Proceedings of the 2023 International Conference on Mechatronics and Smart Systems
DOI: 10.54254/2755-2721/28/20230190

Sequential recommendation based on graph transformer

Zixuan Sun

Sun Yat-sen University, Shenzhen, Guangdong 528406, China

sunzx3@mail2.sysu.edu.cn

Abstract. Sequential Recommendation (SR) is an important scenario in recommendation tasks. Sequential recommendations model the sequential pattern between item-item or user-item based on a user's recent activity in a time series to predict their next preference. However, existing methods are based only on the conventional Graph Neural Networks (GNN) as a model architecture for adaptive fine-tuning of specific SR tasks. To get better recommendation results, more advanced GNNs can be used as the network architecture of the SR method. This paper introduces graph transformer, a combination of GNN and a good sequential task processing model. Then a cross-sectional comparison is made with the current SR method model and its suitability for application in SR tasks is discussed. The comparison shows that the graph transformer is similar in principle and structure to the current SR models, and requires the addition of some adaptive components to be applied in SR tasks. The superior performance after application can be demonstrated from the results data of the Benchmarking-GNNs and Long-Range Graph Benchmark on the models.

Keywords: Sequential Recommendation, transformer, Graph Neural Network, graph transformer.

1. Introduction

Recommender system is one of the crucial information service technologies in the field of e-commerce nowadays, and its appearance enables Internet companies to gain considerable revenue. To enhance and improve the user experience, it is necessary for the system to be able to filter out personalized information from the huge amount of information, so Recommender System is proposed to achieve personalized information presentation for users and is dedicated to solving the problem of information overload. Users explore the web for items that interest them, and recommendation systems can accurately model user preferences based on their interaction history. Sequential recommendation (SR) is a very important task in a recommendation system. This paper focuses on obtaining better recommendation performance by applying better network design in SR tasks.

The specific SR task is to predict the user's future choice based on the user's recent behaviours. This is a kind of temporal preference, so modelling serial features is more appropriate. To improve the recommendation performance to obtain excellent prediction results, it is necessary to extract as much valid information from the sequence as possible. An illustration of sequential recommendation is shown in Figure 1.

DOI: 10.54254/2755-2721/28/20230190

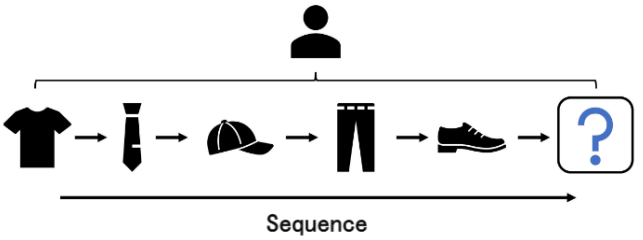

Figure 1. An illustration of sequential recommendation is cited from [1].

Frankly speaking, the history of recommender systems is presented in more detail in two high-quality review articles in [1][2], describing from the past collaborative filtering systems (CF), matrix decomposition methods, etc., to the present day, using deep learning networks. Essentially, most of the information in a recommendation system is presented as a graph structure, and Graph Neural Networks (GNN) have some advantages for representation learning (RL), which can use graphs to obtain high-quality embedding. Therefore, it is becoming increasingly popular to use GNNs to convert the user's sequential behaviour into a graph structure such as a sequence graph to capture the transformation patterns from the user's sequential actions.

For specific sequential recommendation tasks, currently used GNNs-based recommendation models are largely dependent on supplementary sequence models [1]. In SURGE [3] GNN started as a feature extractor only, assisting RNN with sequence modelling. Later in models such as TGSRec [4] the whole task is done using GNN only and the performance of the model is further optimized. Sequential recommendations are essentially sequential tasks, and the transformer [5] was shown to have excellent in-sequence task processing capabilities. Transformer was originally proposed for the task of processing sequential data in natural language processing, but its encoder-decoder structure and proposed self-attention mechanism perform equally well in other computer domains. GNN fusion transformer architecture can be tried for Sequential recommendation tasks.

In this paper, this paper first discusses the current advanced graph transformer model and the commonality of the Sequential Recommender System (SeqRS) model. In the discussion citing specific papers [6-8] work, the graph transformer and the currently popular SeqRS model have similar network structures and the application of the method, verifying the feasibility of the graph transformer model. Then, this paper discusses and points out the superiority of graph transformer over the current SeqRS model and applying its ideas directly or indirectly in the SR task will further enhance the performance of the model in real-life situations. The experimental data of [9] is cited at the end to enhance the reliability of the previous results at the data level.

2. Related Work

Before the formal discussion, we will briefly introduce a mathematical representation of the SR task. Also, some basic information about GNN and transformer is provided in this section to aid a better understanding of what will be discussed later.

2.1. Problem Formulation

In the sequential recommendation task, we have a set of M users $\mathcal{U} = \{u_1, u_2, \dots, u_M\}$ and a set of N items $I = \{i_1, i_2, \dots, i_N\}$. The user-item interaction matrix is represented by $Y \in \mathbb{R}^{M \times N}$, where $y_{ui} = 1$ represents that user u interacts with the item i and otherwise, $y_{ui} = 0$. A session is a succession of sequential interaction objects that a user can interact with in distinct time steps. Given a user u, one of his sessions is denoted as $S^u = (s_1^u, s_2^u, \dots, s_L^u)$, where $L = |S^u|$ is the session length and $S_j^u \in I$ is an item index that the user has interacted with throughout the session. I_u is denoted as a set of items that the user u has interacted with. Ultimately, the goal of the task is to predict a list of items from the item set I as a recommendation to each other, based on the earlier session $S_{1:t}^u$ $(t < L)$ of every user $u \in \mathcal{U}$.

DOI: 10.54254/2755-2721/28/20230190

The main purpose of putting in this paper is to highlight the better prediction recommendation results on the performance of the model on a specific dataset.

2.2. Graph Neural Networks

Most of the modelling operation in the SR task is based on item-item interaction and user-item interaction. Recommender systems store a large amount of user-item interaction data, which can be presented in graphs. The use of graph data is of great help to eliminate data sparsity and cold starts in recommender systems. Compared with other deep learning models, GNN-based models have significant advantages for feature extraction tasks on non-Euclidean structured data. The stacking structure of graph neural networks enables each node of the graph to access information from higher-order neighbours, enhancing collaborative filtering signals. In addition, GNNs use the de-aided propagation of edges, integrate the states of nodes and neighbours, update the state of the current node, and the structured information is captured by the model and expressed on each node, thus solving the problem of the sparsity of recommender systems. Therefore, GNN-based recommender is very popular recently.

By aggregating the features of nearby nodes, GNN and GCN models update the features of the current central node. Let $G = (V, E)$ denote a graph where $V = \{v_1, v_2, \cdots, v_n\}, n = |V|$ is the number of nodes. The node update formula of GCN is as

$$h_{\mathrm{i}}^{l+1} = \delta\left(W_{self}^{l} h_i^l + W_{neigh}^{l} \sum_{j \in N(i), i \neq j} h_j^l\right) \tag{1}$$

where $N(i)$ is the set of first or higher-order neighbours of $v_{I,}$ h_i^l as the representation of v_I at the *l-th* layer, δ represents a nonlinear function (e.g. tanh or ReLU), W_{self}^{l} and W_{neigh}^{l} are trainable parameter matrices, the bias term is omitted for simplicity of the equation.

However, the training of GCN requires knowledge of the entire graph structure (including the nodes to be predicted), which is not possible in a realistic recommender system. In addition, a pooling function is furthermore needed to transfer a collection of node representations into a compact form to achieve an accurate representation of the graph. To integrate the representations of the objects in a sequence, an attention mechanism can be utilized.

2.3. Graph Attention Networks [10]

To incorporate the attention mechanism, Graph Attention Network (GAT) can be easily associated. graph Attention Network (GAT) proposes a weighted summation of neighbouring node features using the attention mechanism. GAT introduces masked self-attention and multi-head attention mechanism based on GNN (using multiple $\mathrm{W^k}$ to calculate self-attention at the same time), which solves the problem that GNN aggregates neighbour nodes without taking the different importance of different neighbour nodes into account. Let the input feature $h = \{\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{\mathcal{N}}}\}, \overrightarrow{h_{\mathrm{i}}} \in \mathbb{R}^F$, and the normal node formula (does not consider multi-head attention) of GAT is obtained as:

$$\overrightarrow{h_{\mathrm{i}}'} = \sigma\left(\sum_{j \in \mathcal{N}_I} \alpha_{ij} W \overrightarrow{h_{\mathrm{j}}}\right) \tag{2}$$

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU\left(\vec{a}^T [W\overrightarrow{h_{\mathrm{i}}} \| W\overrightarrow{h_{\mathrm{j}}}]\right)\right)}{\sum_{\kappa \in \mathcal{N}_I} \exp\left(LeakyReLU\left(\vec{a}^T [W\overrightarrow{h_{\mathrm{i}}} \| W\overrightarrow{h_{\kappa}}]\right)\right)} \tag{3}$$

where $W \in \mathbb{R}^{F' \times F}$ is a weight matrix applied to every node and $\mathcal{N}$ is the number of nodes, $\cdot^{\mathrm{T}}$ represents transposition and $\|$ is the concatenation operation. The structure of the graph attention layer in the GAT is shown in figure 2.

DOI: 10.54254/2755-2721/28/20230190

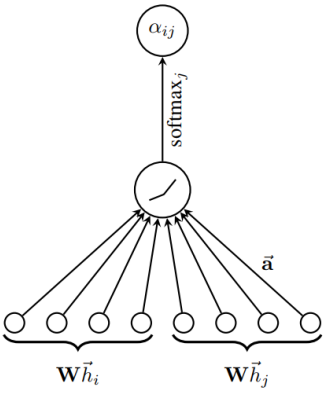

Figure 2. One of the graph attention layers in GAT is cited from [10].

GAT does not require the use of pre-constructed graphs and can also be used for Inductive learning, enabling the processing of unseen graph structures. But GAT does not make full use of the edge information, only the connectivity. If GAT is applied to the SR tasks, a large amount of potentially valid information will be lost.

2.4. Transformer [5]

Based on the preceding inference, the transformer, which has become very popular in sequential tasks recently, can be introduced into the GNN network structure. Transformer originally comes from the field of NLP, which uses a self-attentive mechanism to construct the features of each word. Transformer is a global attention mechanism, which can also be seen as a fully connected GAT.

Transformer takes the $\{Q(query) = Q^l H^l, K(Key) = K^l H^l, V(Value) = V^l H^l\}$ to construct the attention mechanism (e.g. the current word h_l^l has query value $Q^l h_l^l$), where both the triple and the output are vectors, the output is a weighted sum of $V(Value)$, whose weights are the values computed from the corresponding combinations of $Q(query)$ and $K(Key)$.The node formula of the transformer is obtained as：

$$h_i^{l+1} = \delta(\sum_{j \in S} \omega_{ij}(V^l h_j^l) \tag{4}$$

where $\omega_{ij} = softmax\left(Q^l h_i^l \cdot K^l h_j^l\right)$. The formulas of GNN and transformer are similar. When the data is a fully connected graph, then GNN's node formula is equivalent to transformer's node formula to some extent, which provides a feasible basis for combining GNN and transformer. Therefore, the combination of GNN and transformer is a sensible approach.

3. Methods

This section will discuss the feasibility of graph transformers in sequential recommendation tasks and their superiority over the current SeqRS model. To apply GNNs well to the SR task, the feasibility, and adaptability of graph transformer in the SR task should be discussed first. Second, the advanced nature of graph transformer should also be noted. Graph transformers is compared in two main aspects: optimization of GNN-based models and control of the computational cost of current SeqRS methods. This part will be talked about in the discussion of superiority.

3.1. Graph Transformer

Graph transformer combines GNN and transformer, which retains the features of both to form a model with better performance. There are also many ways to combine GNN and transformer. Nowadays, there are many approaches to splice the transformer directly into the GNN model, e.g. U2GNN [11] uses the transformer only in aggregators. GMT [12] concatenates the transformer directly behind GNN as a pooling process. GraphTrans [13] adds the transformer on top of the standard GNN layer and proposes

DOI: 10.54254/2755-2721/28/20230190

a new mechanism called Readout. Although these were able to fetch good experimental results, they only used the transformer idea and did not explore the commonality of GNN and transformer. The graph transformer model with superior performance captures the core of GNN and transformer and is more likely to achieve better results in the sequential recommendation task.

3.1.1. Feasibility. In this section, we focus on whether the graph transformer models can be applied to the sequential recommendation task. To get better performance in actual recommendation tasks, it is necessary to consider many other factors such as dynamic time variation, social network, and knowledge graph in the recommendation system. However, in this paper, only the basic network structure and methods for sequential prediction are discussed. Next, the feasibility of the graph transformer model will be explored from several specific model cases.

Graphormer [6] and GraphFormer [7] are the classical graph transformer models. And both demonstrate rich extensibility. The network structure of Graphormer and GraphFormer is shown in Figure 3 and Figure 4. Graphormer is using the topological properties of GNN which allows the transformer to focus not only on global information but also to capture local information by incorporating topological structure properties on the graph space in the transformer. Three main encoding methods are introduced in Graphormer, spatial encoding introduces information about the importance of nodes for the model; centrality encoding allows the model to measure the importance between nodes; Edge encoding is used to introduce the information on edges into the attention mechanism. GraphFormer, on the other hand, fully considers the neighbourhood information from GNN in the transformer encoding stage, i.e., cascading transformer and GNN. This structure allows a better fusion of local and global information in the text domain than the concat method.

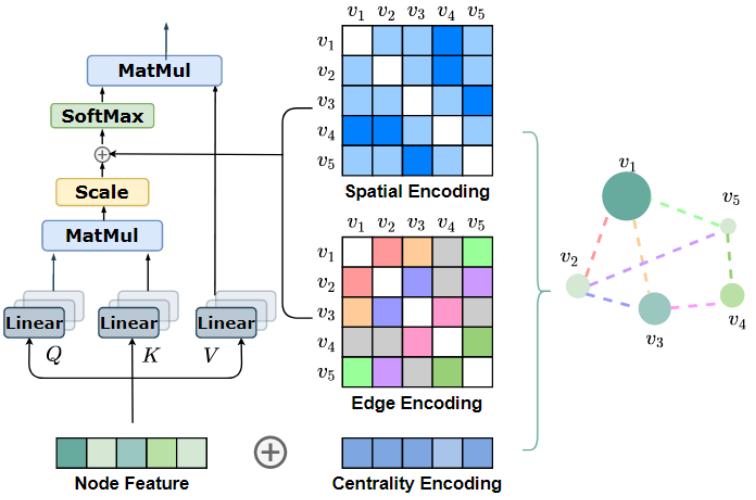

Figure 3. The model architecture of Graphormer is cited from [6].

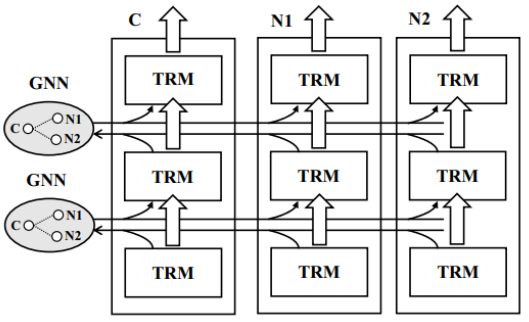

Figure 4. The model architecture of GraphFormer is cited from [7].

DOI: 10.54254/2755-2721/28/20230190

In most of the latest SeqRS methods, the sequences are modelled only by GNN. In TGSRec [4], the SR problem is linked to the graph embedding method. TGSRec aggregates item embeddings and temporal collaborative signals primarily through GNN. This enhances the Temporal Collaborative Transformer layer structure's representation of the item to enhance the model's performance. In SGRec [8], the article first generates graph-enhanced variants of the sequence data, then feeds them into the graph attention layer (like the GAT layer) to understand user preferences and POI representations. They both prefer to use the common GNN or GAT as a means of feature extraction, and then put it into the sequence prediction model for prediction.

The nature of graph transformers is the same, they add some tricks to make the whole feature extraction and project prediction tasks can be further nested to get better prediction results. From another perspective, TGSRec and SGRec in the SR task are essentially solving the problem of sequence prediction, which is the same purpose of the graph transformer, which combines the graph topology property of GNN and the transformer with an excellent ability to handle sequence data. Graphormer and GraphFormer were not originally designed for molecular property prediction scenarios or edge prediction tasks only, the introduction of similar encoding methods and structures may slightly improve the accuracy of predictions of existing SeqRS models.

3.1.2. Priority. The network design of many current models for sequential recommendation work is GNN or GAT, or GNN is only used as an auxiliary tool and thus its superior RL capability is ignored, so it is necessary to discuss the advanced graph transformer model.

Graph transformer is the combination of GNN and transformer. Graph transformer combines the core of transformer and GNN, both the attention mechanism of global attention and the consideration of topological properties of the graph. There are also many excellent models of graph transformers. For example, SAN [14] uses Learnable Position Encoding (LPE) in the transformer architecture, which can learn the position of each node in a given graph using the full Laplace spectrum. SAT [15] proposed a new self-attention mechanism, which can fuse structural information and significantly enhance the model's performance by using GNN to extract the subgraph representation. GraphGPS [9] has designed different Position Encoding (PE) for different categorizations. Also, GraphGPS solved the over-smoothing problem in the MPNNs layer and achieved excellent performance in various benchmark tasks.

In the context of sequential recommendation, long sequences usually reflect only implicit user behaviours and are somewhat noisy, making it difficult to accurately capture user preferences. Graph transformer has the advantage of capturing long-range dependencies compared to GNN. According to the introduction in 2.4, the attention to the global attention mechanism in transformer is that the attention factor is calculated between each word and all words, so the maximum path length is only 1, no matter how long the distance is. Furthermore, compared to ordinary transformers, GAT and graph transformers introduce topological structure properties of graphs on top of the transformer so that the model has a priori structural location in high-dimensional space. If simply using the transformer to process graph data, the structural information between nodes depends only on their semantic similarity, ignoring the structural prior simple graph transformer Layer [16] itself as shown in Figure 5.

DOI: 10.54254/2755-2721/28/20230190

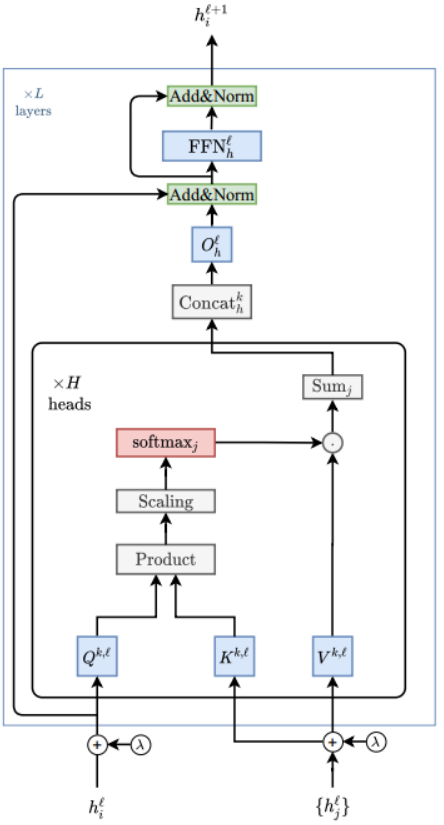

Figure 5. A simple graph transformer layer is cited from [16].

Graph transformer also has priority for GAT, which also uses the attention mechanism. GAT trained on graph datasets learns structured node information that does not vary with node position, so GAT is only local attention. Therefore, in comparison, graph transformer will have a greater ability to characterize and capture the complex relationships between nodes. Compared with GAT, graph transformer can handle the case of missing nodes and edges.

The ability to handle large-scale data is also a noteworthy issue in recommender systems. The vector of each node in GAT is passed through a multilayer perceptron, and the new vector of outputs is then used to calculate a weighted summation between every two nodes. Graph transformer employs Scaled Dot-Product Attention, which greatly improves the parallel processing capability of the model. Compared to GAT, graph transformer can handle large-scale graph data, including graph data with millions of nodes and edges, more efficiently. The advantages of graph transformer are better demonstrated in the Result in Section 4.

4. Results

This section compares mainly traditional graph neural networks (GCN and GAT) and popular graph transformers (Graphormer, SAN, GPS). To ensure the diversity of benchmarking tasks in the background of SR tasks, the experiment uses datasets from Benchmarking-GNNs [17] and Long-Range Graph Benchmark [18]. The experimental data in this section were obtained from [9].

4.1. Benchmarking-GNNs

From Benchmarking-GNNs, GNNs are tested on the ZINC, PATTERN, CLUSTER, MNIST and CIFAR10, shown in table 1. ZINC consists of 12K molecular graphs. These graphs have 28 different types of heavy atoms for each node, and three different types of bonds are represented by each edge. The goal is to regress the molecule's bound solubility. MNIST and CIFAR10 data were derived from a similarly named image classification dataset by constructing 8 nearest neighbour graphs of SLIC

DOI: 10.54254/2755-2721/28/20230190

superpixels for each image. The task is to classify the data. PATTERN and CLUSTER are made-up datasets that were drawn from a random block model. Unlike the previously mentioned tasks, this prediction task is based on inductive node-level classification.

The observation reveals that graph transformers give better SOTA results in five tasks. It is demonstrated that the graph transformer models perform better in a series of synthetic tasks created to evaluate the expressiveness of the model, compared to the network models used in the existing SR methods.

Table 1. Results of each model in five benchmarks (mean ± s.d. of performance in 10 runs) cited from [9].

Model	ZINC	MNIST	CIFAR10	PATTERN	CLUSTER
	MAE ↓	Accuracy ↑	Accuracy ↑	Accuracy ↑	Accuracy ↑
GCN	0.367 ± 0.011	90.705 ± 0.218	55.710 ± 0.381	71.892 ± 0.334	68.498 ± 0.976
GAT	0.384 ± 0.007	95.535 ± 0.205	64.223 ± 0.455	78.271 ± 0.186	70.587 ± 0.447
Graphormer	0.122 ± 0.006	–	–	–	–
SAN	0.139 ± 0.006	–	–	86.581 ± 0.037	76.691 ± 0.65
GPS	0.070 ± 0.004	98.051 ± 0.126	72.298 ± 0.356	86.685 ± 0.059	78.016 ± 0.180

4.2. Long-Range Graph Benchmark (LRGB)

The models listed above are assessed on LRGB. In the background of SR tasks, methods are expected to possess the capacity to capture long-range dependencies in graph data, which is LRGB and its datasets intended to test. This experiment set a 500k model parameter budget and followed the experimental setup which graph transformer used to test in LRGB. PascalVOC SP and COCO-SP are the products of SLIC superpixelation of Pascal VOC and MS COCO datasets. Each node in both datasets is a member of a distinct object class, making them both suitable for node classification tasks. PCQM-Contact is a derivative of PCQM4Mv2. As a link prediction task, the challenge requires detecting pairings of nodes that have structures with special 3D connections in the 2D graph but are distant. Data for both Peptide-func and Peptide-struct are derived from SATPdb, which is a database of structurally annotated therapeutic peptides. The task of Peptide-func is to classify the multi-labelled graphs into 10 non-exclusive peptide function classes. And the Peptide-struct task is to perform a graphical regression of all the peptide's various 3D structural features.

Table 2. Results of each model in LRGB (mean ± s.d. of performance in 4 runs) cited from [9].

Model	PascalVOC-SP	COCO-SP	Peptides-func	Peptides-struct	PCQM-Contact
	F1 score ↑	F1 score ↑	AP↑	MAE ↓	MRR ↑
GCN	0.1268 ± 0.0060	0.0841 ± 0.0010	0.5930 ± 0.0023	0.3496 ± 0.0013	0.3234 ± 0.0006
Transformer+LapPE	0.2694 ± 0.0098	0.2618 ± 0.0031	0.6326 ± 0.0126	0.2529 ± 0.0016	0.3174 ± 0.0020
SAN+LapPE	0.3230 ± 0.0039	0.2592 ± 0.0158	0.6384 ± 0.0121	0.2683 ± 0.0043	0.3350 ± 0.0003
GPS	0.3748 ± 0.0109	0.3412 ± 0.0044	0.6535 ± 0.0041	0.2500 ± 0.0005	0.3337 ± 0.0006

Only GCN, transformer, SAN, and GPS were evaluated in five LRGB datasets, shown in table 2. The experiment utilized LapPE positional encodings in both transformer and SAN and GPS with components. From the experimental results, graph transformers show better performance in five datasets in LRGB, which means they are equipped with a more powerful ability to capture long-range dependencies.

5. Conclusion

In this paper, we discuss the viability and superiority of popular graph transformers in the current sequential recommendation task, analysing it in terms of principle and architecture. This paper shows the performance of the graph transformers and the current models applied in the SR task in Benchmarking-GNNs and Long-Range Graph Benchmark for different datasets. In the ablation studies,

DOI: 10.54254/2755-2721/28/20230190

the effectiveness of graph transformers on various benchmarks further supports the point that graph transformers, or their variations will perform well in sequential recommendation tasks.

At present, graph transformer models with better performance are rarely used in sequential recommendation tasks. In the future, more graph transformer models for SR tasks will emerge to achieve more advanced prediction levels. Admittedly, the current graph transformer-based recommendation model has some limitations. The combination of GNN and transformer increases the model complexity to some extent, and the redundancy of the existing graph transformer structure has not been explored in detail. If these challenging tasks can be optimized in the future, this will have a non-negligible impact on the real-time and scalability of the graph transformer-based recommendation model. On the other hand, given the time-ordered developmental characteristics of recommendation systems, the dynamics of graph transformer-based recommendation models are insufficient. We can further investigate dynamic recommendation models to make them better at sequential recommendation tasks to accomplish dynamic user preference recommendations.

References

[1] Gao C, Zheng Y, Li N, Li Y, Qin Y, Piao J, Quan Y, Chang J, Jin D, He X and Li Y 2023 Graph neural networks for recommender systems: challenges, methods, and directions J. *TOIS.* 1-51.
[2] Wu S, Sun F, Zhang W, Xie X and Cui B 2022 Graph neural networks in recommender systems: a survey J. *CSUR.* 1–37.
[3] Chang J, Gao C, Zheng Y, Hui Y, Niu Y, Song Y, Jin D and Li Y 2021 Sequential recommendation with graph neural networks J. *ACM SIGIR.* 378–87.
[4] Fan Z, Liu Z, Zhang J, Xiong Y, Zheng L and Yu P 2021 Continuous-time sequential recommendation with temporal graph collaborative transformer J. *ACM CIKM.* 433-42.
[5] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A, Kaiser L and Polosukhin I 2017 Attention is all you need J. *NeurIPS.* 6000-10.
[6] Ying C, Cai T, Luo S, Zheng S, Ke G, He D, Shen Y and Liu T 2021 Do transformers really perform badly for graph representation J. *ACM CIKM.* 2735–43.
[7] Yang J, Liu Z, Xiao S, Li C, Sun G and Xie X 2021 GraphFormers: GNN-nested language models for linked text representation J. Preprint *arXiv*.2105.02605.
[8] Li Y, Chen T, Luo Y, Yin H and Huang Z 2021 Discovering collaborative signals for next POI recommendation with iterative seq2graph augmentation. J. *IJCAI.* 1491-97.
[9] Rampáek L, Galkin M, Dwivedi V, Luu A, Wolf G and Beaini D 2022 Recipe for a General, Powerful, Scalable Graph Transformer J. *NeurIPS.*
[10] Velikovi P, Cucurull G, Casanova A, Romero A, Liò P and Bengio Y 2018 Graph attention networks J. *ICLR.* 1–12.
[11] Nguyen D, Nguyen T and Phung D 2022 Universal graph transformer self-attention networks J. *WWW.* 193-96.
[12] Baek J, Kang M and Hwang S 2021 Accurate learning of graph representations with graph multiset pooling J. *ICLR.*
[13] Jain P, Wu Z, Wright MatthewA, Mirhoseini A, Gonzalez JosephE and Stoica I 2021 Representing long-range context for graph neural networks with global attention J. *NeurIPS.*
[14] Kreuzer D, Beaini D, Hamilton WilliamL, Létourneau V and Tossou P 2021 Rethinking graph transformers with spectral attention J. *NeurIPS.*
[15] Chen D, O'Bray L and Borgwardt K 2022 Structure-aware transformer for graph representation learning J. *ICML.* 3469-89.
[16] Dwivedi V and Bresson X 2020 A generalization of transformer networks to graphs J. Preprint *arXiv*.2012.09699.
[17] Dwivedi V, Joshi ChaitanyaK, Laurent T, Bengio Y and Bresson X 2020 Benchmarking graph Neural Networks J. Preprint *arXiv*:2003.00982.
[18] Dwivedi V, Rampáek L, Galkin M, Parviz A, Wolf G, Luu A and Beaini D 2022 Long range graph benchmark. J. *NeurIPS.*