Session-based Recommendation Algorithm Based on Heterogeneous Graph Transformer

Qiushi Wang, Wenyu Zhang*

Abstract—**Session-based recommendation is a technique that leverages the user's interaction sequence within a brief period to generate personalized recommendations. Currently, traditional transformer-based methods lack the capability to mitigate noise node interference in session sequences, while approaches based on heterogeneous graphs face challenges in capturing long-term sequential dependencies effectively. To tackle these challenges, a heterogeneous graph-enhanced Transformer session-based recommendation method is proposed in this paper. The proposed method constructs a heterogeneous session graph and utilizes a self-attention layer enhanced by the shortest path matrix. This enables the capturing of item embeddings that contains crucial graph structure information. Moreover, the method incorporates adversarial training to prevent excessive reliance on local graph information and achieve a more balanced integration of local and global information. After evaluating experiments on three public datasets, the results clearly demonstrate a remarkable enhancement compared to other robust baseline models, thus providing strong evidence of the method's effectiveness.**

Index Terms—**Adversarial Training, Heterogeneous Graph, Session-based Recommendation, Self -attention Mechanism, Transformer**

I. Introduction

THE recommendation system is an effective tool widely used in network platforms to address the challenge of "information overload" [1].In practical applications, recommendation systems rely on users' personal information to make recommendations. Nevertheless, there are numerous scenarios in which user identities remain unknown, and historical data is unavailable. Traditional recommendation methods typically leverage long-term historical interactions to learn user preferences, which are often unsuitable for such scenarios. In contrast, session-based recommendation methods utilize the users' short-term behavioral sequence information to generate recommendations that align with their current preferences, regardless of historical data. This approach also addresses the cold start problem encountered with new user recommendations. Moreover, session behavior sequences are usually heterogeneous, consisting of multiple behaviors. For instance, in a paper citation network, there are nodes representing authors and papers, with edges denoting author-author co-creation relationships and author-paper affiliations. Therefore, by leveraging heterogeneous graphs, session sequences can be represented in a manner that captures richer information, leading to enhanced accuracy in recommendations.

At present, heterogeneous session-based recommendation methods can be broadly categorized into two categories: methods based on recurrent neural networks (RNN) and those based on graph neural networks (GNN). RNN-based methods are generally used to process sequence data and model user interests based on users' click sequence. However, in real-world scenarios, clicking sequences often contain noisy information, limiting model's ability to capture all correct dependencies accurately. Furthermore, as the session sequence growing longer, the performance of RNN-based methods drops significantly, making it diffcult to learn dependencies between items that are distant from each other. On the other hand, GNN-based methods exhibit good performance in extracting user's short-term intentions. By utilizing the underlying graph structure, these methods can effectively capture the intricate transfer relationships between items. However, these methods also struggle to learn dependencies between items that are far apart. They primarily aggregate closely connected items in the session graph, resulting in limited perception of the user's long-term interests and potential information loss when the session sequence is relatively long. Recently, the Transformer model, proposed by Vaswani et al. [2], has exhibited impressive performance in modeling long sequences by making full use of the attention mechanism. Some researchers have applied it to the field of session-based recommendations. While the Transformer has shown promising results, it is vulnerable to generate noise in session-based recommendations. The presence of noise in the users' click sequence can lead to inaccurate capturing of user interests.

To address the aforementioned issues, Graph-Enhanced Transformer model (GETrm) is proposed in this paper. Firstly, it constructs the session sequence as a heterogeneous graph, leveraging the inherent properties of the graph to enable local noise control. This prevents the noise propagation and accumulation, particularly in long sequences. Secondly, it enhances the self-attention layer in the Transformer by utilizing the shortest path matrix between each node in the heterogeneous graph. Specifically, the weight matrix corresponding to the query (Q), key (K), and value (V) in the self-attention mechanism is initialized. It is initialized using the shortest path matrix of the graph to integrate the structural information between each node in the heterogeneous graph. This allows the self-attention mechanism to model long sequences effectively while fusing

Manuscript received March 26, 2023; revised September 4, 2023.

Qiushi Wang is a postgraduate student of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, 114051, China. (e-mail: 614918334@qq.com).

Wenyu Zhang is a Professor of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, 114051, China. (corresponding author to provide phone: 131-8800-8518; e-mail: zhangwenyu8518@126.com).

graph information, thereby addressing the problem of global information loss which can occur when modeling long sequences solely based on graph structure. To prevent the model from becoming overconfident and over-reliant on heterogeneous graph structure information, an anti-noise structure is added to the graph-enhanced self-attention layer to enhance the model's generalization ability.

II. Related work

A. Heterogeneous Graph-based Methods

Heterogeneous graphs, consisting of multiple node types or multiple types of connections, facilitate the extraction of more comprehensive information from diverse behavior sequences. This, in turn, enhances recommendation accuracy by capturing a broader range of user preferences and interactions. Unlike classical recommendation approaches, heterogeneous graphs can leverage diverse information for recommendation, resulting in better performance for new users and new products. Moreover, different types of connections require different weights or processing methods. Meng et al. proposed the Micro-behaviors and item Knowledge into Multi-task learning for Session-based Recommendation (MKM-SR) model to capture complex item transitions in item sequences by assuming sequential dependencies among different behavior types [3]. Xu et al. proposed the Heterogeneous Graph Transformer (HGT) model, using different edge types in heterogeneous graphs to transfer information and enhance the representation ability of item embeddings [4]. Pang et al. proposed the Heterogeneous Global Graph Neural Network (HG-GNN) model to learn long-term user preferences and item representations with rich semantics [5]. Wang et al. proposed the Heterogeneous Graph Attention Network (HAN) model, which considerd node-level and semantic-level attention mechanisms to capture complex structure and rich semantic information in heterogeneous graphs [6]. However, these methods only aggregate the information of adjacent nodes in the graph, which can result in information loss for long sequences.

B. Transformer-based Methods

The Transformer model has emerged as a cutting-edge technology in the field of natural language processing and serves as the foundation for current pre-trained language models (PLMs) and large-scale language models (LLMs). The Transformer's ability to efficiently train in parallel makes it scalable to handle large amounts of training data and model size. Unlike traditional RNN structures, Transformers excel at capturing long-range dependencies in sequences by employing self-attention mechanisms. Several studies have applied the Transformer model to session-based recommendation tasks. For example, Wang et al. proposed the Self-attentive Sequential Recommendation (SASRec) model that drew on the Transformer structure to model a user's historical behavior information and extract more valuable information [7]. Similarly, Chen et al. proposed the Behavior Sequence Transformer (BST) model that used the Transformer to capture associated features in user behavior for recommendation [8]. Additionally, Yang et al. proposed the Weighted Graph Interest Network (WGIN) model, which integrated data processed by the Repetitive Weighted Graph Neural Network (RWGNN) model into the Transformer to improve its computing power using parallelism [9]. However, Transformer-based models model sequences as fully connected graphs, which can destroy the sequential structure of sessions and introduce noise when aggregating neighbor features.

III. Method

A. Preliminaries

The purpose of the session-based recommendation is to predict the next click item that an anonymous user is most likely to click, based on his behavior during the current session. Given the anonymous nature of the user and the lack of available information, this task poses a greater challenge compared to conventional recommendation approaches. Let $V=\{v_1,v_2,\cdots,v_n\}$, denote the set of all n items involved in all sessions. Each anonymous session is defined as $S=\{v_1^S,v_2^S,\cdots,v_n^S\}$, where $v_i^s\in V$ is the item clicked by the user during session S in step i, and n is the length of session S. The objective of the model is to predict the next item V_{n+1} of the session, which the user is most likely to click, based on the historical click sequence. In this paper, the user's interest representation is modeled based on the proposed neural network, according to the anonymous session S, and the probability $\hat{y}$ of the user's next clicking on an item is calculated. Finally, the model recommends the user the top K items with the highest probabilities.

B. Architecture

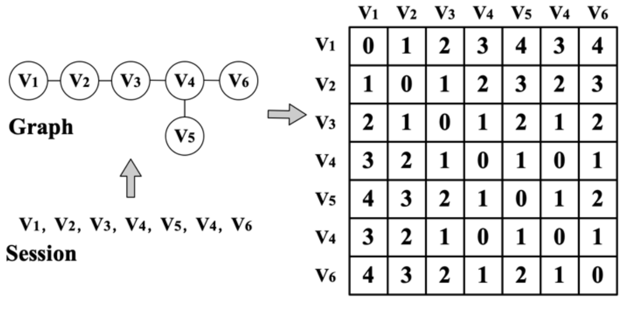
The GETrm model mainly consists of an embedding layer, a graph-enhanced self-attention layer, and a recommendation layer. Firstly, the session sequence is transformed into a heterogeneous session graph. The embedding layer converts the nodes in the graph into embedding vectors, and the resulting matrix, which combines Item Embedding and Position Embedding, are used as inputs for the model. Additionally, the shortest path matrix is obtained based on the shortest distance between different nodes in the session graph. The graph enhanced self-attention layer assigns different weights to the parameters of Q, K, and V according to the shortest path matrix. Finally, the recommendation layer sorts the candidate items based on the user interest vector and recommends the *Top-K* items to the user. Figure 1 shows the workflow of the GETrm model.

Building Session Heterogeneous Graph

To capture the complex transfer more effectively relationships between items across sessions, the initial step of the model involves transforming the session sequence into a session graph. This conversion is achieved by transforming the original training sequence $(S_1,S_2,\cdots,S_n)$ in the dataset into a fixed-length sequence $k=(k_1,k_2,\cdots,k_n)$, where n represents the maximum length that the model can handle. If the sequence length exceeds n, only the most recent n actions will be considered. Conversely, if the sequence length is less than n, a padding item of 0 is repeatedly appended to the left of the sequence until the length becomes n.

Subsequently, in order to calculate the attention network, the model obtains the shortest path distance between items within a session, each session sequence is constructed as an undirected session graph $G_s = (V_h, V_r)$, where the starting node $V_h = \{V_1, V_2, \cdots, V_n\}$ and the final node $V_r = \{V_1, V_2, \cdots, V_m\}$ correspond to the set of items that appear in the session sequence. The edges in the graph G_s represent two adjacent items $\langle V_h, V_r \rangle$, indicating that the user clicks on item V_r after item V_h. To ensure considerations of the shortest path in an undirected graph, the reverse edge $\langle V_r, V_h \rangle$ is also regarded as an edge in the graph.

Finally, the Floyd algorithm is used to calculate the shortest path distance between any two nodes in the graph, and then the shortest path matrix is then obtained. Specifically, a session graph is constructed based on the session sequence, and the Floyd algorithm is used to calculate the shortest path matrix on the graph, as illustrated in Figure 2. Incorporating this information into the modeling of item representations allows for a better capture of the relationships between items.



	V1	V2	V3	V4	V5	V4	V6
V1	0	1	2	3	4	3	4
V2	1	0	1	2	3	2	3
V3	2	1	0	1	2	1	2
V4	3	2	1	0	1	0	1
V5	4	3	2	1	0	1	2
V4	3	2	1	0	1	0	1
V6	4	3	2	1	2	1	0

(a) Session Heterogeneous Graph **(b) the Shortest Path Matrix Caculation**

Fig. 2. Illustration of the construction of session heterogeneous graph and the calculation process of the shortest path matrix

Embedding Layer

Firstly, an item embedding matrix $M \in R^{|I| \times d}$ is created, where d is the dimension of the latent space. The input embedding matrix $E \in R^{n \times d}$ is then retrieved, where $E_i = M_{ki}$. Since the self-attention module in Transformer does not include any loop or convolution modules, the position information of the input sequence cannot be obtained. Therefore, a learnable positional encoding $P \in R^{n \times d}$ is injected into the input embedding layer, where P is updated during iterations:

$$\hat{E} = \begin{bmatrix} M_{k1} & + & P_1 \\ M_{k2} & + & P_2 \\ & \cdots & \\ M_{kn} & & P_n \end{bmatrix} \tag{1}$$

Graph Enhanced Self-Attention Layer

The graph-enhanced self-attention layer leverages the shortest path distance matrix to calculate self-attention, which enables the proposed method to model the complex transformational relationships between items. Unlike the conventional self-attention layer, the graph-enhanced self-attention layer uses the graph structure to distinguish explicitly the relationship between items with different shortest path distances. This enables the layer to compute adaptive weights that capture the varying importance of different relationships within the graph structure.

Specifically, the graph-enhanced self-attention layer converts the embedding layer input into three different matrices (Q, K, and V) through linear projection and inputs them into the self-attention layer using scaling dot product. Unlike the self-attention network, different Q, K, and V mapping matrices are allocated for item embedding based on the shortest path distance of different nodes in the session graph, and then the graph-enhanced self-attention is calculated through the Q, K, and V values. Specifically, different Q, K, and V are used between nodes at various distances in the graph to obtain the item's context representation matrix. To enhance the model's generalization capability, an anti-noise mechanism is introduced in the self-attention layer of graph enhancement. This approach aims to mitigate the model's excessive dependence on heterogeneous graph structure information, thereby addressing the concern of global information loss caused by an overemphasis on local information. In this process, adversarial noise is added to the shortest path matrix, and the Projected Gradient Descent (PGD) algorithm [10] is used for adversarial training to enhance the generalization of the model. As a result, the model will not only focus on the local structure of the heterogeneous graph but also give increased consideration to the global characteristics of the learning graph. The formula is as follows:

$$Q = (W^q_{distance} + \varepsilon)x + b^q_{distance} \tag{2}$$

$$K = (W^k_{distance} + \varepsilon)x + b^q_{distance} \tag{3}$$

$$V = (W^v_{distance} + \varepsilon)x + b^v_{distance} \tag{4}$$

$$\beta = \frac{Q}{\sqrt{d}} K \tag{5}$$

where $W_{distance}$ represents the shortest path matrix, ε is the anti-noise, b is the bias, and x is the model input. Besides, $\sqrt{d}$ is introduced to avoid excessive values of the inner product when the dimension is too high and to ensure that the attention score β satisfies the normal distribution. By using different Q, K, and V between nodes with different distances, the influence of node distance can be considered in the attention calculation process. Therefore, the graph-enhanced self-attention not only retains the advantages of the sequence model, which can model the user's sequence interest, but also incorporates the benefits of the graph model, which can express the complex transfer relationship between items. The resulting attention scores are then normalized using the SoftMax function, expressed as:

$$a = softmax(\beta) \tag{6}$$

Finally, using attention to get a new representation vector for each item x_i:

$$x_i = \sum_{i=1}^{n} a_i V_i \qquad (7)$$

where i represents the i-th item interacted with by the user.

Although graph-augmented attention can capture session-global information, its aggregation of session-global sequential information is still a linear model. To introduce nonlinearity and account for the interaction between different latent dimensions, this paper introduces a feedforward neural network that applies two linear transformations and extracts nonlinear features through an intermediate ReLU activation function. To propagate low-level features to higher-level representations, improve the capture of global information, and mitigate overfitting issues in deep neural networks, residual networks and Dropout regularization techniques are employed.

$$FFN = max(0, xW_1 + b_1)W_2 + b_2 \qquad (8)$$

where $W_1, W_2 \in R^{d \times d}$; b_1, b_2 represent bias vectors.

Recommendation Layer

Within a session, each user click significantly impacts his subsequent clicks. In this study, the representation vectors of all items in the session are averaged to obtain an average vector that captures the overall of the entire session. Since the average vector within a session is used to represent the session, it can be converted into a fixed-length vector for subsequent processing. Therefore, the average vector within the session is used to consider the user's holistic interest, and it is then combined with the user's immediate interest to generate a session-level embedding vector for recommendation. Equation (9) obtains session-level vectors through attention f_i, as follows:

$$f_i = q^T sigmiod(w_0 x_{last} + w_1 x_{avg} + w_2 x_i + b) \qquad (9)$$

where x_{last} represents the last item vector, x_{avg} represents the average vector in the whole session; w , q is the weight matrix; b is the bias. The weighted sum S of all vectors is used as the user interest representation, and its formula is (10).

$$S = \sum_{i=1}^{n} f_i x_i \qquad (10)$$

Finally, the inner product of the user interest representation and the candidate item vector is used as the recommendation score for each item y_i. By using the SoftMax function, the recommendation scores of all candidate items are converted into the probability of the next click:

$$y_i = S^T e_i \qquad (11)$$

$$\hat{y}_i = softmax(y_i) \qquad (12)$$

where $\hat{y}_i$ denotes e_i, which is the probability of the item appearing in the next click in session S. The top k items with the highest probabilities in the $\hat{y}_i$ list will be recommended.

In this paper, the cross-entropy loss function is used as the training objective. The objective function is as follows:

$$L = -\sum_{i=1}^{|V|} y_i log(\hat{y}_i) log(1 - \hat{y}_i) \qquad (13)$$

where y_i is the one-hot vector of ground-truth items for the autoencoder. By minimizing the objective function, the model can generate effective recommendation lists.

IV. Experimental

A. Datasets

This paper verifies the performance of the GETrm model on three real datasets: Diginetica [11], Tmall [12], and RetailRocket [13]. The Diginetica dataset came from the 2016 CIKM Cup, where only its transactional data is used. The Tmall dataset came from the IJCAI-15 competition, which contains anonymous users shopping on the Tmall online shopping platform. The RetailRocket dataset was from a real-world e-commerce website, which published the dataset with six months of user browsing activities. In this paper, the three datasets are preprocessed according to [14], where sessions of length 1 and items that occur less than 5 times in the dataset are filtered out. The preprocessed data are shown in Table 1.

TABLE I
Statistics Of The Dataset

Dataset	Tmall	RetailRocket	Diginetica
#clicks	2643331	4005557	3218949
#trains	2272872	3892416	2973403
#tests	370459	113141	245546
#items	40728	36969	43098

B. Experimental Settings

This experiment uses Adam as the optimizer with an initial learning rate of 1e-3 and a linear schedule decay rate of 0.1 every three epochs. During training, L2 regularization is set to 1e-5 to avoid overfitting. All means are initialized to 0, and the standard deviation is a normal distribution of 0.05. We will report the results of the model under the fair setting and the optimal hyperparameter setting.

C. Evaluation Metrics

In the experiments, to maintain the same setting as the previous baseline model, we also choose to use the top 10 and top 20 items to evaluate the recommender system. This paper adopts two ranking-based metrics: $P@K$ and $M@K$, to verify the effectiveness.

$P@K$: $P@K$ is widely used as a measure of forecast accuracy. $P@K$ represents the proportion of correct recommended items in the top K positions of the recommendation list provided to the user. In this paper, $P@10$ and $P@20$ are used for all tests.

$$P@K = \frac{n_{hit}}{N} \qquad (14)$$

where N represents the total number of test sample data in the recommender system and n_{hit} represents the number of predicted accurate items in the top K ranked lists.

$M@K$: The average bottom rank is the average of the average bottom ranks of the correctly recommended items, and when the rank exceeds K , the bottom rank is set to 0. A large $M@K$ value indicates that the correct recommendation is at the top of the ranking list.

$$M@K = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{p_i} \tag{15}$$

where N represents the total number of users; p_i represents the position of the i-th user's real access value in the recommendation list, if the value does not exist in the recommendation list, then $p_i \to \infty$.

D. Baseline Methods

To evaluate the performance of the GETrm model, this paper compares it with nine other representative methods:

Item-KNN [15]: This method uses cosine similarity to calculate the similarity score between the user's last interactive item and candidate items in the session and recommends the top N items with high similarity scores.

FPMC [16]: This method uses a combination of first-order Markov chain and matrix factorization to predict the sequence of sessions and recommend the user's next click.

GRU4Rec [17]: This approach uses RNN models for session-based recommender system tasks and utilizes Gated Recurrent Units (GRUs) to extract sequence information.

NARM [18]: This method applies the RNN model to extract sequence information and adds an attention mechanism to capture the user's main purpose for recommendation.

STAMP [19]: This method employs attention layers to replace all RNN encoders in previous work and uses a self-attention mechanism to improve session-based recommendation performance.

SR-GNN [14]: This method applies a gated neural network to capture complex transitions of items for session-based recommendation.

GC-SAN [20]: This method applies a self-attention network to learn global and local dependency information between items in a session to make recommendations for users.

S^2-DHCN [21]: This method constructs two kinds of hypergraphs to learn inter-session information and intra-session information and utilizes self-supervised learning to enhance session-based recommendation.

GCE-GNN [22]: This method constructs two kinds of session derivation graphs, capturing different levels of local and global information.

E. Experimental Results and Analysis

Comparison With Baseline Methods

To verify the performance of the proposed method in this paper, GETrm and nine other baseline models are compared on three datasets using four evaluation indicators ($P@10$, $M@10$, $P@20$ and $M@20$). Compared to the baseline model, GETrm outperformed Gobal Context Enhanced Graph Neural Network (GCE-GNN) by an average of 11% on the Tmall dataset, 8.6% on the RetailRocket dataset, and 2.0% on the Diginetica dataset. From Table 2, it can be observed that GETrm achieves the best performance on all four metrics on all three datasets, thus confirming the effectiveness of the proposed method.

The conventional Item K-Nearest Neighbor (Item-KNN) algorithm recommends only based on the last item of the user's historical interaction sequence, which is a limitation in the session-based recommendation scenario. In contrast, Factorizing Personalized Markov Chains (FPMC) utilizes first-order Markov chains and matrix factorization to obtain more inter-session information to a certain extent. Nevertheless, Item-KNN outperforms FPMC on the Tmall and RetailRocket datasets, indicating that the assumption that traditional Markov chain-based methods heavily rely on the independence of consecutive items is unrealistic.

In session recommendation based on deep learning [23], it is observed that this category of methods generally exhibits superior performance compared to the previous two categories. For instance, GRU4Rec despite its poor performance on the Tmall and Diginetica datasets, still demonstrates the effectiveness of RNNs in sequence modeling. Neuro Affective Relational Model (NARM) not only uses RNN to model the conversation sequence, but also integrates the attention mechanism to capture the information in the main conversation sequence. The indicators on the three datasets are significantly improved compared to Gated Recurrent Unit for Recommendation (GRU4Rec). This outcome emphasizes the significance of key session sequence information in recommendation tasks.

GNN-based methods outperform other approaches in most cases. Session-based Recommendation with Graph Neural Networks (SR-GNN) first proposes to use graph structure data for session-based recommendation by using GNN to capture the rich information of the current session node. Graph Contextualized Self-Attention Network (GC-SAN) combines GNN with the self-attention mechanism leveraging the complementary strengths of both approaches to improve the performance of the recommendation system. Self-Supervised Multi-Channel Hypergraph Convolutional Network (S^2-DHCN), on the other, leverages both inter-session and intra-session information in hypergraph modeling and achieves good performance. GCE-GNN achieves higher accuracy than other baseline models by utilizing cross-session information. It learns two levels of item embeddings from session-level and global-level, respectively, and then combines them via element-level summation. All the aforementioned methods use graph structures to represent the information within the session sequence, which enriches the connections between sessions and improves the accuracy of recommendations to a significant extent.

The GETrm method outperforms almost across all baselines on all datasets. Notably, it demonstrates a significant advantage on the Tmall dataset, highlighting the effectiveness of the heterogeneous graph Transformer structure when applied to actual e-commerce data. Using the graph structure can also mitigate local noise and exhibit clear advantages over other graph-structured models. The information in the session sequence is aggregated into the heterogeneous session graph, and the impact of the session's global information and other factors on the user preferences is considered to make more effective recommendations.

It can be clearly seen from Table 2 that the proposed GETrm model achieves better performance on the four indicators under the three datasets.

Ablation Experiments

To analyze the impact of the shortest path matrix and adversarial training on the recommendation effect, this section conducts ablation experiments: without Shortest Path Routing (w/o SPR) removing the shortest path matrix from GETrm and without Adversarial Training (w/o AT) removing adversarial training from the middle. The experimental results are shown in Table 3.

Without Shortest Path Routing (w/o SPR): As shown in Table 3, GETrm achieved better results on the three datasetsthan the method of removing the shortest path matrix. This outcome demonstrates that using the shortest path distance between nodes in the graph can enable self-attention to better perceive the relationship between items in the graph, resulting in better modeling of user interests.

Without Adversarial Training (w/o AT): As shown in Table 3, GETrm achieved better results on the three datasets than the method of removing adversarial training. This result indicates that adding noise to the graph-augmented self-attention layer can effectively improve the robustness and generalization of the model, leading to improved recommendation accuracy.

V. Conclusion

This paper focuses on addressing the challenging task of session-based recommendation that the user's information is anonymized, and the user's historical behavior data is unavailable. Previous methods cause information loss and noise issues. Therefore, this paper proposes a session-based recommendation algorithm based on a heterogeneous graph Transformer. The algorithm converts the session sequence into an undirected session heterogeneous graph and considers the session context information to obtain richer target session information. Additionally, location information is added for each node in the graph, preserving the original access information. To enhance the self-attention layer in the Transformer, the paper introduces the use of shortest path matrices between nodes in the heterogeneous graph. This integration fights against noise and enhances the model's generalization capabilities. Experimental results on three publicly available datasets demonstrate the significant superiority of the proposed method over other algorithms.

References

[1] C. Zhang, W. Zheng, Q. Liu, J. Nie, and H. Zhang, “SEDGN: Sequence enhanced denoising graph neural network for session-based recommendation,” *Expert Systems with Applications*, vol. 203, pp. 117391, 2022.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez,L. Kaiser, and I. Polosukhin, "Attention is all you need," *2017 the 31st Conference on Neural Information Processing Systems (NIPS)*, vol. 30, pp. 1-11, 2017.

[3] W. Meng, D. Yang, and Y. Xiao, "Incorporating user micro-behaviors and item knowledge into multi-task learning for session-based recommendation," *2020 the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1091-1100, 2020.

[4] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," *2020 the 29th Web of Conference (WWW)*, pp. 2704-2710, 2020.

[5] Y. Pang, L. Wu, Q. Shen, Y. Zhang, Z. Wei, F. Xu, E. Chang, B. Long, and J. Pei, "Heterogeneous global graph neural networks for personalized session-based recommendation," *2022 the 15th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 775-783, 2022.

[6] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," *2019 the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2022-2032, 2019.

[7] W.-C. Kang, and J. McAuley, "Self-attentive sequential recommendation," *2019 the 27th ACM International Conference on Information and Knowledge Management*, pp. 197-206, 2019.

[8] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou, "Behavior sequence transformer for e-commerce recommendation in Alibaba," *2019 the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data with KDD (DLP KDD)*, pp. 1-4, 2019.

[9] Z. Yang, H. Wang, and M. Zhang, " WGIN: A Session-Based Recommendation Model Considering the Repeated Link Effect," *IEEE Access*, vol. 8, pp. 216104-216115, 2020.

[10] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *2018 the 6th International Conference on Learning Representations (ICLR)*, 2018.

[11] P. Kumar, and R. S. Thakur, "Recommendation system techniques and related issues: a survey, " International Journal of Information Technology, vol. 10, pp. 495-501, 2018.

[12] Z. Li, H. Zhao, Q. Liu, Z. Huang, T. Mei, and E. Chen, "Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors," *2018 the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1734-1743, 2018.

[13] G. Bonnin, and D. Jannach, "Automated generation of music playlists: Survey and experiments," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, pp. 1-35, 2014.

[14] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," 2*019 the 33rd AAAI Conference on Artificial Intelligence*, pp. 346-353, 2019.

[15] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," *2001 the 10th International Conference on World Wide Web (WWW)*, pp. 285-295, 2001.

[16] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," *2010 the 19th International Conference on World Wide Web (WWW)*, pp. 811-820, 2010.

[17] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks, " *2016 the International Conference on Learning Representations (ICLR)*, 2016.

[18] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," *2017 the 17th ACM Conference on Information and Knowledge Management (CIKM)*, pp. 1419-1428, 2017.

[19] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: short-term attention/memory priority model for session-based recommendation," *2018 the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1831-1839, 2018.

[20] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph Contextualized Self-Attention Network for Session-based Recommendation," *2019 the 28th International Joint Conference on Artificial Intelligence*, pp. 3940-3946, 2019.

[21] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang, "Self-supervised hypergraph convolutional networks for session-based recommendation," *2021 the 35th AAAI Conference on Artificial Intelligence*, pp. 4503-4511, 2021.

[22] Z. Wang, W. Wei, G. Cong, X.-L. Li, X.-L. Mao, and M. Qiu, "Global context enhanced graph neural networks for session-based recommendation," *2020 the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 169-178, 2020.

[23] Jianfeng Zhang, Tianwei Shi, Wenhua Cui, Ye Tao, and Huan Zhang, "Cross-Dimensional Feature Fusion MLP Model for Human Behavior Recognition," *Engineering Letters*, vol. 30, no.4, pp.1457-1464, 2022.

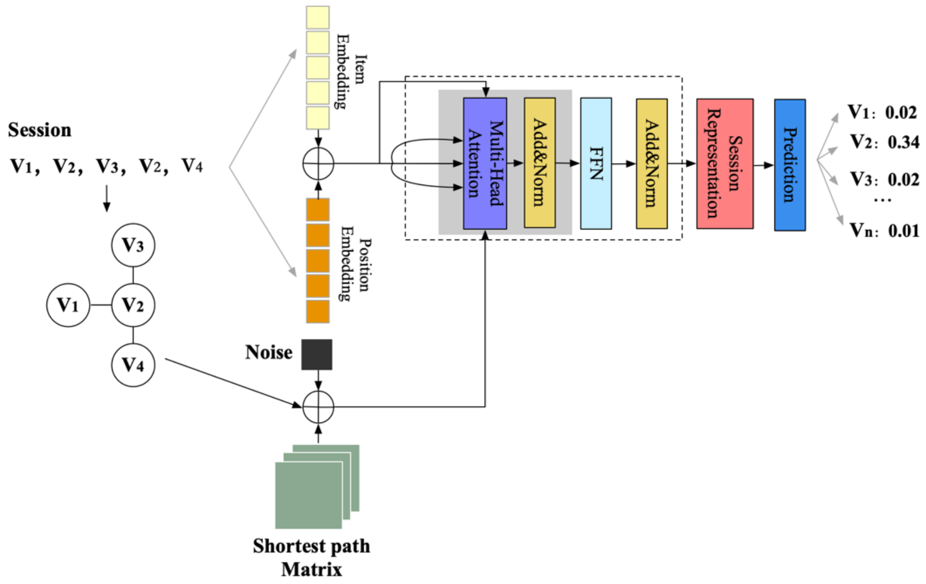


Fig. 1. GETrm model

TABLE II
Performance Comparison Of GETrm And Other BaselineAodels

Methods		Item-KNN	FPMC	GRU4Rec	NARM	STAMP	SR-GNN	GC-SAN	S^2-DHCN	GCE-GNN	GETrm
Tmall	**P@10**	6.65	13.10	9.47	19.17	22.63	23.41	23.92	26.22	28.01	30.27
	M@10	3.11	7.12	5.78	10.42	13.12	13.45	13.64	14.60	15.08	17.32
	P@20	9.15	16.60	10.93	23.30	26.47	27.57	27.89	31.42	33.42	35.11
	M@20	3.31	7.32	5.89	10.70	13.36	13.72	13.86	15.05	15.42	17.88
Retail-Rocket	**P@10**	21.41	25.99	38.35	42.07	42.95	43.21	44.10	46.15	47.06	48.02
	M@10	9.78	13.38	23.27	24.88	24.61	26.07	26.92	26.85	27.24	28.65
	P@20	17.25	32.37	44.01	50.22	50.96	50.32	51.18	53.66	54 .34	55.77
	M@20	5.56	13.82	23.67	24.59	25.17	26.57	27.40	27.30	27.60	29.18
Diginet-ica	**P@10**	25.07	15.43	17.93	35.44	33.98	38.42	36.21	40.21	41.16	41.48
	M@10	10.77	6.30	7.59	15.25	14.26	16.89	15.83	17.59	18.15	18.20
	P@20	28.35	26.53	29.45	49.70	45.64	50.73	50.91	53.66	54.22	54.29
	M@20	9.45	6.59	8.33	16.17	14.32	17.59	17.18	18.51	19.04	19.06

TABLE III
Experimental Analysis Of The Impact Of Different Variants On The Model

Dataset	Methods	GETrm	w/o SPR	w/o AT
Tmall	**P@10**	30.27	26.77	28.57
	M@10	17.32	15.12	15.17
	P@20	35.11	31.77	33.37
	M@20	17.88	15.40	15.50
RetailRocket	**P@10**	48.02	46.70	47.08
	M@10	28.65	28.20	28.39
	P@20	55.77	53.86	54.36
	M@20	29.18	28.78	28.90
Diginetica	**P@10**	40.48	38.50	39.10
	M@10	18.20	16.95	17.13
	P@20	54.19	51.27	51.92
	M@20	19.06	17.83	18.02