Proceedings of Machine Learning Research 1–11

Graph Neural Networks for Sequential Recommender Systems

MLULEKI DLADLA 219051318@stu.ukzn.ac.za
University of KwaZulu-Natal

Abstract

Sequential recommender systems (SRSs) are a type of recommender systems that take into account the sequential dependencies among a user's interactions with items. This paper Investigate Session-based Recommender systems with Graph Neural Networks (SBRNN). The Idea is to explore Global graph because most solutions for SBRNN implement session graphs. This limits them to only using session-level item representation which only captures how items transition between pairs within on going session. On the other hand, Global graph involves acquiring the item embedding at a global level by examining the pairwise transitions of items across all sessions. The solution proposed in this study explored different aggregation operations in Global Context Enhanced Graph Neural Networks with Self Contrastive learning for Session-based Recommendation(GCE-GNN+SCL). Aggregation is a crucial component that is responsible for combining information from neighbouring nodes in a graph to generate representations for the target node. We conducted experiments on two commonly used benchmark datasets(TMall and NowPlaying). Our experiments revealed that Max pooling and Sum Pooling aggregation operation boosts the performance of the model in terms of Precision@20 and in Mean Reciprocal Rank@20.

1. Introduction

Sequential recommendation systems [1] are used in modern online information systems such as news, video, and advertisements to predict a user's next behaviour by analysing their historical behaviour sequences. This approach differs from traditional recommendation systems that model user preferences statically and sequential recommender systems can capture a user's dynamic and evolving preferences. In an era of exponential information growth on the internet, recommendation systems have become indispensable. They serve as the gatekeepers of user experience in a wide range of web applications, from search engines to e-commerce platforms and media streaming services. Their primary purpose is to help users navigate the overwhelming volume of available content and find items that align with their interests. However, there's a significant challenge that arises in the world of recommendation systems. Most conventional systems assume the continuous recording of user profiles and long-term historical interactions. But what if this information is not available? In many cases, user identities remain concealed, and we only have access to their behavior during a single session. This limitation prompts the need for innovative recommendation systems that can perform effectively under such constraints. This is where session-based recommender systems come into play. These systems have garnered considerable attention recently as they focus on predicting the next item of interest for users based solely on their actions within a single session. This approach is particularly valuable in scenarios where traditional recommendation methods like collaborative filtering fall short due to the absence of long-term user data or user

profiles. Early research into session-based recommendation systems has mainly fallen into two categories:

Similarity-Based: Similarity-based approaches suggest items based on what's currently popular. They look for things that tend to go together at the same time. However, they might miss the fact that you have a specific order or pattern in how you watch videos. For example, you might always watch comedy before action movies. So, these approaches may not always give you recommendations that fit your viewing habits perfectly.

Chain-Based: Now, think of chain-based approaches as trying to figure out all the different ways you could watch videos. It's like they're trying to predict all the possible paths you might take while watching content. However, when there are a lot of videos available, this can get really complicated. Imagine if there were thousands of videos, and they had to consider every possible combination you might watch them in. This can become a huge computational problem and may not be very practical, especially with a vast number of options.

We illustrate the global-level item transition modelling in Figure 1.To illustrate this concept, consider Session 2. We can gather valuable information about how items transition within it by looking at both "Session 1" and "Session 3." For instance, we might discover a new connection between items, such as the transition from "Iphone" to "Phone Case." This shows how items in different sessions can help us understand the relationships between them.

This is where global context comes into play, which implements a global aggregation strategy that combines information from the target node and its neighbors using weighted interactions and linear transformations.

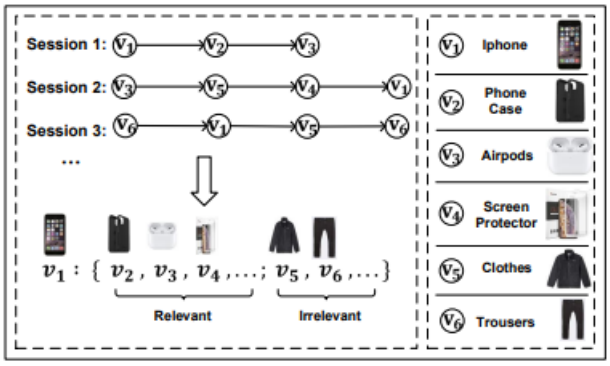

Figure 1: An illustrative example of modeling item transitions at a global level.[12]

Figure 2 shows the structure of GCE-GNN, which has four main parts:

Global-level Item Representation Learning Layer: This layer helps us learn how items are connected in all the sessions. It uses a special attention mechanism that understands the context of each item in relation to its neighbors in the global graph.

Session-level Item Representation Learning Layer: This part uses a GNN model to figure out how items are connected within one session. It's all about understanding the relationships between items in the current session.

Session Representation Learning Layer: Here, we try to capture what the user likes in the current session by bringing together the item information we learned in both the session and global levels.

Prediction Layer: Finally, this layer tells us the likelihood of each item being a good recommendation. It helps us decide which items to suggest to the user.

GCE-GNN is a model that works in several steps. First, it builds a global graph using all the training session sequences. Then, for each session, it uses both a global feature

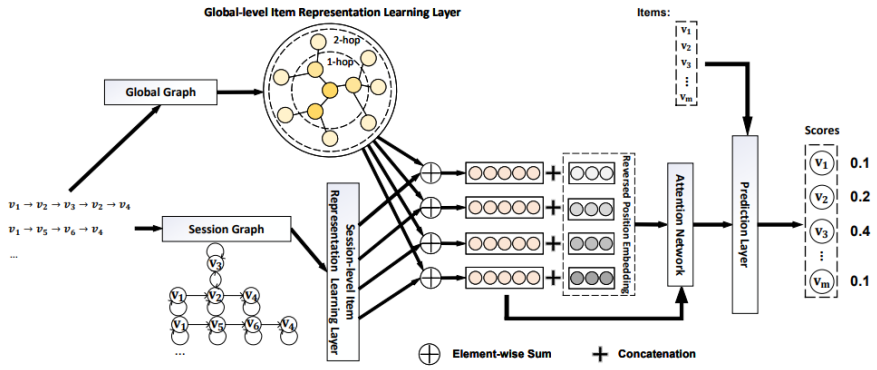

Figure 2: The GCE-GNN (Global Context Enhanced Graph Neural Network) [12]

encoder and a local feature encoder to gather information about each item in the session. This allows the model to consider both the broader context (global) and the immediate context (local) when analyzing the items.To further improve its predictions, the model takes into account the position of each item in the session, learning how each item contributes to predicting the next one. Finally, the model assigns scores to candidate items, helping it make informed recommendations or predictions.

This study aim to answer the following research questions:

- **RQ1**: Can GCE-GNN+SCL outperform the performance of the state of the art GCE-GNN method in Tmall and Now Playing dataset with a low computational cost.
- **RQ2**: Which aggregation operation works best for GCE-GNN+SCL?

The remainder of this paper is organised as follows: Section 2 presents the literature review, Section 3 presents Preliminaries, Section 4 presents Methodology, Section 5 presents Results, Section presents 6 Conclusion and Future Work.

2. Literature Review

This literature review aims to synthesize key works in the field of sequential recommender systems. In recent years, there has been a growing interest in using neural network-based methods to improve Sequential Recommendation Systems (SBR). Researchers have made significant contributions in this area, each with their own unique approaches and enhancements. Hisasi et al. [2] introduced the GRU4REC model, which marked an early attempt to utilize Recurrent Neural Networks (RNN) for SBR. They employed a multi-layer Gated Recurrent Unit (GRU) to model sequences of user-item interactions. Building upon this, Tan et al.[8] extended GRU4REC by incorporating data augmentation techniques to further enhance the recommendation system.

Li et al. [3] proposed NARM, which integrated an attention mechanism into a stacked GRU encoder. This allowed them to capture more informative item-transition patterns in SBR. Liu et al.[4] in-

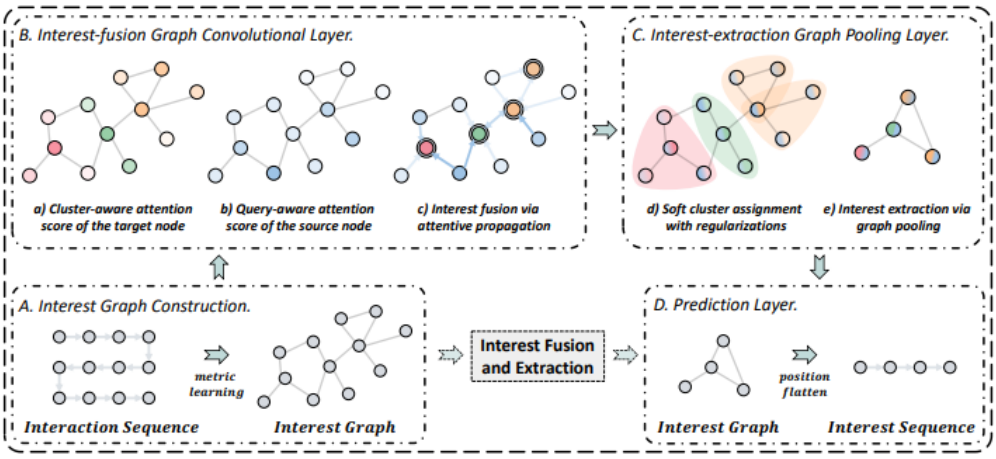

Figure 3: Sequential Recommendation with Graph Neural Networks model [1]

troduced STAMP, an attention-based short-term memory network that focused on capturing the user's immediate interests without relying on RNNs. Both NARM and STAMP highlighted the significance of the user's last click through the use of attention mechanisms. MCPRN [9] introduced a novel approach to modeling multiple purposes within a given session, utilizing a mixture-channel model for SBR. However, it's important to note that both RNN-based and MC-based methods primarily focus on modeling sequential transitions between adjacent items to infer user preferences, thus struggling to capture complex item-transition patterns, such as non-adjacent transitions.

More recently, several proposals have emerged that leverage Graph Neural Network (GNN)-based models. These models construct graphs from the current session's interactions to learn item embeddings for recommendation. Wu et al.[14] presented SR-GNN, a gated GNN model, which learns item embeddings on the session graph and calculates representative session embeddings by considering the relevance of each item to the last one. Following the success of SR-GNN, various variants like GC-SAN [16] were introduced for SBR. Qiu et al.[5] proposed FGNN, which learns item representations by aggregating neighboring item embeddings using multi-head attention and repeatedly combines these learned embeddings, taking into account their relevance to the session over time.

Incorporating GNNs can enable the modelling of sequentialdependencies within sessions, allowing for better understanding of user preferences over time. By leveraging the temporal information present in session data, GNNs can capture the evolving user interests and preferences, leading to more accurate recommendations. Session-based recommender systems often encounter challenges in handling implicit signals, where user preferences are not explicitly expressed. GNNs can aid in uncovering implicit preferences by leveraging the collaborative signals present in session data. By considering the interactions between items within a session, GNNs can capture the underlying relationships and uncover latent preferences that

may not be evident in individual item interactions. GNNs provide a powerful framework on Figure 3 for session-based recommender systems to capture sequential dependencies, leverage temporal information, and uncover implicit preferences, ultimately improving the accuracy and effectiveness of recommendations. This model is about making recommendations in a sequence using Graph Neural Networks.Here's how it works: First, it turns each interaction sequence into an interest graph using a method called metric learning (A). Then, it dynamically combines and extracts interests from the graph (B and C). After that, it identifies the currently most important interests (D) by simplifying the graph, and these interests are used for making further recommendations.Think of it like this: it takes all the things you've interacted with, turns them into a kind of map, picks out the most important places on the map, and uses those to suggest what you might like next.[1]

In contrast to these existing approaches, the proposed model focuses on learning item-transition information not just within the current session but also considers information from across multiple sessions. This approach aims to enhance the recommendation system by capturing more comprehensive patterns of item transitions.

3. Preliminaries

3.1. Self Contrastive Learning (SCL)

SCL is a solution proposed to address issues related to item representation in recommender systems.

The core idea behind SCL is to make sure that the representations of items are spread out evenly in a representation space. To do this, . Shi et al.[6] introduce an additional loss objective in the training process. This objective is designed to penalize the closeness of item representations to each other. In other words, it encourages each item representation to be distinct from all the others. They achieve this by calculating a Loss of SCL (LSCL) that measures the similarity between item representations and tries to push them apart on a unit hypersphere (a mathematical space where all items have a consistent length of 1). The temperature parameter controls the strength of this push, and the use of cosine similarity helps measure the distance between items.

(1)

$$L_{SCL} = -\sum_{i=1}^{n} log \frac{g(x_i, x_i)}{\sum_{j=1}^{n} g(x_i, x_j)}$$

This SCL Loss (LSCL) is then integrated into existing recommendation models. The overall loss function used for training now includes the original model's loss and the LSCL loss, with a hyperparameter Beta determining how much importance is given to each of these objectives. Importantly, LSCL has a significant positive impact on ensuring that the item representations are evenly distributed in the representation space.

Shi et al. highlight [6] three main advantages of using SCL:

(i) Improved representation spaces: SCL helps create more uniform item representation spaces, which, in turn, leads to better recommendation system performance.

(ii) Simplified modeling process: SCL eliminates the need for creating complex positive/negative sample pairs and data augmentation techniques. Instead, each item representation acts as a positive sample, and all others are treated as negative samples without any further modifications. This simplifies the recommendation system's construction and makes it more efficient.

(iii) Seamless integration: SCL can be easily integrated into existing recommendation systems without requiring architectural changes. This makes it a versatile and

adaptable solution for various recommendation system scenarios. SCL is a technique that improves recommendation systems by ensuring that item representations are evenly spread out in the representation space, simplifying the training process, and seamlessly fitting into existing models and it is effective with low computational cost

3.2. Aggregation Operations

In this section, we delve into the various aggregation methods[13]: **Max Pooling**:A technique that selects the maximum values from the local and global features. We extract the highest value from each feature dimension, and for every i-th dimension of an item's representation, h'_{vi} we express it as follows:

(2)
$$h'vi = max(h^g_{vi}, h^s_{vi})$$

Concatenation Mechanism: An approach that joins local and global features side by side. the resulting representation is formed by combining vectors h^g_v and h^s_v through concatenation.

(3)
$$h'v = M(h^g_v||h^s_v)$$

where $M \in \mathbb{R}^{dx2d}$ is a transformser weight.

Sum Pooling: A process that adds up the local and global features element-wise. it involves taking the summation of values from each dimension or feature of a set of vectors. The final representation is derived by adding up the values across all dimensions, resulting in a single vector that encapsulates the cumulative information from the original vectors.

(4)
$$h'_v = \sum_{l=0}^{L} h^{(l)}_v$$

Gating Mechanism: A method that emmploys learned weights to combine local and global features effectively. We employ a linear technique that combines the local feature representation denoted as h^l and the global feature representation as h^g.

(5)
$$r_v = \sigma(W_s h^l_v + W g h^g_v)$$

$$h'v = r_v h^g_v + (1 - r_v)h^s_v$$

Where σ represents the sigmoid activation function, and r_v is a parameter that is learned to adjust the relative importance of two features.

4. Methodology

In this section we discuss the methodolody we used to evaluate solution to the research questions raised in this study.

4.1. Experimental Design

In our implementation, we conducted experiments using the SCL (Self-Constrastive Learning) method with the state-of-the-art GCE-GNN model. Initially, we meticulously replicated the experimental outcomes of these models by adhering to the specifications and guidelines outlined in Wang et al.[12] original research paper. Subsequently, evaluate the integrated SCL approach into the GCE-GNN model. We also going to investigate which aggregation operation works best for GCE-GNN+SCL.

Regarding the hyperparameters used in SCL, we set the temperature parameter to 0.1 and explored a range of values for the loss weight parameter, varying it from 0.1 to 100. It's worth noting that, to showcase the effectiveness of our SCL method, we maintained the hyperparameter settings recommended in the original papers. In other words, we made no alterations to the pre-existing configurations, except for the introduction of our SCL method.

In this research work, we will delve deeper into hyperparameter tuning, which has the potential to further enhance the performance of the GCE-GNN+SCL method, building upon the groundwork we've established in this initial implementation phase.

4.2. Datasets and Pre-processing

We utilized three benchmark datasets: Tmall and Nowplaying. The Tmall dataset was sourced from the IJCAI-15 competition and contains anonymized user shopping logs from the Tmall online shopping platform. The Nowplaying dataset, on the other hand, is obtained from a previous work [17] and provides insights into user music listening behavior. To prepare these datasets for analysis, we followed a preprocessing procedure similar to previous studies [14]. This process involved filtering out sessions with only one item and removing items that appeared fewer than five times across all three datasets.

Consistent with the approach in [4], we divided the data into two segments: the latest data from the last week was reserved for testing, while the historical data was used for training. This division allowed us to assess the performance of our methods on recent data while training our models on a broader historical context.

4.3. Evaluation Metrics

In accordance with the methodology established in prior studies [15], our assessment of the effectiveness of the proposed SCL (Self-Constrative Learning) method relies on two key metrics: Precision at k (P@k) and Mean Reciprocal Rank at k (MRR@k), with the specific value of k set at 5, 10, or 20. Precision at k (P@k) is a widely-used indicator of predictive accuracy, offering insight into the proportion of correctly recommended items within the top-k recommendations.

On the other hand, Mean Reciprocal Rank at k (MRR@k) considers the ranking order of the recommended items and calculates the average reciprocal rank of the correctly recommended items. A higher MRR@k value signifies that accurate recommendations are positioned at the top of the list, indicating superior performance

4.4. Impact of Dropout Setting

To mitigate the risk of overfitting in GCE-GNN and GCE-GNN+SCL model, we implement dropout regularization techniques, as recommended in reference [7]. This approach has demonstrated its effectiveness in a wide range of neural network designs, including graph neural networks as highlighted in references [10] and [11]. The fundamental concept behind dropout involves the random exclusion of neurons with a specified probability (denoted as p) during the training phase, while during testing, all neurons are utilized.

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

Table 1: Statistics of the used datasets.

When the dropout ratio is low, the model struggles to perform well as it tends to overfit the data, reaching its peak performance at 0.6. On the other hand, when the dropout ratio is high, the model's performance declines because it finds it challenging to learn from the data due to limited available neurons.

5. Results

5.1. Comparison of GCE-GNN and GCE-GNN+SCL

The results depicted in Table 2 showcase the performance comparison, where we integrated the proposed SCL method with the GCE-GNN model. These experiments reveal a consistent enhancement in model performance, particularly in the context of Precision at k (P@k) and Mean Reciprocal Rank at k (MRR@k), across two datasets, namely Tmall and Nowplaying. This improvement signifies a notable advancement, positioning the approach as the new state-of-the-art in these performance metrics. For a more comprehensive breakdown of the experimental outcomes on each dataset,

Please refer to Table 2. We displays the results of the results of the comparisons of the GCE-GNN and the GCE-GNN+SCL on the development set across two datasets (RQ1). Please note that the results marked with a single asterisk (*) have been directly extracted from the original GCE-GNN research paper[12], while results marked with a double asterisk (**) We independently reproduced in our study. The table also illustrates the extent of performance improvement in comparison to our reproduced results. The best-performing outcomes in each column are emphasized with bold font for clarity. GCE-GNN+SCL** results were obtained by using Sum Pooling which was used in the original paper GCE-GNN.

5.2. Impact of Aggregation Operations (RQ2)

Given the utilization of both local and global feature encoders in our model, it is pertinent to evaluate the performance of GCE-GNN and GCE-GNN+SCL with different aggregation operation. The aggregation operations considered were max pooling, concatenation mechanism, and Sum Pooling.

As we incorporate local feature and global feature e into our model, it is important to explore and contrast the performance of GCE-GNN and GCE-GNN+SCL under different aggregation techniques. This comparison helps us understand how various methods of merging information from local and global features affect the model functionality and outcomes.

The results in Table 3 shows a comparison of the different aggregation operations in the context of the GCE-GNN as reported in the original paper [12]. We could not reproduce the results, giving to the fact that GCE-GNN requires a lot of computational resources and it was taking a lot of time to run the experiments.

Table 4 Displays the perfomance of Aggregation operations for GCE-GNN+SCL and the best results are highlighted in bold , we can see how well GCE-GNN+SCL performs using different aggregation operations on two datasets. What stands out is that when we use Max pooling with GCE-GNN+SCL, it does better than other operations in terms of P@20 and MRR@20 on the Tmall and NowPlaying datasets. In simpler terms, using Max pooling with GCE-GNN+SCL gives us the best results which even outperforms the state of the art method.The gate mechanism and concatenation methods may underperform because having too many parameters can lead to overfitting.

5.3. Impact of Global Feature Encoder

In the following steps, we will perform experiments using the Tmall dataset to assess how well the global-level feature encoder and session-level feature encoder perform. We

Method	TMall		Now Playing	
	P@20	MRR@20	P@20	MRR@20
GCE-GNN*	33.42	15.42	22.37	8.40
GCE-GNN**	32.52	15.20	22.42	8.45
GCE-GNN+SCL**	**33.65**	**15.55**	**22.81**	**8.47**

Table 2: Comparison of GCE-GNN and GCE-GNN+SCL

Dataset	TMall		Now Playing	
	P@20	MRR@20	P@20	MRR@20
Max Pooling	31.87	15.39	19.13	6.71
Concatenation	31.55	14.89	19.88	7.93
Sum Pooling	**33.42**	**15.42**	22.37	**8.40**
Gate Mechanism	32.80	15.33	**22.47**	7.83

Table 3: Results of comparison of different aggregation operation as reported in the original paper GCE-GNN[12]

Dataset	TMall		Now Playing	
	P@20	MRR@20	P@20	MRR@20
Max Pooling	**35.11**	**15.83**	21.83	**8.78**
Concatenation	31.67	14.6	20.12	7.85
Sum Pooling	33.65	15.55	**23.62**	8.42
Gate Mechanism	32.98	15.22	22.55	7.93

Table 4: Aggregation operations results for GCE-GNN+SCL

have designed three different models for comparison in this context.

- GCE-GNN+SCL w/o session:GCE-GNN+SCL without session-level feature encoder and only with global feature.

- GCE-GNN+SCL-1-hop: with global-level feature encoder,which sets the number of hops to 1.

- GCE-GNN+SCL-2-hop: with global-level feature encoder,which sets the number of hops to 2.

TMall	P@20	MRR@20
w/o session	32.96	12.41
1-hop	33.42	15.42
2-hop	32.58	14.83

Table 5: Imapct of Global feature Encoder: Results are from the original paper GCE-GNN[12].

TMall	P@20	MRR@20
w/o session	18.17	8.89
1-hop	35.11	15.83
2-hop	33.91	15.41

Table 6: GCE-GNN+SCL results for Impact of Global Feature Encoder

Table 6 and 5 presents a comparison of various contrast models. It's evident that when utilizing a global-level feature encoder, GCE-GNN+SCL achieves superior performance. When we compare GCE-GNN without session information to GCE-GNN+SCL without session information, the former performs better, suggesting that GCE-GNN+SCL benefits from both local and global features, as global features alone are not as effective.

GCE-GNN+SCL, equipped with 1-hop and 2-hop global-level feature encoders, can extract valuable item-transition information from other sessions, improving the model's predictive accuracy. Furthermore, on the Tmall dataset, it's noticeable that GCE-GNN+SCL with 1-hop and 2-hop outperforms GCE-GNN with 1-hop and 2-hop highlighting that deeper exploration can yield more valuable insights from the global graph. However, it's worth noting that, on Tmall, GCE-GNN+SCL with 1-hop performs better than GCE-GNN+SCL with 2-hop, suggesting that overly deep exploration may introduce noise.

6. Conclusion & Future Work

In this research, We studied the different aggregation operations for GCE-GNN and implemented them in GCE-GNN+SCL method. SCL which aims to make item representations more consistent in modern session-based recommendation systems, played a key role in this research. Introducing different aggregation operations significantly boosted the performance of GCE-GNN+SCL. The experiments clearly show that GCE-GNN+SCL is performs much better than GCE-GNN. We also took a closer look at how the Global Feature Encoders work and observed that Global Feature Encoders have significant impact on the model performance.

As part of our future work we will investigate whether SCL can improve the perfomance of other state of the art session-based recommender systems algorithms

References

[1] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. Sequential recommendation with graph neural networks. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 378–387, 2021.

[2] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[3] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1419–1428, 2017.

[4] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin 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*, pages 1831–1839, 2018.

[5] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 579–588, 2019.

[6] Zhengxiang Shi, Xi Wang, and Aldo Lipani. Self contrastive learning for session-based recommendation. *arXiv preprint arXiv:2306.01266*, 2023.

[7] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[8] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 17–22, 2016.

[9] Shoujin Wang, Liang Hu, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Longbing Cao. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 2019.

[10] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pages 165–174, 2019.

[11] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[12] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui 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*, pages 169–178, 2020.

[13] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[14] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353, 2019.

[15] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4503–4511, 2021.

[16] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, volume 19, pages 3940–3946, 2019.

[17] Eva Zangerle, Martin Pichl, Wolfgang Gassler, and Günther Specht. # nowplaying music dataset: Extracting listening behavior from twitter. In *Proceedings of the first international workshop on internet-scale multimedia management*, pages 21–26, 2014.