Acadlore Transactions on AI and Machine Learning
https://www.acadlore.com/journals/ATAIML

Enhancing Session-Based Recommendations with Popularity-Aware Graph Neural Networks

Qingbo Sun[1], Weihua Yuan[2], Qi Zhang[1], Zhijun Zhang[2*]

[1] School of Cyberspace Security, Shandong University of Political Science and Law, 250014 Jinan, China
[2] School of Computer Science and Technology, Shandong Jianzhu University, 250101 Jinan, China

[*] Correspondence: Zhijun Zhang (zhangzj@sdjzu.edu.cn)

Received: 07-23-2022 **Revised:** 08-21-2022 **Accepted:** 09-20-2022

Citation: Q. B. Sun, W. H. Yuan, Q. Zhang, and Z. J. Zhang, "Enhancing session-based recommendations with popularity-aware graph neural networks," *Acadlore Trans. Mach. Learn.*, vol. 1, no. 1, pp. 22-29, 2022. https://doi.org/10.56578/ataiml010104.

Abstract: Real-time and reliable recommendations are essential for anonymous users in session-based recommendation systems. Graph neural network-based algorithms are attracting more researchers due to their simplicity and efficiency. However, current methods overlook the influence of edge frequency on feature aggregation in graph modeling and fail to account for the impact of item popularity on user interest. To address these issues, a novel approach called Popularity-Aware Graph Neural Networks for Session-based Recommendations is proposed. This study integrates both edge frequency and item popularity into the modeling process to enhance the learning of item features and user interests. A graph that includes the number of edge occurrences is constructed, and a graph neural network with an attention mechanism is utilized to learn user interests and item features by aggregating information from the graph. Finally, the session's final representation is learned based on the occurrence frequency of items. The proposed study evaluates the model on two classical e-commerce datasets and demonstrates its superiority over existing methods.

Keywords: Session-based recommendation; Graph neural networks; Popularity-aware modeling; Item popularity; User interests

1. Introduction

In the fast-paced digital economy era, finding useful information from massive and complex data is crucial. Recommendation systems have emerged as effective tools to overcome information overload and are widely used in online platforms. Traditional recommendation systems use collaborative filtering to mine users' interests from their historical behaviors and then predict the products they might like. However, long-term historical behaviors may overshadow users' current interests during a session, resulting in poor user experience. To address this issue, session-based recommendation systems have been developed, which aim to predict users' behavior during the current session. Session-based recommendation systems have been widely adopted by various e-commerce platforms, such as T-mall and Amazon.

In the field of recommendation systems, there are two main categories of traditional session-based recommendation algorithms: collaborative filtering and Markov chain-based algorithms. Collaborative filtering [1] typically utilizes user or item similarity to make recommendations. In contrast, the Markov chain-based recommendation algorithm [2] treats session-based recommendation as a sequence prediction task and predicts the next item based on the sequence relationship between items. However, it is important to note that the Markov chain-based approach faces the challenge of dimension explosion, which makes it difficult to apply in actual production scenarios. This limitation arises because the number of possible item combinations can increase exponentially as the sequence length grows, leading to a computational explosion. As a result, it becomes impractical to compute the probabilities for all possible combinations of items.

With the development of deep learning technology, many researchers have applied neural network technology in recommendation systems. Several session-based recommendation models based on deep learning have been proposed, such as GRU4REC [3], Tan et al. [4], NARM [5], STAMP [6], SR-GNN [7], HA-GNN [8], TPA-GNN

[9], GCE-GNN [10], DIDN [11], and CORE [12]. These models have achieved good performance in recommendations; however, they have some limitations that can be improved.

GRU4REC uses multi-layer GRU to model the whole session and capture the continuous preferences of users. However, it only captures the session relationship between one-way adjacent items and may not capture long-term dependencies between items.

NARM combines the attention mechanism with RNN to capture users' interests. However, it also suffers from the limitation of capturing long-term dependencies between items.

STAMP captures users' long-term interests by using the multi-layer perceptron and the attention mechanism, and combines users' short-term preferences for recommendations. However, it still lacks the ability to capture the influence of popularity on user interest.

SR-GNN uses graph neural networks to learn the transformation relationship between items and capture the high-order relationship in graphs. However, it still uses the attention mechanism to filter out noise and may not effectively capture the influence of popularity on user interest.

HA-GNN captures the high-order relationship in graphs by using the attention mechanism. However, it does not explicitly consider the influence of popularity on user interest.

TPA-GNN combines time series with graph neural networks to capture users' interests. However, it still lacks the ability to effectively capture the influence of popularity on user interest.

GCE-GNN uses the features of the session graph and the global graph to learn items. However, it also suffers from the limitation of capturing long-term dependencies between items.

DIDN incorporates dynamic intent-aware and iterative denoising modules to learn dynamic item embeddings and filter out noisy clicks within sessions. However, it still lacks the ability to capture the influence of popularity on user interest.

CORE unifies the representation space for both the encoding and decoding processes in session-based recommendation to address the issue of inconsistent predictions. However, it also does not explicitly consider the influence of popularity on user interest.

To address the aforementioned issues, we propose a novel model called the Popularity-Aware Machine for Session-Based Recommendations (PASR). Figure 1 illustrates the workflow of the proposed PASR method. Firstly, PASR constructs a popularity-aware item graph, which effectively captures users' preferences for popular items. Secondly, PASR aggregates the features of neighboring nodes based on the type and frequency of edges in the graph, which enhances the model's ability to capture the dependencies between items. Finally, popularity embeddings are integrated into the attention mechanism to learn users' interests and improve the accuracy of recommendations. The main contributions of this work are as follows:

(1) The PASR model uses the number of edge occurrences to learn item features, which captures dependencies between items for the first time.

(2) The PASR model learns the user's interest by considering the popularity of items to reflect their importance in the session.

(3) Experiments were conducted on two widely-used datasets (Tmall dataset and Nowplaying dataset). The results of the experiments demonstrate that our model exhibits excellent performance.

2. Model

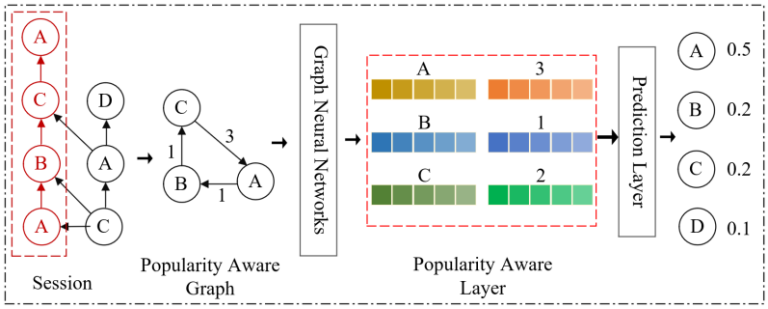

Figure 1. Process of PASR

A. Problem description

In a session-based recommendation system, we define the item set $V = \{v_1, v_2, \ldots, v_m\}$, where m is the total number of items, and the session sequence $s = [v_1, v_2, \ldots, v_t]$, where t is the length of the session. The primary objective of a session-based recommendation system is to predict the user's next click behavior based on the item sequence. For example, the next clicked item $v_{t+1}$is predicted according to $s = [v_1, v_2, \ldots, v_t]$, the session-based recommendation model outputs n candidate items that may interact according to the current interaction sequence.

B. Graph construction

To aggregate the representations of nodes in the graph, the session sequence is transformed into a graph using a graph neural network. For any given session sequence s, a directed graph can be constructed $G_s = \{V_s, E_s\}$, where V_s represents the node set of the session, E_s represents the edge set of session s, which includes item v_{n-1} and item v_n, and $\{v_{n-1}, v_n\} \in E_s$.

C. Graph aggregator

For session s, we defines the embedding of each item as $H = \{h_{v_1}, h_{v_2}, \dots, h_{v_m}\}$, where h_{v_i} refers to the unique hot code of the item $v_i (1 \le i \le t)$, m is the number of unique items in session s. Many works [13, 14] have proved that self-loops in graphs are beneficial to feature learning. We add self-connections for each node in the graph. For any item $\mathrm{v_n}$, there are 4 different types of edges:

$\mathrm{e_{self}}$: It represents the self-connection of the item.
$\mathrm{e_{out}}$: It represents the edge from item $\mathrm{v_n}$ to item $\mathrm{v_{n+1}}$.
$\mathrm{e_{in}}$: It represents the edge from item $\mathrm{v_n}$ to item $\mathrm{v_{n-1}}$.
$\mathrm{e_{in-out}}$: It indicates that there are edges from item $\mathrm{v_{n+1}}$ to item $\mathrm{v_n}$ and from item $\mathrm{v_n}$ to item $\mathrm{v_{n-1}}$.

Constructing a graph-based model with these edges allows the model to capture both the local and global structure of the session and learn features that are specific to each item, as well as the relationships between items. This, in turn, can lead to more accurate and relevant recommendations for the user.

Moreover, different edges appear in the dataset with varying frequencies, and frequently occurring edges often indicate common browsing habits. By segmenting the appearance times of edges and training different weight vectors for each segment, we can effectively capture the influence of edge frequency on feature aggregation in graph modeling. Specifically, we divide the number of occurrences of edges by a multiple of 10, and consider edges with more than 100 occurrences as a rare interval. By incorporating edge frequency into the weight vector training process, we can improve the accuracy and effectiveness of the PASR model in session-based recommendation tasks.

In the graph, the importance of item neighbors is reflected in the weight of edges. We believe that the weight is influenced by the type and number of occurrences of edges. Therefore, we define the following function to aggregate the features of neighbors:

$$X_{v_i} = Agg\left(h_{v_i}, h_{v_j}, e^{type}, e^{num}\right) \tag{1}$$

where, h_{v_i} represents the features of the target node, h_{v_j} represents the features of the neighbor, e^{type} indicates the type of edge, e^{num} represents the interval that the number of occurrences of the edge.

To capture the different types and frequencies of edges in the graph, we use a learnable embedding matrix $\mathrm{E^{type}} = [\mathrm{a}_1^{type}, \mathrm{a}_2^{type}, \mathrm{a}_3^{type}, \mathrm{a}_4^{type}]$, where each element represents the features of $\mathrm{e_{self}}$, e_{out}, $\mathrm{e_{in}}$, $\mathrm{e_{in-out}}$. In addition, we use the learnable embedding matrix $\mathrm{E^{num}} = [\mathrm{a}_1^{num}, \mathrm{a}_2^{num}, \dots, \mathrm{a}_n^{num}]$, where $\mathrm{a_i}$ represents the features of the edge in partition i. To capture the dependencies between items and improve the accuracy of recommendations, the PASR model learns to assign appropriate weights to each edge based on its type and frequency, by utilizing the aforementioned embeddings. We use the following method to learn the edge weights:

$$\mathrm{e_{ij}} = \gamma(\mathrm{w_1 a_{ij}^{type}} + \mathrm{a_{ij}^{num}} + \mathrm{b_1}) \tag{2}$$

$$e'_{ij} = \frac{\exp(e_{ij})}{\sum_{v_k \in N_{v_i}^s} \exp(e_{ik})} \tag{3}$$

where, $\mathrm{w_1, w_2} \in \mathrm{R^d}$ are weight, $\mathrm{b_1} \in \mathrm{R^d}$ is a bias, $\mathrm{a^{type}, a^{num}} \in \mathrm{R^d}$ are features of edges, γ is sigmoid or tanh activation function. $\mathrm{e_{ij}}$ is the weight of the learned edge. $N_{v_i}^s = \{\mathrm{v_j | v_i, v_j} \in \mathrm{V_s; j = i \pm 1}\}$ represents the neighbor set of item $\mathrm{v_i}$.

Finally, the features of each node can be updated to:

$$H_i = \sum_{v_j \in N_{v_i}^s} e'_{ij} X_{v_j} \tag{4}$$

D. Interest encoder

To learn the user's interest based on the item representation learned by the graph aggregator, we use an attention

mechanism. Unlike previous models, we focus on the popularity of each item to better understand the contribution of each item to the user's interest. To achieve this, we incorporate a learnable popularity embedding matrix $P_e = [p_1, p_2, \dots, p_n]$, where $e_i \in R^d$ represents the popularity embedding of item i, based on the number of occurrences.

By incorporating popularity into the attention mechanism, the model can better capture the diversity of user interests and make more accurate recommendations. We integrate the popularity vector into the calculation process of the attention mechanism, as follows:

$$\alpha_i = Q^T \sigma(wX_{v_i} + wX_a + b + p_1) \tag{5}$$

$$X_a = \frac{1}{n}\sum_{i=1}^{n} X_{v_i} \tag{6}$$

$$H^s = \sum_{i=1}^{l} \alpha_i X_{v_i} \tag{7}$$

where, $Q \in R^d$, σ is sigmoid activation function, $W_2 \in R^{d\times d}$, $W_3 \in R^{d\times 2d}$ are weight matrix, h_a is the average embedded in the session.

E. Prediction layer

Once the user's interest H^s has been obtained, the model calculates the dot product of the embedding and the interest of each candidate item. Subsequently, the softmax normalization is applied to obtain the probability of the user clicking on each item next time.

$$\hat{y}_i = \text{sofmax}\left(h_s^T h_{v_i}\right) \tag{8}$$

To learn the parameters of the model, the cross-entropy loss function is used, and the backpropagation algorithm is applied to train the model.

$$L(\hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{9}$$

where, y_i is the unique hot code of the session tag item.

3. Experiments

The experiment aims to demonstrate the effectiveness of the PASR model by addressing the following two questions:

Q1: Does the PASR model outperform the latest baselines?

Q2: Is the popularity-aware mechanism of the PASR model effective?

A. Experimental setup

Table 1. Statistics of the datasets

Dataset	Tmall	Nowplaying
# click	818,479	1,367,963
# train	351,268	825,304
# test	25,898	89,824
# items	40,728	60,417
avg. len.	6.69	7.42

Dataset: We employ two classic e-commerce datasets, namely the Tmall (https://tianchi.aliyun.com/dataset/dataDetail?dataId=42) and Nowplaying (http://dbis-nowplaying.uibk.ac.at/#nowplaying) datasets. To ensure the validity of the model, we adopt the same processing method as [6, 7, 10]. Prior to the experiment, we filter out sessions with a length of 2, or items that appear less than 5 times. We also filter out sessions with more than 20 reverse positions, as long-distance projects have little impact on the prediction results of the model but can slow down the running speed of the model.

For the Tmall dataset, we use click data from the last 100 seconds as the test set, and the remaining data as the

training set. For the Nowplaying dataset, we use data from the last two months as the test set, and the remaining data as the training set. Additionally, we employ data augmentation techniques by generating a series of sessions and corresponding tags for the session $s = [v_1, v_2, \ldots v_l]$. These include ($[v_1, v_2, \ldots, v_{l-1}], v_l$) , $([v_1, v_2, \ldots, v_{l-2}], v_{l-1}, \ldots, ([v_1], v_2)$. Such methods expand the training data and ensure the model's parameters are sufficiently trained.

The statistics of the datasets after preprocessing are summarized in Table 1.

Consistent with previous work, we adopt two commonly used evaluation indicators: P@20 and MRR@20.

P@20 measures the proportion of recommended items that are predicted correctly among the top 20 recommendation results. Its calculation formula is:

$$P@20 = \frac{1}{M}\sum_{i=1}^{M} y_i \tag{10}$$

The evaluation metric MRR@20 is defined as the reciprocal of the position of the highest ranked correct recommended item in the top 20 recommendation results.

where, *M* is the number of items that the user is actually interested in among the top 20 recommended items.

MRR@20 is calculated using the following formula:

$$MRR@20 = \frac{1}{M}\sum_{i=1}^{M} \text{Reci}\,(i)$$
$$\text{Reci}\,(i) = \begin{cases} \dfrac{1}{\text{Rank}\,(i)}, \text{Rank}\,(i) \le 20 \\ 0, \text{Rank}\,(i) > 20 \end{cases} \tag{11}$$

where, Rank(i) represents the order of labels in session i, and Reci(i) represents the reciprocal of the rank. Reci(i) is assigned a value of 0 if the rank is greater than 20.

These two indicators respectively reflect the accuracy of the model and the ranking of the tags in the candidate items. The larger the value of two indicators, the better the recommendation performance.

Parameters: We set the dimension of the hidden layer to 100 and the number of multiple attention heads to 5. The minimum batch size is set to 100. All weight matrices and embedding layers are initialized with a Gaussian distribution with a mean of 0 and a variance of 0.01. The initial value of all biases is set to 0.

To optimize the model, we use the Adam optimizer with an initial learning rate of 0.001 and an attenuation value of 1 every three epochs. The L2 penalty item is set to 10^{-5}. These parameters are chosen based on previous studies and empirical experiments to achieve the best performance of the PASR model on the given datasets.

B. Baselines model

To evaluate the performance of the proposed PASR model, we compare it with the following latest baseline models:

(1) POP: This model recommends the most popular items in the training set.

(2) Item-KNN [15]: This model uses cosine similarity between items for recommendation.

(3) FMPC [16]: In this model, the Markov chain is used for recommendation.

(4) GRU4REC [17]: This model uses GRU to learn the user's final interest.

(5) NARM [18]: This model combines GRU and attention mechanism for recommendations.

(6) STAMP [6]: This model uses the soft attention mechanism to learn the user's long-term interest and then combines the user's short-term interest for recommendations.

(7) SR-GNN [7]: This model uses a graph neural network to capture the transformation relationship between items and then uses the soft attention mechanism to learn the user's interest.

(8) GCE-GNN [10]: This model is an improvement of SR-GNN. It uses session graphs and a global graph to learn the representation of items.

(9) DIDN [11]: This model incorporates user behavior patterns hidden behind items in the click process.

(10) CORE [12]: This model unifies the representation space for both encoding and decoding processes to address the inconsistent prediction issue while recommending items. This algorithm can be used as the latest baseline.

These baseline models are chosen based on their popularity and effectiveness in previous studies.

C. Comparison with baselines

Table 2. Comparison with baselines

Method	Tmall		Nowplaying	
	P@20	MRR@20	P@20	MRR@20
POP	2.00	0.90	2.28	0.86
Item-KNN	9.15	3.31	15.94	4.91
FPMC	16.06	7.32	7.36	2.82
GRU4REC	10.93	5.89	7.92	4.48
NARM	23.30	10.70	18.59	6.93
STAMP	26.47	13.36	17.66	6.88
SR-GNN	27.57	13.72	18.87	7.47
GCE-GNN	33.42	15.42	23.11	7.55
DIDN	34.25	15.01	<u>23.16</u>	<u>7.59</u>
CORE	<u>34.67</u>	<u>15.56</u>	22.09	7.41
PASR	**35.91**	**15.83**	**23.32**	**7.83**

To address Q1, we compared the performance of the proposed PASR model with that of the commonly used baseline models listed in Section 3.B. In the experiment, we conducted 10 runs with different random seeds and recorded the average results. The experimental results are presented in Table 2, where the best-performing baseline and PASR model in each column are underlined and boldfaced, respectively. It can be observed from Table 2 that the PASR model achieves the best prediction performance on both datasets, outperforming the baseline models.

Among the traditional recommendation algorithms, POP is a very simple method that ignores the differences among users and has poor recommendation performance. Item-KNN and FPMC have improved to some extent, but they still have limitations. Item-KNN does not consider the sequence information of items, while the dimension explosion problem of FPMC makes it difficult to apply in actual production.

Recommendation algorithms based on deep learning, such as GRU4REC, NARM, and STAMP, have achieved better performance than traditional recommendation algorithms. However, GRU4REC and NARM based on GRU still suffer from the problem of gradient disappearance. STAMP only relies on the attention mechanism to learn the user's interest, but ignores the dependency between items.

Recommendation algorithms based on graph neural networks have further improved the performance of session recommendation because they can capture the transformation relationship between long-distance items. SR-GNN constructs the session into a graph, uses the gating graph neural network to learn the representation of the item, and then uses the soft attention mechanism to learn the user's interest. GCE-GNN uses the attention mechanism to evaluate and consider the importance of each item and generates the final item representation. DIDN incorporates item-aware, user-aware, and temporal-aware information to learn dynamic item embeddings and filter out noisy clicks within sessions. CORE applies a weighted sum for item embeddings to encode sessions and robust distance measuring techniques to prevent overfitting.

Although these models have achieved good recommendation performance, the PASR model still outperforms them. Unlike previous models, the PASR model constructs the item graph based on popularity, effectively capturing the preference of users for popular items. This ensures that the recommendations are tailored to the interests of the majority of users. The use of learnable embedding matrices E^{type} and E^{num} enables PASR to capture the different types and frequencies of edges in the graph. This allows the model to assign appropriate weights to each edge based on its type and frequency, which can better capture the dependencies between items and improve the accuracy of recommendations. Finally, the integration of popularity embeddings into the attention mechanism helps to better learn the user's interest and the contribution of each item to it. All of these factors together make PASR a powerful model for session-based recommendation, outperforming other state-of-the-art models.

D. Influence of popularity aware mechanism on recommendation performance

To address Q2 and demonstrate the effectiveness of the popularity-aware mechanism, we conducted the following comparative experiments:

(1) PASR-NP1: The number of occurrences of edges in the graph aggregator is not considered.

(2) PASR-NP2: The number of occurrences of items in the interest encoder is not considered.

(3) PASR-NP12: Neither the number of occurrences of edges in the graph aggregator nor the number of occurrences of items in the interest encoder is considered.

In this study, we fully tested the above comparison models and the PASR model on two datasets, and the experimental results are shown in Figure 2. The results demonstrate that the PASR model outperforms the other models, proving the importance of the popularity-aware mechanism. This is because the PASR model considers both edge occurrences in graph aggregators and item popularity in the interest encoder.

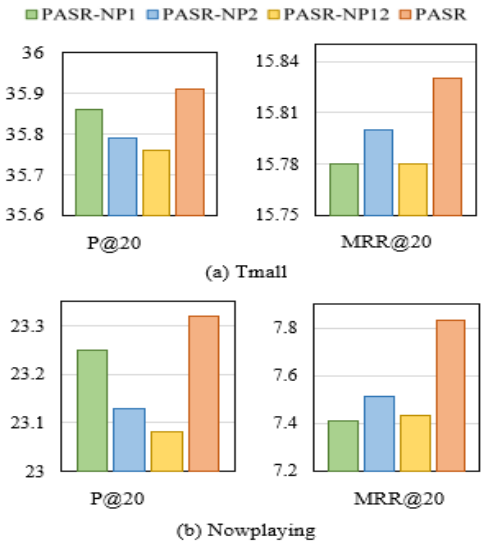

Figure 2. Influence of popularity aware on recommendation performance

The performance of PASR-NP1 and PASR-NP2 is lower than that of the PASR model under normal conditions. This is because the former models ignore the number of occurrences of edges in the graph aggregator, making it difficult to accurately learn the weight of edges. The latter model does not consider the number of occurrences of items in the interest encoder, leading to interest bias.

The PASR-NP12 model has the worst performance because it ignores both the number of edges in the graph aggregator and the number of items in the interest encoder. This makes the model unable to capture the impact of item popularity on user interests.

Overall, these comparative experiments demonstrate the effectiveness of the popularity-aware mechanism in the PASR model, which can accurately learn the contribution of each item and improve the accuracy of recommendations.

4. Conclusion

In this study, we proposed a popularity-aware graph neural network model for session-based recommendation systems. Our model investigates the impact of the number of edges and items in the graph neural network on recommendation performance. Through comprehensive experiments on two different datasets, we demonstrated that our proposed PASR model outperforms ten baseline schemes.

In future work, we plan to investigate the applicability of the PASR model in other recommendation systems, such as conversational recommendation systems. We also aim to explore the integration of other factors such as novelty and diversity into our model to further improve the quality of recommendations.

In conclusion, our proposed PASR model provides a powerful solution for session-based recommendation systems, which effectively captures the dependencies between items and improves the accuracy of recommendations. The success of our model also suggests that incorporating popularity-aware mechanisms can be a promising direction for improving recommendation performance in various domains.

Funding

This paper is partially supported by the Natural Science Foundation of Shandong Province (Grant No.: ZR2021MF099, ZR2022MF334); Undergraduate Education Reform Project of Shandong Province (Grant No.: M2021130, M2022245, Z2022202); Construction Project of High Quality Professional Degree Teaching Casebase in Shandong Province (Grant No.: SDYAL2022155); Shandong Province Key R&D Program (Soft Science Project) (Grant No.: 2021RKY03056).

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, Article ID: 421425, pp. 1-19, 2009. https://doi.org/10.1155/2009/421425.

[2] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims, "Playlist prediction via metric embedding," In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12), Beijing, China, 2012, pp. 714-722. https://doi.org/10.1145/2339530.2339643.

[3] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," In 4th International Conference on Learning Representations, 2016, pp. 1-10. http://arxiv.org/abs/1511.06939.

[4] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 2016, pp. 17-22. https://doi.org/10.1145/2988450.2988452.

[5] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, Singapore, 2017, pp. 1419-1428. https://doi.org/10.1145/3132847.3132926.

[6] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: Short-term attention/memory priority model for session-based recommendation," In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, United Kingdom, 2018, pp. 1831-1839. https://doi.org/10.1145/3219819.3219950.

[7] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 1, 2019, pp. 346-353. https://doi.org/10.1609/aaai.v33i01.3301346.

[8] S. Sang, N. Liu, W. Li, Z. Zhang, and Q. Qin, "High-order attentive graph neural network for session-based recommendation," *Appl. Intell.*, vol. 52, pp. 16975-16989, 2022, pp. 1-15. https://doi.org/10.1007/s10489-022-03170-7.

[9] Q. Sun, Z. Zhang, S. Sang, and F. Dong, "Time and position aware graph neural networks for session-based recommendation," In 2021 7th International Conference on Computer and Communications (ICCC), 2021. https://doi.org/10.1109/ICCC54389.2021.9674354.

[10] Z. Y. Wang, W. Wei, G. Cong, X. L. Li, X. L. Mao, and M. H. Qiu, "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, Virtual Event China, 2020, pp. 169-178. https://doi.org/10.1145/3397271.3401142.

[11] X. Zhang, H. Lin, B. Xu, et al., "Dynamic intent-aware iterative denoising network for session-based recommendation," *Info. Processing Manag.*, vol. 59, no. 3, Article ID: 102936, 2022. https://doi.org/10.1016/j.ipm.2022.102936.

[12] Y. Hou, B. Hu, Z. Zhang, and W. X. Zhao, "CORE: Simple and effective session-based recommendation within consistent representation space," In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 2022, pp. 1796-1801. https://doi.org/10.1145/3477495.3531955.

[13] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," In Proceedings of the International Conference on Learning Representations (ICLR), vol. 1, 2017, pp. 1-14. http://arxiv.org/abs/1609.02907.

[14] H. Yang, K. Ma, and J. Cheng, "Rethinking graph regularization for graph neural networks," In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 5, 2021, pp. 4573-4581. https://doi.org/10.1609/aaai.v35i5.16586.

[15] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," In Proceedings of the Tenth International Conference on World Wide Web (WWW '01), Hong Kong, China, 2001, pp. 285-295. https://doi.org/10.1145/371920.372071.

[16] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized Markov chains for next-basket recommendation," In Proceedings of the 19th International Conference on World Wide Web (WWW '10), Raleigh, North Carolina, USA, 2010, pp. 811-820. https://doi.org/10.1145/1772690.1772773.

[17] R. Devooght and H. Bersini, "Collaborative filtering with recurrent neural networks," *Info. Retrieval*, vol. 1, 2016. https://doi.org/10.48550/arXiv.1608.07400.

[18] Y. J. Ko, L. Maystre, and M. Grossglauser, "Collaborative recurrent neural networks for dynamic recommender systems," In Proceedings of Machine Learning Research (PMLR), 2016, pp. 366-381. http://proceedings.mlr.press.