this system, as shown in formula (15):

$$L = \sum_{u} (\gamma L_u(\hat{y}^u, B_{t>t_q}^u) + (1-\gamma)L_R) \tag{15}$$

where $\gamma \in (0, 1)$ is a parameter that balances the relative importance of the two-part loss function. Due to the large number of items, if all negative samples are used to reduce the Loss function, the time complexity and space complexity of training will be too high. Therefore, in this paper, the system uses the negative sampling technique to train the $L_U(\cdot)$ function.

4 Experiment

4.1 Dataset

We collect amazon reviews and q&a records in [26]. Users are used to comment on items they buy and subsequent users can take comments as an advice. We first obtain the comment records to obtain the initial characteristics of the user and the item, and then leverage the user's history to construct the commodity graph and the user graph. To extract features from item review data, we first used Stanford CoreNLP (A kind of NLP's analysis tool) to extract feature words and emotional scores (on a scale of 1 to 5), and then we use TF-IDF technique to select the feature with the highest frequency as the feature of the current item.

We conduct experiments with two sub-datasets of the amazon dataset, Electronics and Baby, in [26]. Since our model is mainly aimed at recommending to users after asking questions, we filter out users who have never asked a question. The statistics of the two datasets are shown in Table 1.

Table 1. Statistics of dataset

	Number of User	Number of items	Number of users with purchase after question	Number of users who purchase question-related
Electronics	196,114	65,124	19,445	4,881
Baby	17,217	4,982	1,321	213

4.2 Baselines

As the method in this paper firstly includes the representation learning of users as well as items and the sequential recommendation module, we compare our model with two user-item representation models and two sequential recommendation models:

- Item2vec [9], which represents item vector based on Word2Vec and we use it to replace the GCN module in this article as the baseline;
- node2vec [11], which represents item vector based on GNN. Based on depth-first search and breadth-first search, the model constructs sequences by exploring nodes that are similar to the structure and essence of the target node, so as to obtain the representation of each node and we use it instead of the GCN module as the baseline in this paper;
- DREAM [18], which is a sequential recommendation model based on CNN while instant interest is not considered. We take the historical interaction as the input of LSTM for personalized recommendation;
- Caser [19] ,which is a sequential recommendation model based on CNN and embeddings of both item and user are considered. In order to adapt to the problems in this paper, user embedding represents users' stable long-term preferences. Caser's vertical convolution kernel and horizontal convolution kernel are used to extract users' short-term preferences, which is different from the immediate interest of this paper;
- BINN [20], which is a sequential recommendation model based on RNN and contains the item2vec module to learn the item representation. Bidirectional RNN is used to model the user's long-term preferences, and the recent session is used to model the user's short-term preferences, which is different from the instant interest of this paper.

4.3 Evaluation Metric

To evaluate the performance of the MGCN4REC model in this article, we use Recall @K and HT@K, which are widely used in recommendation systems. Given the top-K recommendation results R_K^u of the user u, the calculation formula is shown in (16):

$$\begin{aligned} Recall@K &= \frac{1}{|U|}\sum_{u}\frac{|R_K^u \cap B_{t>t_q}^u|}{|B_{t>t_q}^u|}, \\ HR@K &= \frac{I(R_K^u \cap B_{t>t_q}^u \neq 0)}{|U|} \end{aligned} \tag{16}$$

where $\left|B^{u}_{t>t_q}\right|$ is the number of purchases after user asks, and|U| is the number of users.

4.4 Experimental Results

Performance Comparison

The results of the MGCN4REC model and comparative algorithms are as shown in Table 2, in which the first column represents the two data sets, the second column shows performances on different evaluation metrics: each index of the optimal data is in bold and suboptimal results are shown underlined. The last column represents the improvement of MGCN4REC model compared with the suboptimal results.

Table 2. Performance comparisons of MGCN4REC with baseline methods on two datasets (Bold scores are the best in each row, while underlined scores are the second best).

Datasets	Metric	Item2vec	Node2vec	DREAM	Caser	BINN	MGCN4REC	Improve
Baby	Recall@5	0.0041	0.0426	0.0043	0.0225	0.0038	**0.051**	**+19.7%**
	Recall@10	0.0481	0.0571	0.0337	0.0388	0.0452	**0.0672**	**+17.7%**
	Recall@20	0.168	0.197	0.126	0.156	0.155	**0.226**	**14.6%**
	HR@5	0.0721	0.078	0.0313	0.0562	0.0681	**0.091**	**+16.7%**
	HR@10	0.136	0.1459	0.08375	0.11	0.125	**0.162**	**+11.1%**
	HR@20	0.205	0.224	0.156	0.192	0.212	**0.253**	**+12.9%**
Electronics	Recall@5	0.0162	0.0186	0.0115	0.01	0.0158	**0.0223**	**+19.8%**
	Recall@10	0.0263	0.0271	0.0215	0.025	0.0254	**0.0301**	**+11.1%**
	Recall@20	0.1682	0.186	0.11	0.135	0.1574	**0.201**	**+8.1%**
	HR@5	0.0595	0.0659	0.0408	0.0408	0.0569	**0.076**	**+15.3%**
	HR@10	0.109	0.109	0.0766	0.0733	0.112	**0.134**	**+19.6**
	HR@20	0.1954	0.2031	0.112	0.1506	0.1685	**0.2325**	**+14.5%**

According to the experiment results, the performances of the 3 baselines (Item2vec, node2vec, BINN) with the item representation are almost higher than that of the remained two baselines, which proves that it is very important to study the representation of the user and the item in recommendation system. In addition to the MGCN4REC algorithm proposed in this paper, Item2vec with aggregated preferences has achieved almost the best performance in all metrics. This is because item2vec algorithm learns item representation and models users' long- and short-term preferences. At the same time, it proves that the user-item representation learning based on multi-graph are effective. Compared with the recommendation system with long- and short-term preferences aggregation, DREAM has the lowest performance, because it only considers users' long-term preferences.

In the Baby dataset, our MGCN4REC model achieves an average improvement of more than 10% in the performance comparing with the other baselines in all metrics. On the Electronics dataset, MGCN4REC outperforms than the baselines on all metrics:

each metric is improved by an average of 15% over the comparison algorithm that works best with the same parameters.

Impact of Multi-graph

We also study the effect of multi-graph user-item representation on the performance of the recommendation system. Table 3 shows the comparison results of MGCN4Rec with and without multi-graph on HR@10.

Table 3. MGCN4Rec with and without multi-graph on HR@10

Datasets	Metrics	Only G_{bought_seq}	Only G_{also_bought}	Only G_{also_view}	Multi-graph
Baby	**HR@10**	**0.138**	**0.143**	**0.139**	**0.162**
Electronics	**HR@10**	**0.116**	**0.118**	**0.117**	**0.134**

According to the Table 3, our model MGCN4Rec with multi-graph outperforms than without multi-graph model over 10% on HR@10. Therefore, the user-item representation based on mulit-graph can better learn the similarity between users and items, then improving recommendation performance.

Impact of Preferences Dimensionality d

We also study the effect of user preferences dimension d on the performance of the recommendation system. Figure 3 shows HR@10 for MGCN4REC and other baselines with the preferences dimensionality d varying from 16 to 256 while keeping other optimal hyper-parameters unchanged. We make some observations from this figure.

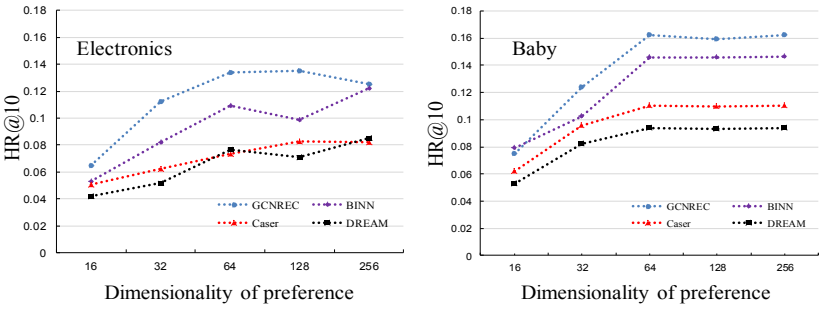

Fig. 3. Effect of the preferences dimensionality d on HR@10 for neural sequential models.

The performance of each model tends to converge as the dimensions increase, and larger dimensions do not necessarily lead to better model performance. Moreover, the model presented in this paper is superior to the baseline in almost all dimensions.

5 Conclucion

This paper proposes a sequential recommendation model based on multi-graph to learn the representation of users and items, and then to model the preferences and instant interest simultaneously. For the part of user-item representation learning, this paper utilize graph convolutional neural network (GCN) to learn the similarity between user-item representation from different graphs. For preferences, this paper utilize recurrent neural network (RNN) to learn users' stable preferences from the shopping basket of historical interaction sequence. For instant interests, this paper utilize the user's question text to model the user's instant interests. In order to aggregate preferences and instant interests, attention mechanism is introduced to calculate users' attention distribution, so as to obtain users' dependence on these two preferences. By comparing with two state-of-art methods of representation learning and three sequential recommendation systems, the results show that our MGCN4REC model can more effectively represent users and items, thus making more effectively recommendation.

References

1. Karatzoglou, A., Baltrunas, L., Shi, Y.: Learning to rank for recommender systems. In: Proceedings of the 7th ACM Conference on Recommender Systems, pp. 493–494. ACM (2013)
2. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **8**, 30–37 (2009)
3. Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 447–456. ACM (2009)
4. Liu, Q., Zeng, X., Zhu, H., Chen, E., Xiong, H., Xie, X., et al.: Mining indecisiveness in customer behaviors. In: 2015 IEEE International Conference on Data Mining, pp. 281–290. IEEE (2015)
5. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 353–362. ACM (2016)
6. Shang, S., Ding, R., Zheng, K., Jensen, C.S., Kalnis, P., Zhou, X.: Personalized trajectory matching in spatial networks. VLDB J. **23**(3), 449–468 (2013). https://doi.org/10.1007/s00778-013-0331-0
7. Yap, G.E., Li, X.L., Yu, P.S.: Effective next-items recommendation via personalized sequential pattern mining. In: Lee, S., Peng, Z., Zhou, X., M, Y.S., Unland, R., Yoo, J. (eds.) DASFAA 2012. LNCS, vol. 7239, pp. 48–64. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29035-0_4
8. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
9. Barkan, O., Koenigstein, N.: Item2vec: neural item embedding for collaborative filtering. In: 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6. IEEE (2016)
10. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701–710. ACM (2014)

11. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
12. Ricci, F., Rokach, L., Shapira, B.: Introduction to recommender systems handbook. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 1–35. Springer, Boston, MA (2011). https://doi.org/10.1007/978-0-387-85820-3_1
13. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 346–353 (2019)
14. Grbovic, M., Cheng, H.: Real-time personalization using embeddings for search ranking at airbnb. In: Proceedings of the 24th ACM SIGKDD International Con-ference on Knowledge Discovery & Data Mining, pp. 311–320. ACM (2018)
15. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th in-ternational conference on World wide web, pp. 811–820. ACM (2010)
16. Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S., Cheng, X.: Learning hierarchical representation model for nextbasket recommendation. In: Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 403–412. ACM (2015)
17. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommenda-tions with recurrent neural networks (2015). arXiv preprint arXiv:1511.06939
18. Yu, F., Liu, Q., Wu, S., Wang, L., Tan, T.: A dynamic recurrent model for next basket recommendation. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 729–732. ACM (2016)
19. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 565–573. ACM (2018)
20. Li, Z., Zhao, H., Liu, Q., Huang, Z., Mei, T., Chen, E.: Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1734–1743. ACM (2018)
21. Donkers, T., Loepp, B., Ziegler, J.: Sequential user-based recurrent neural network recommendations. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 152–160. ACM (2017)
22. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 130–137. ACM (2017)
23. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
24. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Sig. Process. **45**(11), 2673–2681 (1997)
25. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
26. He, R., McAuley, J.: Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In: Proceedings of the 25th International Conference On World Wide Web, pp. 507–517. International World Wide Web Conferences Steering Committee (2016)