Global-Local GraphFormer: Towards Better Understanding of User Intentions in Sequential Recommendation

Hong Chen*
h-chen20@mails.tsinghua.edu.cn
DCST, Tsinghua University

Bin Huang*
huangb23@mails.tsinghua.edu.cn
DCST, Tsinghua University

Xin Wang†
xin_wang@tsinghua.edu.cn
DCST, BNRist,Tsinghua University

Yuwei Zhou
zhou-yw21@mails.tsinghua.edu.cn
DCST, Tsinghua University

Wenwu Zhu†
wwzhu@tsinghua.edu.cn
DCST, BNRist, Tsinghua University

ABSTRACT

Transformer-based model has gained great success in the multimedia sequential recommendation task due to its strong ability to handle sequential data. However, existing Transformer-based models regard the items in the sequential data as a user-specific fully-connected graph (local graph) and only explicitly consider the temporal information in the local graph to capture the users' intentions, ignoring the fact that the user-item bipartite graph (global graph) may carry important relation patterns to the sequential items. Additionally, it is still unclear whether (and how) the information hidden in the global graphs can help the Transformer-based models better understand the users' sequential behavior according to the current literature. To investigate this important problem, we propose to utilize the global graph information to help the Transformer-based sequential recommendation, where the information from different modalities, i.e., user-item interactions in the global graph and the temporal patterns in the historical sequences, are taken into account jointly. In concrete, we propose two Global-Local (GL) GraphFormer models for utilizing both the global graph and local temporal information. One GL-GraphFormer is able to gift the Transformer-based model with both first- and second-order graph information through two specifically designed encodings. The other GL-GraphFormer transfers higher-order graph information into the local Transformer with pretrained Graph Neural Networks (GNNs). Extensive experiments on several real-world datasets demonstrate that i) our proposed GL-GraphFormers can bring substantial improvement over baseline methods, and ii) the benefits of different orders of global graph information vary with the dataset sparsity.

CCS CONCEPTS

• **Information systems → Recommender systems**.

*Equal contribution.

†Corresponding authors. DCST is the abbreviation of Department of Computer Science and Technology. BNRist is the abbreviation of Beijing National Research Center for Information Science and Technology.

MMAsia '23, December 6–8, 2023, Tainan, Taiwan

ACM ISBN 979-8-4007-0205-1/23/12.
https://doi.org/10.1145/3595916.3626377

KEYWORDS

Graph, Sequential data, Transformer, Recommendation

ACM Reference Format:
Hong Chen, Bin Huang, Xin Wang, Yuwei Zhou, and Wenwu Zhu. 2023. Global-Local GraphFormer: Towards Better Understanding of User Intentions in Sequential Recommendation. In *ACM Multimedia Asia 2023 (MMAsia '23), December 6–8, 2023, Tainan, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3595916.3626377

1 INTRODUCTION

With the rapid growth of modern web services and mobile devices, the increasing number of available sequential user behaviors are playing an important role in improving accuracy of multimedia recommendation. The sequential behaviors of a user is driven by their personal evolving intentions in the course of time. A deep understanding of user sequential historical behaviors can help to discover user intentions, thus resulting in more accurate personalized sequential recommendation, which aims at leveraging the sequences of behaviors to recommend the next item that the target user may be interested in. Inspired by the power of the Transformer [24] in handling temporal information in sequential data, recent works [1, 11, 16, 21] utilize the Transformer-based model for sequential recommendation and have achieved impressive success.

The Transformer-based models for sequential recommendation generally regard the historical user behaviors as a user-specific fully-connected graph (local graph), and simply append temporal encoding to the node features to capture user intentions. However, different from traditional sequential data, the items within the historical user behaviors tend to form more complex relations with each other since there exists a global user-item bipartite graph that connects the items through users to other items. For example, if more users simultaneously interact with both item A and item B, then item A and item B should have a closer relationship with each other. Also, if item A is consumed by many users, then its popularity in the global graph may have an impact on its role in the user-specific local sequential fully-connected graph. These intuitive observations indicate that the information from the global graph will influence both the nodes and edges in the local graph. However, existing literature ignores the connection between global and local information, which is important in understanding the user's intentions.

In this paper, we propose to utilize the global graph information to help the Transformer-based sequential recommendation. Our target is to explicitly incorporate the global graph information into

the Transformer, and it is necessary to investigate the following problems:

- What kind of global graph information can help the local Transformer better understand the sequential behavior?
- What is the most appropriate strategy to combine the global graph information with the local sequential temporal information so that the sequential recommendation can be improved?

To address the two problems, we propose two Global-Local Graphformer (GL-GraphFormer) models based on the basic Transformer structure, so that our proposed strategies can fit other models based on the Transformer structure. Specifically, one GL-GraphFormer is designed to utilize the first-order and second-order information of the global graph, i.e., the item popularity and the number of common users consuming a pair of items. We respectively encode the first-order and second-order information to the node and edge features in the local fully-connected graph of the Transformer, expecting this kind of information can help the model to more accurately capture the user intentions. Inspired by the power of Graph Neural Network (GNN) in recommendation [2, 6, 22, 27, 28] for capturing higher-order graph information, the other GL-GraphFormer utilizes the GNN to pretrain on the global graph and then transfer the higher-order knowledge to the local graph. We conduct experiments on several real-world datasets and the results show that i) the proposed two GL-Graphformer models can bring significant benefits to the sequential recommendation and ii) datasets with different sparsity may benefit from different orders of global graph information. We summarize our contributions as follows,

- We propose to investigate the problem of utilizing the global graph information to help the Transformer-based sequential recommendation, which may potentially serve as a novel and promising research trend.
- We present two Global-Local GraphFormer models that combine different orders of global graph information to the local temporal Transformer, which has the advantageous ability of generalization and can be applied to many other Transformer-based sequential recommender systems.
- Experiments on several real-world datasets demonstrate that global graph information employed by our model can significantly improve the sequential recommendation, and datasets with different sparsity benefit from different orders of global graph information.

2 RELATED WORK

General Recommendation. Recommender systems aim to predict to what extent the user will like an item. Early recommender systems generally employ the collaborative filtering [3, 9, 13, 19, 20], based on the matrix factorization idea to learn the latent representation for users and items. With the development of deep learning, more and more deep models are used for recommendation and have gained great success. Multi-Layer-Perceptron(MLP) has been used to capture the non-linear relationships between user and item embeddings [7]. Auto-encoder based methods show robustness to behavior noise [14]. Due to the graph nature of user-item relation, recent efforts [2, 6, 22, 27] design different kinds of Graph Neural Networks(GNN) for collaborative filtering. The GNN based models take the advantage of message passing to obtain high-order connectivity from the user-item interaction and show strong ability in improving the recommendation accuracy. However, these works neglect the sequential nature of the user's behavior, which may limit them for more personalized recommendation.

Sequential Recommendation. With consideration to the sequential characteristics of user's behavior, another line of recommendation, sequential recommendation, has gained increasing attention in the community. The sequential recommender aims to predict the user's preference towards an item based on the user's personalized historical behavior. Early sequential recommenders utilize the first or second-order Markov chains [23, 25] to model the users' sequential behavior. Inspired by the expressive power of sequential deep models, later works [8, 10, 30] have achieved further improvement by utilizing the recurrent neural networks and the more recent Transformer-based models [1, 11, 16, 21]. The Transformer-based models generally regard the items in the sequential historical behavior as a fully connected graph with temporal encoding for nodes, and then utilize the self-attentive mechanism to extract user intentions from the graph. However, this kind of graph construction method neglects the complex relationships among the historical items in the global user-item bipartite graph, which has shown effectiveness in general collaborative filtering. Although some other works [15, 29, 31] provide other ways to construct the graph for the sequential historical items, like utilizing additional sequences [26]. However, there are still rare works exploring whether the bipartite global graph which shows strong ability in the general recommendation can help the Transformer-based sequential recommendation.

Long in history, the global graph information has been utilized for general recommendation through graph representation learning models such as GNNs. The novel idea of marrying global graph information with local fully-connected graphformer, as well as experimental discoveries on the benefits brought by different orders of graph information in this work, can be regarded as a pivot to bridge the traditional graph-based general recommendation with the sequential recommendation.

3 METHOD

In this section, we present the two Global-Local GraphFormer. The overall framework of the two GL-GraphFormer in fig. 1. We will first introduce the preliminaries and how Transformer is generally used for extracting the user's intention in sequential recommendation in section 3.1. Then, in section 3.2 and section 3.3, we respectively give the detailed description for how we design the two GL-GraphFormer. Finally, in section 3.4 we describe how we train our GL-GraphFormer.

3.1 Preliminaries

Assuming that there are totally M users and N items, constituting the user set U and the item set I. Then the users and the items will form a bipartite graph $G = \{V, E\}$, which we call the global graph. The nodes in V contain all the users and items, i.e., $V = \{U, I\}$, $|V| = M + N$, and there exist an edge in E if a user has interaction with an item. Note that the edges will only exist between the user node and the item node, there is no edge between two users or between two

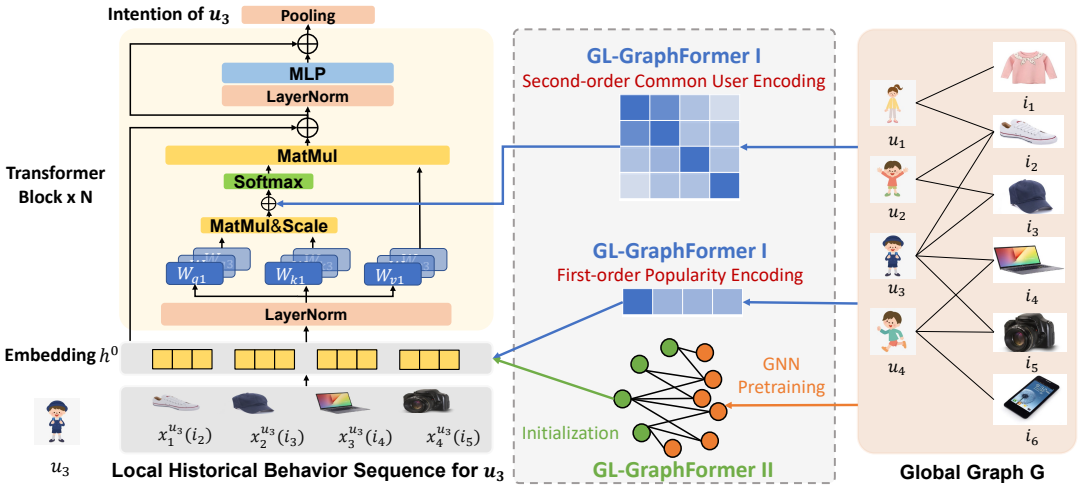

Figure 1: The overall framework of the GL graph-former for sequential recommendation. We explore two ways to combine the global graph information to the Transformer-based encoder. One is to add the first and second-order encoding to the item and relation embedding. The other is to utilize GNN to capture higher-order item interaction and initialize the item embeddings with the pretrained embeddings in GNN.

items. In the general recommendation utilizing GNN, we usually split the edges in E into the training, validation and testing dataset. The task is to utilize the edges in the training dataset to conduct the message passing so that the model can predict the edges in the validation and test dataset.

In the sequential recommendation scenario, we focus on the user-centric data, the user's sequential historical behavior. For each user $u \in U$, we have their historical behavior, $\mathbf{x}^u = \{x_1^u, x_2^u, \cdots, x_T^u\}$. $\mathbf{x}^u$ contains T items user u recently interacted with in chronological order, and each $x_i^u \in I$. The task for the sequential recommendation is to predict the next item that the user u will most likely interact with. Next, we describe how the general Transformer-based sequential recommender extract the users' future intention.

We first map each item x_i^u in the historical sequence to their embedding and obtain the embedding sequence, $\mathbf{m}^u = [m_1^u, m_2^u, \cdots, m_T^u]$, $\mathbf{m}^u \in R^{T\times d}$, where d is the dimension of the item embedding. Then to capture the temporal relations of each item, each item is equipped with its time position embedding, and then we obtain the time-aware item embedding z_i^u as follows,

$$z_i^u = f(m_i^u, p_i), i = 1, 2, \cdots, T \tag{1}$$

where p_i is the embedding for position i, $f()$ is a function that is used to fuse the item embedding and the position embedding. The widely adopted f could be adding the two embedding(assuming the two embedding have the same dimension) or concatenating them followed by a linear layer, etc. For simplicity, we adopt the addition operation for $f()$ in our experiment and assume the dimension of z_i^u is d. The Transformer encoder takes the time-aware historical embedding sequence $\mathbf{z}^u = [z_1^u, z_2^u, \cdots, z_T^u]$ as input and will output the representation for the user's future intention. The Transformer-based model can be stacked with many Transformer encoder blocks. The input for each Transformer encoder block is a sequence of embedding with the same size $T \times d$. Denoting the input for the l^{th} block as $h^{l-1} \in R^{T\times d}$, we describe the flows in one block of the Transformer encoder. Each Transformer encoder block is mainly composed of the multi-head attention layer and the feed-forward layer. The multi-head attention layer is conducted as follows,

$$K_j = h^{l-1}W_{kj},\ Q_j = h^{l-1}W_{qj},\ V_j = h^{l-1}W_{vj}, \tag{2}$$

$$A_j = \frac{Q_j K_{j'}}{\sqrt{d_k}},\ Attn(h^{l-1})_j = softmax(A_j)V_j, j = 1, \cdots, m, \tag{3}$$

where $W_{kj}, W_{qj}, W_{vj} \in R^{d\times d_k}$, $d = d_k \times m$, and m is the number of the head. In Eq.(2), each item embedding in h^{l-1} will obtain its key K_j, query Q_j and value V_j under the j^{th} head. Then in Eq.(3), $A_j \in R^{T\times T}$ gives the relation matrix of the items under the j^{th} head, and $Attn(h^{l-1})_j \in R^{T\times d}$ gives the user's intention at each time stamp under the j^{th} head. Finally, we concatenate the output of all the heads and obtain $Attn(h^{l-1}) = [Attn(h^{l-1})_j]_{j=1}^m$, and $Attn(h^{l-1}) \in R^{T\times d}$. To accelerate convergence, the LayerNormalization(LN) is usually conducted before the multi-head attention and the skip connection is adopted. We obtain the hidden state h_1^l after the multi-head attention as follows,

$$h_1^l = h^{l-1} + Attn(LN(h^{l-1})) \tag{4}$$

After the multi-head attention, a Multi-Layer-Perceptron(MLP) is used to model the non-linear combination for different intentions. Similar to the multi-head attention, the LayerNormalization and the skip connection is adopted, we obtain the output of one Transformer encoder block as follows,

$$h^l = h_1^l + MLP(LN(h_1^l)) \tag{5}$$

h^0 is usually initialized with the historical embedding sequence $\mathbf{z}^u$. The Transformer-encoder block can be stacked multiple times, it depends on how many orders of relations we want to capture among the items. Assuming that we stack the Transformer-encoder block for total M times and we will obtain the final $h^M \in R^{T\times d}$,

which contains the intentions of user u. Usually, the intention of user u with the previous T historical behavior is obtained from the pooling result of h^M. Different pooling methods are adopted to extract the user intention from h^M in previous works. SASrec [11] adopt the last pooling, i.e., regarding the h_T^M, the last embedding in the embedding sequence h^M, as the user's future intention, while CDR [1] adopt the average of T embeddings in h^M as the user's future intention.

$$intent_T^u = pooling(h^M) \tag{6}$$

Next, we describe the two proposed GL-GraphFormer that combines the global graph information with the temporal sequential recommendation.

3.2 GL-GraphFormerI: First and Second-order Information Encoding

We focus on better modeling the relations among the items in the sequence of the user's historical behavior. For each item in the global graph G, its information of the graph can be obtained from its neighbors or high-order neighbors. In this part, we only consider the first-order information and the second-order information, and propose two kinds of encoding for the Transformer-based model.

3.2.1 Item popularity encoding. The first-order neighbors of an item are the users who have interaction with this item. If many users have interaction with one item, this item may play a different role in the historical behavior. This observation fits a common concept in recommendation, item popularity. Users' purchase behavior is not only determined by their preferences but also affected by the popularity of the items. People always give priority to more popular items, but this kind of popular item may hardly reflect each user's personalized interest. Explicitly incorporating the item popularity information can make the Transformer better capture the general interests of all users and the personalized interest for each user.

To this end, we define item popularity as the degree of an item in the global user-item bipartite graph G, i.e., the number of the first-order neighbors of an item. We develop item popularity encoding according to the degree of the item, and add it to the time-aware item embedding as follows,

$$z_i^u = m_i^u + p_i,\ h_i^0 = n_{deg(x_i^u)} + z_i^u, \tag{7}$$

where h^0 is the input for the first Transformer block and h_i^0 is the i^{th} embedding in h^0, m_i^u is the embedding of x_i^u, $n \in R^d$ is learnable embedding vectors specified by the degree $deg(x_i^u)$, and p_i is the position embedding. By using item popularity encoding, the GL-GraphFormer can utilize the first-order information in the global graph and capture the item importance in the attention layer.

3.2.2 Common user encoding. The second-order neighbors of an item in the global bipartite graph G are the items who have common users with this item. The relations between two items can be bridged by the user who has interacted with both of them. If more users interact with both item i_1 and item i_2, item i_1 and item i_2 should have a closer relationship with each other. However, such information is not available in the sequential Transformer recommender. We propose to explicitly encode this information into the local Transformer. Specifically, we use the common user encoding to solve this problem. For item x_a^u and item x_b^u in the historical sequence $\mathbf{x}^u$, we denote the number of users that bought them simultaneously as $s_(a, b)$ and $a, b \in \{1, 2, \cdots, T\}$. This second-order information describes the number of common users between two items, so it is a natural practice to encode this information to the relation matrix A_j in Eq.(3).

$$A_{j,ab} = \frac{Q_{j,a} \cdot K'_{j,b}}{\sqrt{d_k}} + c_{s_{(a,b)}}, j = 1, 2, \cdots, m, \tag{8}$$

$$Attn(h^{l-1})_j = softmax(A_j)V_j, j = 1, 2, \cdots, m, \tag{9}$$

where $A_{j,ab}$ is the (a, b)-element of the relation matrix as shown in Eq.(3). $c_{s_{(a,b)}}$ is a learnable scalar indexed by $s_{(a,b)}$, and shared across all layers. The kind of encoding will be added for all the m attention heads.

By introducing common user encoding, we inject the second-order global graph information into the local Transformer explicitly, expecting the GL Transformer can pay more attention to the items that have more common users.

In sum, the item popularity encoding focuses on the first-order information of each item and the common user encoding focuses on the second-order information. Regarding the items in the historical sequence in the Transformer encoder as a fully-connected graph, the item popularity encoding focuses on the node features and the common user encoding focus on the edge features. Replacing the h_i^0 in section 3.1 with Eq.(7) and replacing Eq.(3) with Eq.(8)(9) gives the complete GL-GraphFormerI.

3.3 GL-GraphFormerII: High-order Information through Graph Pretraining

In section 3.2, we introduce two kinds of encoding methods that combine the first- and second-order global graph information into the local Transformer. In this section, we explore whether higher-order global graph information can bring further benefits.

However, higher-order information of the global graph becomes less intuitive than the first- and second-order information, and it is hard to come up with a natural encoding method like section 3.2. To tackle this problem, inspired by the power of Graph Neural Networks(GNN) in capturing the high-order interactions among nodes, we utilize a pretrained GNN on the global graph to gift the item embeddings with higher-order knowledge. For simplicity, we adopt the efficient and effective LightGCN [6] structure for collaborative filtering as the pretrained knowledge extractor. Next, we describe the details of the adopted GNN and how we utilize it to extract the higher-order knowledge.

3.3.1 Message Passing. The parameters for LightGCN are only the embeddings for the users and items, and it only relies on the linear message passing to obtain the higher-order interaction between different nodes. Recall that the bipartite graph $G = \{V, E\}$, where the nodes V contain the user set U and the item set I. For user u in U and item i in I, their embeddings are respectively $\mathbf{e}_u^0$ and $\mathbf{e}_i^0$ with dimension d. Then, in the k^{th} layer of the LightGCN, the model

will conduct linear message passing as follows,

$$\mathbf{e}_u^k = \sum_{i \in N_u} \frac{1}{\sqrt{deg(u)}\sqrt{deg(i)}} \mathbf{e}_i^{k-1}, \tag{10}$$
$$\mathbf{e}_i^k = \sum_{u \in N_i} \frac{1}{\sqrt{deg(i)}\sqrt{deg(u)}} \mathbf{e}_u^{k-1},$$

where N_u and N_i means the neighbors of user u and item i in G, and $deg()$ is the same as our previous notation for the degree of one node. In each layer of the LightGCN, the embedding of each user will be updated by the linear combination of its neighbor item embeddings, and symmetric for the embedding of each item. The degree of the items and users are used for normalization to avoid the embedding scale explosion.

3.3.2 Model Pretraining. The embeddings of each layer are averaged to make the final embedding contain different orders of information. The final embedding for user u and item i is obtained as follows,

$$\mathbf{e}_u = \sum_{k=0}^{L} \frac{1}{L+1} \mathbf{e}_u^k, \ \mathbf{e}_i = \sum_{k=0}^{L} \frac{1}{L+1} \mathbf{e}_u^i. \tag{11}$$

Then whether user u will interact with item in the global graph G i is calculated with the inner product: $\hat{y}_{ui} = \mathbf{e}_u^{'} \mathbf{e}_i$. Then the pretraining of the LightGCN parameters will be conducted by optimizing the following Bayesian Personalized Ranking(BPR) loss [19].

$$L_{BPR} = -\sum_{u \in U} \sum_{i \in N_u} \sum_{j \notin N_u} ln(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda ||w||^2, \tag{12}$$

where λ is the coefficient for the L2 regularizer. After training on the graph of the training dataset, we can obtain the embedding for each item. Replacing the m_i^u with the pretrained item embedding gives the GL-GraphFormerII. In our experiments, we utilize the pretrained $\mathbf{e}_0^i$ for item i in the historical sequence, where $\mathbf{e}_0^i$ implicitly contains the higher-information through the back-propogation process. Also, we provide the results utilizing the pooling embedding $\mathbf{e}^i$ in section 4.

3.4 GL-GraphFormer Training

In this section, we describe how we train the GL-GraphFormer for sequential recommendation. With the GL-GraphFormer, we can obtain the intention for user u from their historical sequence $\mathbf{x}^u = \{x_1^u, x_2^u, \cdots, x_T^u\}$ using Eq.(6). Based on $intent_T^u$, we predict the user's preference towards an item i, $r_{u,i}^T$, with their inner product,

$$r_{u,i}^T = < intent_T^u, e_i >, \tag{13}$$

where e_i is the embedding for item i. We adopt the same prediction loss as [11] as follows,

$$L_{ui,T} = log(\sigma(r_{u,i}^T))|_{i=target^T} + log(1 - \sigma(r_{u,j}^T))|_{j \neq target^T}, \tag{14}$$

where $target^T$ is the ground truth item and we follow [11] to sample one negative item j for each time step for user u.

4 EXPERIMENTS

In this section, we present the experimental setup and results.

Datasets. We conduct experiments on 5 recommendation datasets. Two of them are from Amazon [5], named Amazon-Beauty and Amazon-Games. Two of them are from MovieLens [4], named MovieLens-1M and MovieLens-10M. Another one is the Steam [17]. These datasets have different scales and sparsity. The statistics of the datasets are shown in table 1. We split the datasets as the previous works [11, 16, 21], i.e., using the last item of each user's sequence for test and the last but one for validation, and the remaining data are used for training.

Table 1: Dataset statistics

	Beauty	Games	Movie-1M	Movie-10M	Steam
#users	52,024	31,013	6,040	71567	334,730
#items	57,289	23,715	3,416	10677	13,047
avg.seq	7.6	9.3	163.5	139.7	11.0

Implementation Details. We implement our model based on the basic Transformer structure, the same as SASRec [11], the earliest work utilizing Transformer for sequential recommendation. Many later works [1, 16, 21] add more structures based on SASRec, thus the improvement on the basic model will be more meaningful. The layer for the Transformer block is 2 and the hidden dimension d is searched from{10,20,30,40,50} using the basic Transformer model(SASRec). The optimizer is Adam [12] with learning rate 1e-3. The batch size is 128 and the dropout rate for the MovieLens is 0.5 and 0.2 for other datasets. The pretrained LightGCN model is a 3-layer model recommended by the original paper and optimized with BPRLoss. The optimizer is Adam with learning rate 1e-3. The batch size is 2048, and the model is trained for 400 epochs. Additionally, when utilizing the pretrained embedding to initialize the historical sequence item embedding, we scale each embedding by dividing a scale factor 5.0. This operation is because the pretrained embedding from LightGCN has a large scale and directly using them for initialization causes hard convergence.

Recommendation Performance. We evaluate the models in terms of Recall@10 and normalized discounted cumulative gain (NDCG@10), which are two widely adopted metrics in recommendation. Higher value in both metrics indicates better performance. Different from SASRec, we do not calculate these metrics with the sampled negative items, we calculate the metrics with all the items, which is more accurate as indicated in previous work [18].

The experimental results are shown in table 2. We report the performance of the basic Transformer-based sequential recommender, SASRec, and LightGCN, the model we utilize for pretraining. The GL-GFormerI and GL-GFormerII indicates the two Global-Local GraphFormer in section 3.2 and section 3.3. Except for these models, we also explore the variants with different combinations of the proposed first-, second- and higher-order information .

From the results in table 2, we can see that the sequential recommender generally performs better than the collaborative filtering model in sequential recommendation (except on the Beauty dataset). More importantly, we obtain the following observations:

- **Incorporating the global graph information into the local Transformer can bring benefits.** Comparing the SASRec baseline model with two kinds of GL-GraphFormer, we can see that on the Amazon-Beauty, Amazon-Games, and Steam dataset, GL-GFormerII achieves significant improvement than SASRec. On

Table 2: Model performance. The variants of the GL graph-former as denoted as follows: Base means the original Transformer(SASRec) for sequential recommendation. 1 means adding the first-order global graph information (popularity encoding), 2 means adding the second-order common user encoding, and P means using the pretrained embedding from LightGCN. Pretraining of LightGCN on the MovieLens-10M has high computational cost because of the huge number of interactions, and we report it as out of time(OOT).

Model	Dataset									
	Amazon-Beauty		Amazon-Games		MovieLens-1M		MovieLens-10M		Steam	
	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
SASRec	0.0138	0.0072	0.0605	0.0295	0.2450	0.1304	0.1256	0.0621	0.1140	0.0606
LightGCN	0.0253	0.0122	0.0532	0.0264	0.0819	0.0408	OOT	OOT	0.0673	0.0351
GL-GFormerI	0.0150	0.0074	0.0584	0.0298	0.2551	0.1382	0.1269	0.0628	0.1147	0.0652
GL-GFormerII	0.0294	0.0152	0.0755	0.0390	0.2440	0.1303	OOT	OOT	0.1268	0.0718
Base+1	0.0125	0.0066	0.0650	0.0334	0.2505	0.1361	0.1229	0.0608	0.1165	0.0644
Base+2	0.0125	0.0055	0.0615	0.0291	**0.2608**	**0.1416**	**0.1298**	**0.0634**	0.1251	0.0711
Base+1+P	0.0288	0.0148	0.0771	0.0382	0.2531	0.1336	OOT	OOT	0.1214	0.0677
Base+2+P	**0.0320**	**0.0168**	**0.0795**	**0.0400**	0.2550	0.1358	OOT	OOT	**0.1283**	**0.0722**
Base+1+2+P	0.0242	0.0119	0.0737	0.0369	0.2570	0.1358	OOT	OOT	0.1180	0.0660

the two MovieLens dataset, the GL-GFormerI brings substantial recommendation accuracy improvement.

- **The benefits of the global graph information to the Transformer are highly related to the dataset sparsity.** We can see that GL-GFormerI shows better performance on the dense MovieLens-1M and MovieLens-10M dataset, while GL-GFormerII performs better on the more sparse dataset Amazon-Beauty, Amazon-Games and Steam. More specifically, with the results of the variants "Base+X"(X=1,2,1+P, etc.), we can see that the first-order encoding (item popularity encoding) does not play a significant role under different datasets. The second-order encoding brings the best performance on the MovieLens dataset, and the combination of the second-order encoding and pretraining gives the best performance on the sparse dataset. These phenomenons indicate that more sparse dataset needs higher-order global graph information. It is not hard to understand this observation. When the dataset becomes sparse, the common items between two users will become fewer, more valuable relations between items will come from higher-order information.

Effects of Different Initialization. In our previous experiments for GL-GraphFormerII, we utilize the 0^{th} layer of embedding from LightGCN because the backpropogation process will make the item embedding in the 0^{th} layer aware of their higher-order neighbors. We also provide the results of utilizing the average pooling embedding of different LightGCN layers as initialization. The results are shown in table 3, and we observe that utilizing pooling embedding shows comparative performance on the Beauty and Steam dataset, but achieves clear improvement on the Games dataset because the average pooling explicitly contains different order of graph information.

Visualization. We provide a Top1 recommendation example of a one-layer GL-GraphFormer and the original one-layer SASRec. The ground truth is within the Top10 of GL-GFormer while but beyond the Top10 of SASRec recommendation. The SASRec baseline wrongly focuses on the shampoo and gives the wrong matching, indicating the attention mechanism and the learned embedding for the items in the SASRec are not so good. In contrast, our method

Table 3: Performance@10 of different pretrain initialization

Initialization	Dataset					
	Beauty		Games		Steam	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
0^{th} layer	0.0294	0.0152	0.0755	0.0390	0.1268	0.0718
pooling	0.0303	0.0153	0.0830	0.0435	0.1259	0.0722

Figure 2: Attention in GL-GraphFormer and SASRec

gives more reasonable attention about facial cosmetics and predicts the right moisturizing intention of the user.

5 CONCLUSION

In this paper, we present a novel and promising perspective, utilizing the bipartite user-item global graph information to enhance the Transformer-based sequential recommendation. We offer two Global-Local GraphFormers, effectively improving the performance of Transformer model for sequential recommendation. This work also discovers that more sparse datasets benefit from higher-order global graph information. Further investigations including exploring more ways of utilizing the global graph information for sequential recommendation and designing more advanced models based on the GL-GraphFormer will be a promising future direction.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China No.2020AAA0106300, National Natural Science Foundation of China (No. 62222209, 62250008, 62102222), Beijing National Research Center for Information Science and Technology under Grant No. BNR2023RC01003, BNR2023TD03006, and Beijing Key Lab of Networked Multimedia.

REFERENCES

[1] Hong Chen, Yudong Chen, Xin Wang, Ruobing Xie, Rui Wang, Feng Xia, and Wenwu Zhu. 2021. Curriculum Disentangled Recommendation with Noisy Multi-feedback. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. 26924–26936.

[2] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting Graph Based Collaborative Filtering: A Linear Residual Graph Convolutional Network Approach. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press, 27–34.

[3] Mukund Deshpande and George Karypis. 2004. Item-based top-N recommendation algorithms. *ACM Trans. Inf. Syst.* 22, 1 (2004), 143–177.

[4] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4 (2016), 19:1–19:19.

[5] Ruining He and Julian J. McAuley. 2016. 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, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao (Eds.). ACM, 507–517.

[6] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 639–648.

[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*. ACM, 173–182.

[8] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. ACM, 843–852.

[9] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 263–272.

[10] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*. ACM, 505–514.

[11] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[12] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[13] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[14] Xiaopeng Li and James She. 2017. Collaborative Variational Autoencoder for Recommender Systems. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 305–314.

[15] Chen Ma, Liheng Ma, Yingxue Zhang, Jianing Sun, Xue Liu, and Mark Coates. 2020. Memory Augmented Graph Neural Networks for Sequential Recommendation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*. AAAI Press, 5045–5052.

[16] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled Self-Supervision in Sequential Recommenders. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 483–491.

[17] Mark O'Neill, Elham Vaziripour, Justin Wu, and Daniel Zappala. 2016. Condensing Steam: Distilling the Diversity of Gamer Behavior. In *Proceedings of the 2016 ACM on Internet Measurement Conference, IMC 2016, Santa Monica, CA, USA, November 14-16, 2016*. ACM, 81–95.

[18] Steffen Rendle. 2019. Evaluation Metrics for Item Recommendation under Sampling. *CoRR* abs/1912.02263 (2019).

[19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. *CoRR* abs/1205.2618 (2012). http://arxiv.org/abs/1205.2618

[20] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*. ACM, 285–295.

[21] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 1441–1450.

[22] Jianing Sun, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, Xiuqiang He, Chen Ma, and Mark Coates. 2020. Neighbor Interaction Aware Graph Convolution Networks for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 1289–1298.

[23] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*. ACM, 565–573.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 5998–6008.

[25] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning Hierarchical Representation Model for NextBasket Recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*. ACM, 403–412.

[26] Wen Wang, Wei Zhang, Shukai Liu, Qi Liu, Bo Zhang, Leyu Lin, and Hongyuan Zha. 2020. Beyond Clicks: Modeling Multi-Relational Item Graph for Session-Based Target Behavior Prediction. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*. ACM / IW3C2, 3056–3062.

[27] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. ACM, 165–174.

[28] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 1001–1010.

[29] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xianling Mao, and Minghui Qiu. 2020. Global Context Enhanced Graph Neural Networks for Session-based Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 169–178.

[30] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. ACM, 495–503.

[31] Huachi Zhou, Qiaoyu Tan, Xiao Huang, Kaixiong Zhou, and Xiaoling Wang. 2021. Temporal Augmented Graph Neural Networks for Session-Based Recommendations. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 1798–1802.