Enabling Session-Based Recommender Systems Through Graph Convolutional Networks*

Boudjemaa Boudaa[1,*,†], Kheira Amel Belhocine[2,†] and Amel Guelfout[3,†]

[1]Department of Computer Science, University of Tiaret, Tiaret, 14000, Algeria
[2]Department of Computer Science, University of Tiaret, Tiaret, 14000, Algeria
[3]Department of Computer Science, University of Tiaret, Tiaret, 14000, Algeria

Abstract

The explosive expansion of online platforms has generated an abundance of information and choices, underscoring the significance of personalized recommendations in improving the user experience and overall satisfaction. In this paper, our primary emphasis lies in the domain of session-based recommender systems (SBRSs), where we aim to offer precise recommendations by taking into account users' sequential behaviour and short-term preferences. To achieve this goal, we leverage the capabilities of Graph Convolutional Networks (GCNs), known for their extraordinary potential in modelling intricate user-item associations and capturing the underlying patterns within user sessions. The proposed GCN-based SBRS model's effectiveness is rigorously assessed across three publicly available real-world datasets. The experimental results demonstrate the superior performance of our model compared to several established baselines and other architectures in the field of SBRS.

Keywords

Session-based Recommender System, Graph Convolutional Network, Next-Item Recommendation

1. Introduction

Recommender Systems (RSs) have been the subject of extensive research over the past decade and have proven to be valuable in numerous contexts. In the age of the Internet and e-commerce, businesses are increasingly turning to RSs as a means to enhance their sales performance. RSs offer predictions of items that users may find appealing for consumption [1]. Many algorithms designed for this purpose primarily concentrate on delivering recommendations tailored to the user's preferences [2].

Session-based recommender systems (SBRSs) are a type of recommender systems that make recommendations based on users' short-term interests and preferences [3]. They are becoming popular in several domains such as e-commerce, music streaming, and news recommendation. SBRSs are challenging to build due to issues such as data accuracy, sparsity, short-term user behaviour, and the lack of explicit user feedback.

This paper aims to handle the accuracy challenge by improving recommendation results by leveraging the power of Graph Convolutional Networks (GCNs) [4], which have shown remarkable potential in modelling complex user-item relationships and capturing the latent patterns within sessions. The remainder of this paper is structured as follows: in Section II, we provide an overview of SBRS and GCN. Section III discusses related work. Section IV presents a detailed description of our GCN-based Model for predicting the next item in SBRSs. Section V outlines the methodology used to experiment with the proposed model. In Section VI, we present and discuss the obtained results. Finally, Section VII concludes this paper and introduces our future research.

2. Fundamentals

2.1. What is a session?

A session denotes a sequence of user interactions with an application or website occurring within a short time frame. These interactions encompass a variety of actions, including clicks, views, purchases, searches, and others. Each session is associated with distinct session attributes, such as the items viewed, the duration spent on each item, the sequence of item views, and the time intervals between consecutive interactions. While a session typically represents a user's present preference, it's important to note that a user's intention within a session can sometimes undergo local shifts. [5].

6th International Hybrid Conference On Informatics And Applied Mathematics, December 6-7, 2023 Guelma, Algeria

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

*Corresponding author.
†These authors contributed equally.
boudjemaa.boudaa@@univ-tiaret.dz (B. Boudaa);
kheiraamel.belhocine@univ-tiaret.dz (K. A. Belhocine);
amel.guelfout@univ-tiaret.dz (A. Guelfout)
0000-0002-1356-4775 (B. Boudaa)

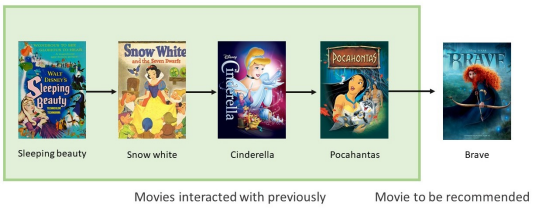

Figure 1: An SBRS example on Movies Watched.

2.2. Session-Based Recommender Systems

As illustrated in Figure 1, the session-based recommendation is a specialized task that focuses on predicting the next item a user is likely to choose based on their recent interactions within a single session. This task is distinctive due to the sequential nature of user behaviour within a session, the potential for repeated interactions with specific items, and the necessity for providing recommendations in a timely manner. This approach is particularly relevant in applications like e-commerce and media streaming. In this study, we introduce simple yet powerful linear models tailored to understand and enhance session-based recommendation systems.

2.3. Graph Convolutional Network

Graph Convolutional Networks (GCNs) are a category of deep learning models specifically designed for processing graph-structured data. In contrast to conventional neural networks, GCNs expand neural network architectures to accommodate non-Euclidean domains, which are typically represented by graphs or networks [6, 7]. The fundamental concept behind GCNs is to adapt the idea of convolutional layers from grid-like data, such as images, to graph structures. In traditional convolutional neural networks (CNNs), convolutions work with local pixel neighbourhoods, taking advantage of the grid's structure data. However, in GCNs, convolutions are defined in the spectral or spatial domain of graphs, leveraging the connectivity patterns between nodes [6, 7]. In GCNs, they usually work within a message-passing framework, where each node aggregates and combines information from its neighbouring nodes. This aggregation process is analogous to the receptive field in CNNs, allowing nodes to gather information from their local graph neighbourhood. The gathered information is then used to update the node's representation or features. By iterative propagating and aggregating information across the graph, GCNs learn to capture the graph structure and perform node-level or graph-level predictions [6, 7]. We can think of GCN as analogous to the filter used in convolution, as it represents the adaptation of convolution from the typical grid structure to more irregular structures such as graphs [8]. The node embeddings are updated as follows [4]:

$$H^{(l+1)} = \delta\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}H^{l}W^{l}\right) \tag{1}$$

where: $H^{(l+1)}$ represents the embedding matrix of the l-th layer convolution, d is the embedding dimension. A is the adjacency matrix with self-loops, and $D_{ii} = \sum_j A_{ij}$ is the degree matrix. All these operations make sure that all the neighbours are receiving an aggregated message from multiple hop neighbours. The weights W^l in the GCN are trained using gradient descent.

3. Related Work

The literature of SBRS knows a few works that are based on GCN. In [9], the authors propose GACOforREC model for session-based recommendation in order to handle long-term and short-term preferences of users and preserve the hierarchy of potential preferences. This model has used convolution operations of GCN to learn the order within the session and the spatiality within the network to capture the user's short-term preferences. For learning long-term preferences, it applied ConvLSTM, a variant of the LSTM network. In addition, GACOforREC proposes a new pair adaptive attention mechanism (Long-Attention and Short-Attention) based on GCNs to pay attention to the influence of different propagation distances of GCNs. To enhance the model's hierarchical learning of various preferences, ON-LSTM has been introduced, it is a network structure that focuses more on hierarchy and neuron ordering. This ordering is essential to the overall perception of the model's user preferences for accurate recommendations. Another GCN-based SBRS Model called AUTOMATE was presented in [10], It integrates a graph convolutional layer based on Auto-Regressive Moving Average (ARMA) filters. It can capture complex transformations between items through sessions modelled as graph-structured data. The core principle behind AUTOMATE revolves around leveraging the ARMAConv layer, which allows us to merge enduring user preferences with real-time session interests to generate the graph transfer signal. Recently, in order to capture the complex high-order information between items in real-world scenarios, the authors in [11] have proposed DHCN (Dual Channel Hypergraph Convolutional Networks). This model is based on a hypergraph using convolution operations and integrates self-supervised learning to generate a high-quality session-based recommendation. Still with the use of hypergraphs, the study in [12] has introduced HyperS2Rec, which it takes into account both item consistency and sequential item dependence simultaneously. This model leverages hypergraph-structured data through HGCN and captures sequential information using GRU to collectively model user preferences. In this

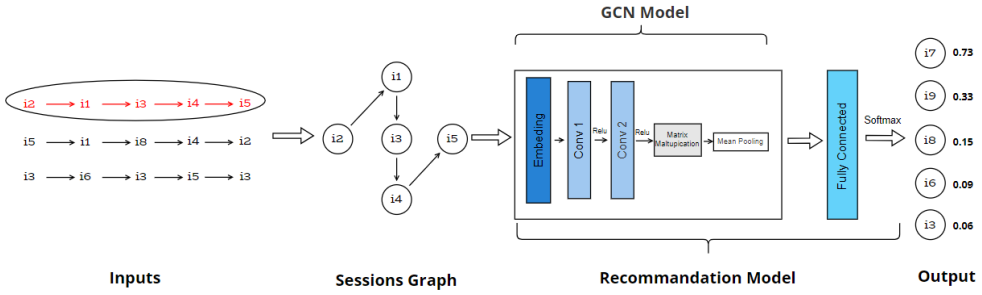

Figure 2: An Overview of GCN-Based SBRSs Model.

proposal, the reversed position embedding mechanism and soft attention mechanism are combined to derive session representations.

4. A GCN-based SBRS Model

This section presents our proposed model for an SBRS based on GCN.

4.1. Model Architecture

Figure 2 depicts the architecture of our proposed model which consists of three main steps: session presentation, processing, and recommendation.

4.2. Session Representation Step

Each session sequence s can be modelled with the adjacency matrix as a directed graph G = (I ; E). In this session graph, each node represents an item i and each edge means that a user clicks item i1 after i2 in the session.

4.3. Processing Step

It consists of an embedding layer, followed by two convolutional layers with ReLU activation functions. The key layers and operations in the model are:

- Embedding Layer: The self embeddings layer maps item indices to dense vectors of hidden dimensions. It learns meaningful representations for the items in the graph.
- Convolutional Layers: The model has two convolutional layers: self.conv1 and self.conv2. These layers perform 1-dimensional convolutions on the input data. Each convolution is followed by a Rectified Linear Unit (ReLU) activation function (using F.relu). The convolutions help capture the local relationships and patterns in the data.

$$X^{(l+1)} = \text{ReLU}(\text{Conv1d}(X^{(l)})) \quad (2)$$

$X^{(l)}$: represents the input features at layer l.
$X^{(l+1)}$: represents the output features at the next layer, which is obtained after applying the convolution operation and ReLU activation.
Conv1d: represents the 1D convolutional operation.
ReLU: represents the Rectified Linear Unit activation function.

- Message Aggregation: After the convolutions, the model performs message aggregation from neighbouring nodes. The adjacency matrix is multiplied with the output of the convolutions (using a torch.matmul (adj, x)). This operation combines information from connected nodes to enrich the representation of each item.
- Mean Pooling: The model uses mean pooling to aggregate the messages from neighbours. The output of the message aggregation step is permuted, and then the mean is taken along the second dimension (using x.mean (dim=1)). This results in a single representation for each item in the graph.

$$X^{(\text{session})} = \text{mean}(X^{(L)}) \quad (3)$$

4.4. Recommendation Step

The model combines the graph convolutional capabilities of the GCN model with a fully connected layer and softmax activation (log-softmax) to generate recommendations.

- The fully connected layer: also known as the linear layer or dense layer is a fundamental component in neural networks. It performs a linear

transformation on the input data, mapping it to a different dimensional space. In the context of the provided model, the fully connected layer (self.fc) takes the item representations generated by the GCN component as input. It applies a linear transformation to these representations, mapping them to the number of output items. Its purpose is to learn and capture complex relationships between the input data and the desired output. It helps them make more complex predictions by combining and weighing the input features. Formally, the fully connected layer computes the following operation:

$$Z = \text{Linear}(X^{\text{mean}}) \quad \text{(Linear transformation)} \tag{4}$$

- The softmax function: is applied after the fully connected layer in the provided model. Its role is to convert the output of the fully connected layer into a probability distribution over the different output classes. In the context of the recommendation system, the softmax function is used to determine the likelihood or probability of each item being the next recommended item. It assigns higher probabilities to items that are more likely to be relevant or preferred by the user based on the given input.

$$Y = \text{Log-Softmax}(Z) \tag{5}$$

In the following, experimentation has been conducted on this GCN-Based model.

5. Expirements

This section outlines the essential components of the experimentation performed on the proposed approach.

5.1. Datasets

We assess the performance of our model on three commonly used transaction datasets: MovieLens 100k[1], MovieLens 1M[2], and YooChoose[3]. These datasets are publicly available and exhibit differences in terms of domain, size, and sparsity. Table I provides detailed statistics for these datasets:

5.2. Evaluation metrics

In our experiments, we use the following widely-used evaluation metrics: Accuracy, Recall@K and MRR@K (by default, we set K=20).

[1]Dataset source: https://grouplens.org/datasets/movielens/100k/
[2]Dataset source: https://grouplens.org/datasets/movielens/1m/
[3]Dataset source: https://s3-eu-west-1.amazonaws.com/yc-rdata/yoochoose-data.7z

Dataset	Rows	Train Sessions	Test Sessions	Avr-L
MovieLens 100k	99057	6530	1970	4.50
MovieLens 1M	994169	99360	26530	4.29
YooChoose1/64	371160	15919	2785	4.62

Table 1
Datasets Basic Informations.

1. **Accuracy**: Accuracy is a metric used to measure the performance of classification models. It represents the ratio of correctly predicted instances to the total instances in the dataset. The accuracy formula is:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{6}$$

2. **Recall@20**: measures the percentage of test cases in which the recommended items are correctly positioned within the top 20 in a ranking list. In this study, Recall@20 is utilized for all tests [10], as defined by this equation:

$$\text{Recall} = \frac{\text{Correctly recommended items}}{\text{Total useful recommended items}} \tag{7}$$

3. **MRR@20**: Mean Reciprocal Rank at N (20, in our case) assesses the quality of the ranking of recommendation results in an evaluation of the SBRS. If an item's rank ($rank_i$) exceeds N, the reciprocal rank is set to zero [10]. Generally, MRR is calculated as follows:

$$\text{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{rank}_i} \tag{8}$$

5.3. Baselines

Some baselines are used to evaluate our proposed GCN model, namely:

- **POP** [10]:
  This baseline model recommends the top-N ranked items based on their popularity in the training data. It serves as a straightforward and robust baseline, especially in specific domains.
- **S-POP** [10]:
  A variation of the baseline model that recommends the top-N most frequent items in both the entire training set and the current session.
- **Item-KNN** [9]:
  Determine the similarity between items A and B by first identifying all users who are directly associated with both items and then assessing the evaluation bias. Following this calculation, we obtain the top k most similar items as a result.

- **GRU4Rec** [13]:
  Utilizing recurrent neural networks, GRU4Rec is designed for session-based recommendations. It adopts a session-parallel mini-batch training process and employs ranking-based loss functions during training.
- **SR-GNN** [14]:
  In this model, separate session sequences are aggregated into a graph structure, and Graph Neural Networks (GNNs) are applied to generate latent item vectors. Each session is then represented using a traditional attention network.
- **GACOforRec** [3]:
  Built upon GCNs, this algorithm accounts for user preferences in the application scenario. By incorporating Convolutional LSTM (ConvLSTM) and Orthogonal LSTM (ON-LSTM), it handles long-term and stable user preferences while preserving preference hierarchy.
- **AUTOMATE** [10]:
  Keyed on the ARMAConv layer, AUTOMATE combines long-term preferences with current session interests to obtain graph transfer signals, resulting in personalized recommendations.

5.4. Training and Testing

In this experiment, the following implementation parameters are used:

- **Hyper parameters** At this part we used the hyperparameters that were explored and optimized after using the grid search algorithm including the hidden dimension size, learning rate, and the number of epochs. These hyperparameters have a significant impact on the model's ability to capture complex patterns in the session data and make accurate predictions.
- **Loss function** The *NLLLoss* (Negative Log Likelihood Loss) is a commonly used loss function in classification tasks. It measures the negative log-likelihood of the predicted probabilities for the correct class. Formally, the *NLLLoss* is calculated as follows:

$$\mathrm{N}LLLoss = -\log(P(\text{correct\_class})) \quad (9)$$

- **Optimizer** The Adam optimizer improves neural network training by dynamically adapting learning rates using gradient moments. It combines adaptive learning rates with momentum, facilitating faster convergence and managing varying parameter magnitudes. Momentum enhances learning by accumulating gradients.

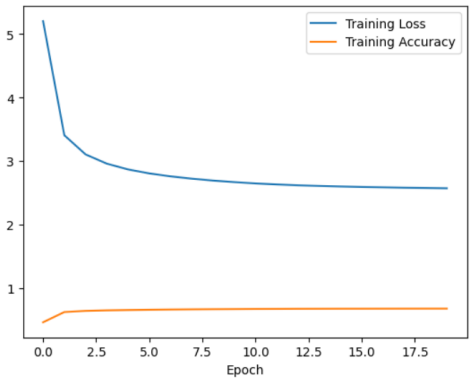

Figure 3: Accuracy and Loss Train on Movielens 1M Dataset.

6. Results and Discussion

Table II shows the results we obtained after applying the same Hyperprameters and loos function and optimizer on the three datasets:

Dataset	Train Accuracy/Loss	Test Accuracy/Loss	time
MovieLens 100k	76.43% / 1.4574	71.97% / 2.0935	2h
MovieLens 1M	67.66% / 2.5720	82.92% / 1.4095	12h
YooChoose1/64	60.01% / 1.4984	11.16% / 4.2414	3h

Table 2
The accuracy and loss results.

In Figures 3 and 4, we present the accuracy and loss during training on the datasets Movielens 1M and 100K. The training loss gradually decreases, indicating that the model is improving its predictions, while the training accuracy increases as the model becomes more accurate in its recommendations. This visualization provides insights into the model's learning process and its convergence over training epochs. The smooth curves observed in these figures are indicative of a stable training process. In machine learning, a smooth training curve suggests that the model is gradually converging to a solution rather than exhibiting erratic behaviour. This is generally a positive sign, indicating that the model is learning effectively without large fluctuations in performance. The stability can be attributed to several factors, including the choice of optimization algorithm (Adam in this case), a suitable learning rate, and well-behaved training data. It is crucial to note that while smooth curves are desirable, they should be interpreted alongside performance metrics to ensure the model is learning meaningful patterns from the data.

Model Class	Methods	MovieLens 1M		YooChoose 1/64		MovieLens 100k	
		Recall@20	MRR@20	Recall@20	MRR@20	Recall@20	MRR@20
Standard Baseline	POP	0.0646	0.0133	0.0671	0.001	0.1034	0.0209
	S-POP	0.0634	0.0132	0.3044	0.002	0.0776	0.0166
Traditional	Item Knn	0.0016	0.0014	0.5660	0.003	0.0045	0.0033
Neural Network	GRU4Rec	/	0.3041	0.6064	0.2289	/	/
	SR-GNN	/	0.3683	0.7003	0.3008	/	/
	GACOforRec	/	/	0.6879	0.2938	/	/
	AUTOMATE	/	/	0.7015	0.3072	/	/
Our Model	GCN-based Model	0.7259	0.2699	0.5154	0.1396	0.44	0.19

Table 3
Performance Comparison with Some Baselines.

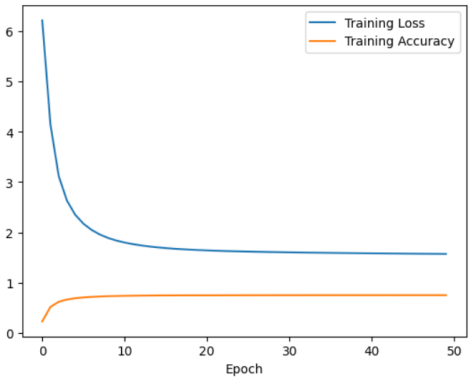

Figure 4: Accuracy and Loss Train on Movielens 100K Dataset.

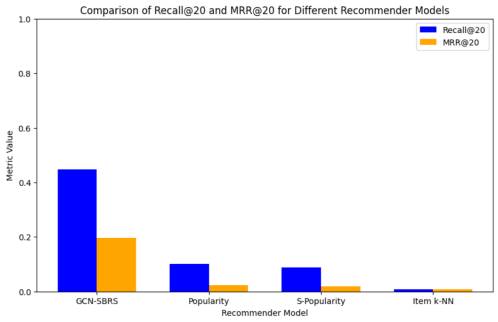

Figure 5: Comparison of MRR@20 and Recall@20 for our model and the baselines with Movielens 100k.

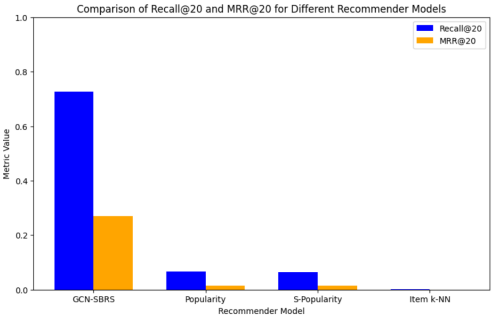

Figure 6: Comparison of MRR@20 and Recall@20 for our model and the baselines with Movielens 1M.

6.1. Comparing Results with Baselines

Table III presents a comparative analysis of our model with different recommender system models and baselines across various evaluation metrics for three distinct datasets: MovieLens 1M, YooChoose 1/64, and MovieLens 100k.

As shown in Table III, our model performs very well with both Movielens 100k and 1M even better than the other models, but YooChoose gave us bad results and that because of the differences between the two datasets MovieLens and YooChoose explained in these two points:

- Data Distribution: The Movielens and YooChoose datasets likely have different data distributions. These differences could affect how well our model generalizes from one to another. Models trained on one dataset may not perform as well on a different dataset if the underlying data patterns are dissimilar.
- Feature Engineering: The features (attributes) in the two datasets may have distinct characteristics. Our model might be well-suited to capturing the patterns present in the features of the Movielens dataset but struggle to do so with the features in the YooChoose. dataset.

As future work, we aim to improve these results and test our model on other popular data sets.

Furthermore, Figures 5 and 6 provide a visual representation of the model's performance with Movielens 100k and 1M datasets, respectively. These figures illustrate the model's performance in terms of MRR@20 and Recall@20 when compared to other baseline models.

7. Conclusion

In this work, the focus was on exploring the application of Graph Convolutional Networks (GCN) to enable Session-based Recommender Systems (SBRS). The research was conducted in three points. At first, we have introduced the concept of session-based recommender systems. Then, we have provided an in-depth understanding of Graph Convolutional Networks, showcasing their ability to capture graph-structured data. Lastly, we presented the design and implementation of an SBRS model utilizing GCN, demonstrating its effectiveness in generating accurate and personalized recommendations based on users' session history.

The findings of this research highlight the potential of leveraging GCN to enable session-based recommender systems and improve their performance. Our future work will be to ameliorate these results and to test them with more baselines on other popular datasets. Also, Using other CNN architectures including hypergraphs GCN to enhance SBRSss is among our nearest research.

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Trans. on Knowl. and Data Eng. 17 (2005) 734–749. URL: https://doi.org/10.1109/TKDE.2005.99. doi:10.1109/TKDE.2005.99.

[2] T. Silveira, M. Zhang, X. Lin, Y. Liu, S. Ma, How good your recommender system is? a survey on evaluations in recommendation, International Journal of Machine Learning and Cybernetics 10 (2019). doi:10.1007/s13042-017-0762-9.

[3] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, D. Lian, A survey on session-based recommender systems, ACM Comput. Surv. 54 (2021). URL: https://doi.org/10.1145/3465401. doi:10.1145/3465401.

[4] A. ÖZCAN, Applying graph convolution networks to recommender systems based on graph topology, DÜMF Mühendislik Dergisi (2022). doi:10.24012/dumf.1081137.

[5] R. Qiu, Z. Huang, T. Chen, H. Yin, Exploiting positional information for session-based recommendation, ACM Trans. Inf. Syst. 40 (2021). URL: https://doi.org/10.1145/3473339. doi:10.1145/3473339.

[6] T. N. Kipf, M. Welling, Semi-Supervised Classification with Graph Convolutional Networks, in: Proceedings of the 5th International Conference on Learning Representations, ICLR '17, 2017. URL: https://openreview.net/forum?id=SJU4ayYgl.

[7] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P. S. Yu, A comprehensive survey on graph neural networks, IEEE Transactions on Neural Networks and Learning Systems 32 (2021) 4–24. doi:10.1109/TNNLS.2020.2978386.

[8] D. K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory, Applied and Computational Harmonic Analysis 30 (2011) 129–150. URL: https://www.sciencedirect.com/science/article/pii/S1063520310000552. doi:https://doi.org/10.1016/j.acha.2010.04.005.

[9] M. Zhang, Z. Yang, Gacoforrec: Session-based graph convolutional neural networks recommendation model, IEEE Access 7 (2019) 114077–114085. doi:10.1109/ACCESS.2019.2936461.

[10] H. Wang, G. Xiao, N. Han, H. Chen, Session-based graph convolutional arma filter recommendation model, IEEE Access 8 (2020) 62053–62064. doi:10.1109/ACCESS.2020.2984039.

[11] Self-supervised hypergraph convolutional networks for session-based recommendation, Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 4503–4511. URL: https://doi.org/10.1609%2Faaai.v35i5.16578. doi:10.1609/aaai.v35i5.16578.

[12] C. Ding, Z. Zhao, C. Li, Y. Yu, Q. Zeng, Session-based recommendation with hypergraph convolutional networks and sequential information embeddings, Expert Systems with Applications 223 (2023) 119875. URL: https://www.sciencedirect.com/science/article/pii/S0957417423003767. doi:https://doi.org/10.1016/j.eswa.2023.119875.

[13] Y. K. Tan, X. Xu, Y. Liu, Improved recurrent neural networks for session-based recommendations, in: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS 2016, Association for Computing Machinery, New York, NY, USA, 2016, p. 17–22. URL: https://doi.org/10.1145/2988450.2988452. doi:10.1145/2988450.2988452.

[14] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, volume 33, 2019. doi:10.1609/aaai.v33i01.3301346.