E-Commerce Recommendation Algorithm based on Graph Neural Network

Lingxiao Zhu *

Faculty of Engineering, The University of Hong Kong, Hong Kong, China

* Corresponding author email: u3582859@connect.hku.hk

Abstract. While recommender system is becoming an increasingly essential component in e-commerce websites, although previous models, which directly calculate the similarity of user/item history record, has obtained evidence of effectiveness, recommendations based solely on users' current sequence of actions, when user identity and history preference are not present, has been a popular area due to the growing privacy concerns. This paper demonstrates a model using a graph neural network, which takes the user's sequence of purchasing events as input and constructs a graph derived from it, to make the prediction of the most likely subsequent product that the customer may purchase and make personalized recommendations by the combination of session preference and user's current interest. Experiments on a real-world e-commerce purchasing event dataset and analysis are carried out to test the model's performance, as well as how the length of sequence may affect the model preference. The result shows that the model performance has attained a local peak on the dataset used.

Keywords: Graph Neural Network; E-commerce Recommendation; Node Prediction.

1. Introduction

As a result of the enormous increase in online information, recommender systems have become vital to many organizations. Specifically, e-commerce websites depend on efficient recommendation systems to attract customers' attention to their favored products, encouraging consumers to purchase them. Consumers depend on recommendation systems to relieve the problem of information saturation and help them identify the products that meet their requirements among the immense sea of available items [1, 2].

Recommendation systems have been popular for decades, and several implementations have been proposed, from the item-based neighborhood approach, which takes advantage of users' interaction history to promote products that are similar to those they have engaged with in the past, to deep learning models, which have evolved into the predominant approach for recommendation system in both scientific research and commercial implementations because of their strong ability for collecting non-linear and non-trivial user-item linkages and their swift integration of a variety of data sources. [3, 4].

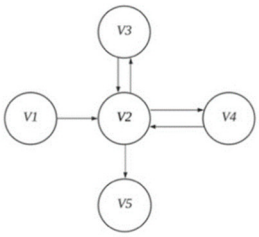
One category of these deep learning algorithms, newly proposed in recent years, is graph-based approaches, which examine the information in recommender systems from a graph-based perspective. When user identities are known, the sequential patterns in the user's historical interactions can be utilized to improve the accuracy of recommendations by a heterogeneous graph neural network [5]. However, due to the fact that user identities are often challenging to track, and could possibly cause privacy concerns, while user identities are not known and only historical behavior in chronological order is present, the heterogeneous method no longer works, and a more generalized method needs to be used, focusing on the sequential purchasing sessions that users had to construct graphs, which is the session-based neural graph network, for predicting the next item that a user may buy simply from the current sequential session, disregarding the user's buying history [6].

2. Methodology

In this section, how the graph is constructed, and the framework of the Session-based graph neural network are introduced.

2.1 Graph Construction

For each session s, all unique items purchased, with features of price, and quantity, are denoted as the vertices set $V_s = \{v_1, v_2, \ldots, v_n\}$. Thus, a session can be represented as a list of items purchased $s = [v_{s,1}, v_{s,2}, \ldots, v_{s,n}]$, where $v_{s,i}$, denotes the *i*th item bought after the (*i-1*)th item was purchased in the sessions. Based on this chronological order, directed edges are added between the two items purchased in the session sequence, based on the time that the items are purchased. Formally, an edge $(v_{s,i}, v_{s,i+1})$ means that item $v_{s,i+1}$ is purchased after a user purchased $v_{s,i}$ in the session s. As there exits the possibility that an item may occur more than one time in the same session, each edge is assigned a weight, which is equal to the quotient of the times that an edge appears and the outdegree of its originating node, yielding the incoming and outgoing connecting matrix $\mathbf{A_s}$ for session s. In this way, a graph $G_s = (V_s, E_s)$ is constructed. The node vector, $\mathbf{v_{s,}}_i$, is defined as the row of item *i* from the connecting matrix $\mathbf{A_s}$ of session s. **Figure. 1** depicts an example graph together with its connecting matrix.



	1	2	3	4	5	1	2	3	4	5
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1/3	1/3	1/3	0	0	1/2	1/2	0
3	0	1	0	0	0	0	1	0	0	0
4	0	1	0	0	0	0	1	0	0	0
5	0	0	0	0	0	0	1	0	0	0
	A_{out}					A_{in}				

Figure 1. An example session graph and its connecting and its connecting matrix $\mathbf{A_s} = [\mathbf{A_{in}}, \mathbf{A_{out}}]$

2.2 Session-based Graph Neural Network Framework

As the average length of sessions of purchasing events is not long, the graph neural network can easily take the whole graph as input. Meanwhile, due to graph neural network's good ability to extract features from session graphs while taking into account dense node connections, they are particularly suited for session-based recommendations [6]. The recurrent learning process of propagation is given as follows:

$$\mathbf{a}_{s,i}^{(t)} = \mathbf{v}_{s,i:}\,[\mathbf{v}_1^{(t-1)}, \mathbf{v}_2^{(t-1)}, \ldots, \mathbf{v}_n^{(t-1)}]^{\mathrm{T}}\,\mathbf{W}_{\boldsymbol{a}} + \mathbf{b}\,, \tag{1}$$

$$\mathbf{z}_{s,i}^{(t)} = \sigma\left(\mathbf{W}_z \mathbf{a}_{s,i}^{(t)} + \mathbf{U}_z \mathbf{v}_i{}^{(t-1)}\right), \tag{2}$$

$$\mathbf{r}_{s,i}^{(t)} = \sigma\left(\mathbf{W}_r \mathbf{a}_{s,i}^{(t)} + \mathbf{U}_r \mathbf{v}_i{}^{(t-1)}\right), \tag{3}$$

$$\widetilde{\mathbf{v}_i^{(t)}} = \tanh\left(\mathbf{W}_v\, \mathbf{a}_{s,i}^{(t)} + \mathbf{U}_v\left(r_{s,i}^{(t)} \odot \mathbf{v}_i{}^{(t-1)}\right)\right), \tag{4}$$

$$\mathbf{v}_i{}^{(t)} = \left(1 - \mathbf{z}_{s,i}^{(t)}\right) \odot \mathbf{v}_i{}^{(t-1)} + \mathbf{z}_{s,i}^{(t)} \odot \widetilde{\mathbf{v}_i^{(t)}}\,, \tag{5}$$

where $\mathbf{z}_{s,i}^{(t)}$ and $\mathbf{r}_{s,i}^{(t)}$ denotes the update and reset gates, which determine the reservation of information and parameters, $\mathbf{W}_{\boldsymbol{a}}$ and $\mathbf{b}$ are the weight and bias in the linear transformation from the list of node vectors $[\mathbf{v}_1^{(t-1)}, \mathbf{v}_2^{(t-1)}, \ldots, \mathbf{v}_n^{(t-1)}]$ to the activation function $\mathbf{a}_{s,i}^{(t)}$, which transmits information between graph nodes through incoming and outgoing edges with edge-dependent parameters. The logistic sigmoid function is denoted as σ, the hyperbolic tangent function is denoted as tanh, and the element-wise multiplication between vectors or matrices is denoted as $\odot$. After obtaining all the node vectors in the sequence, the session-level representation vector **s** can be defined. Taking into account that the features of nodes may have different levels of importance, to aggregate

the influence of the node vectors on the session preference $\mathbf{s}_1$, a soft-attention mechanism is introduced [7]:

$$\mathbf{s}_1 = \sum_{i=1}^{n} ((\mathbf{x}^{\mathbb{T}} \sigma(\mathbf{M_1 v_n} + \mathbf{M_2 v}_i + \mathbf{c}) \times \mathbf{v}_i) \tag{6}$$

where $\mathbf{x} \in \mathbf{R}^d$ and $\mathbf{M_1}, \mathbf{M_2} \in \mathbf{R}^{d \times d}$, determining the weights of information carried in the nodes. As the target of session-based recommendation is to make the prediction of subsequent product that a customer may purchase, which is, to be more specific, the item $v_{s,n+1}$, of the sequence, the session-level preference vector $\mathbf{s}$ can then be formulated by the combination of $\mathbf{s}_1$ and the last node vector in the sequence, $\mathbf{v_n}$:

$$\mathbf{s} = \mathbf{M_3} [\mathbf{v_n}; \mathbf{s_1}] \tag{7}$$

where $\mathbf{M_3} \in \mathbf{R}^{d \times 2d}$, reducing the dimension of the concatenation of $\mathbf{v_n}$ and $\mathbf{s}_1$, $\in \mathbf{R}^{2d}$, into dimension of $\mathbf{R}^d$. Then each item gets its node score by multiplying the session representation vector s and its own node vector:

$$\text{score}_i = \mathbf{s}^{\mathrm{T}} \mathbf{v}_i \tag{8}$$

$$\mathbf{z_s} = [\text{score}_1, \text{score}_2, \ldots, \text{score}_n] \tag{9}$$

where $\mathbf{z}_s \in \mathbf{R}^d$ denotes the vector containing all the node scores. After that $\mathbf{z}_s$ is passed through a softmax layer to have the possibilities $\mathbf{p}$ that each item may appear next in the session s. An example of the workflow is shown in **Figure.2**.

$$\mathbf{p} = \text{softmax}(\mathbf{z_s}) \tag{10}$$

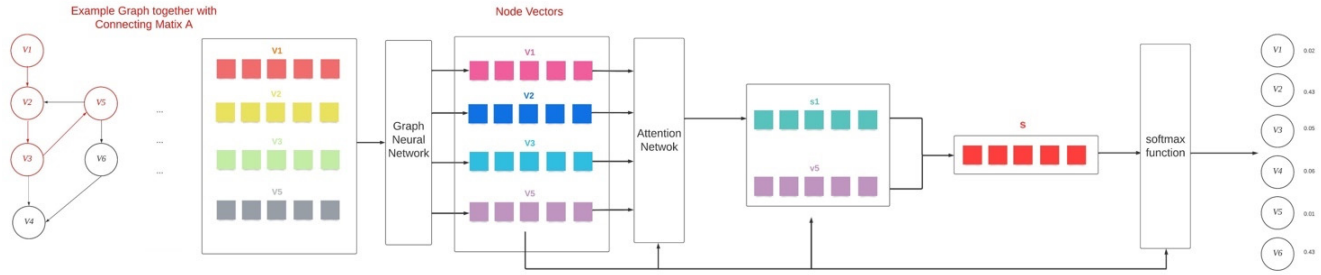


Figure 2. Example of the workflow of the proposed method

3. Experiment

In this section, the datasets used in the experiment, and the processes of training and testing are introduced.

3.1 Dataset and Preprocessing

The proposed method is evaluated by the dataset from RecSys Challenge 2015 (http:// 2015. recsyschallenge.com/challege.html), which contains user purchasing events from an e-commerce website within six months. In the cleaning of data, sessions with lengths smaller than 2, and items purchased fewer than 5 times are filtered out. The remaining part contains 897,305 purchasing events, 276,303 sessions, and 8,937 items, with an average session length of 3.2629. Also, item features, such as price and quantity, are kept for a more personalized recommendation.

3.2 Training and Testing

The purchasing events that happened in the last week is extracted from the dataset as the test set, composed of 12,862 testing sessions. A random 10% of the test set is picked as the validation set. For an input sequence s = [$v_{s,1}$, $v_{s,2}$, …, $v_{s,n}$], labels are added to the purchasing event $v_{s,i}$ as the sequence of purchasing event before it. To be more specific, the sequence of [$v_{s,1}$] the label generated for $v_{s,2}$, and the sequence of [$v_{s,1}$, $v_{s,2}$, …, $v_{s,n-1}$] the label generated for $v_{s,n}$.

The cross-entropy of the recommendation made by the model and the actual subsequent item purchased defines the model's loss function. Then the model is trained utilizing the Back-Propagation Through Time (BPTT) algorithm, as it eliminates the requirement for parameter constraints to guarantee convergence [8]. The weights in the model are updated using mini-batch Adam optimizer, with assigned values of 0.003 to the initial learning rate, 0.13 every 4 epochs to the decay rate, 120 to the batch size, and 205 to the LL2 penalty, after adjustment so that the model has the best performance [9].

4. Result and Analysis

In this section, the evaluation criteria, and the model performance in the experiment on the used dataset, and the analysis of how session length would impact the model performance are introduced.

4.1 Evaluation Criteria

Two measures are used to assess the model's performance: precision (recall@20 for brevity) and mean reciprocal rank (MRR@20 for brevity). Recall@20 measures the percentage of test cases in which the correctly recommended item appears amongst the first twenty of the recommendation lists, while disregarding the targeted item's actual rank. MRR@20 is the mean of the targeted items' rank in the recommendation list, and if the rank surpasses 20, the rank is reset to zero. The targeted item's rank is within the consideration of MRR@20, which is crucial in e-commerce recommendations since if the targeted item is not on the top of the recommendation list, users' attention may not be attracted, and the possibility that a purchasing event is missed increases significantly [10].

4.2 Model Performance

Figure. 3 exhibits the model performance. Despite recall@20 and MRR@20, total loss during the training process is included. The total loss of the model starts to converge and fluctuate around 6560 after 12 epochs of training, and the same trend occurs in the patterns of the two evaluation metrics: the recall@20 of the model reached an ultimate value of 65.8321, and the MRR@20 of the model reached an ultimate value of 25.2033. It can be concluded that the performance of the model has attained a local peak.

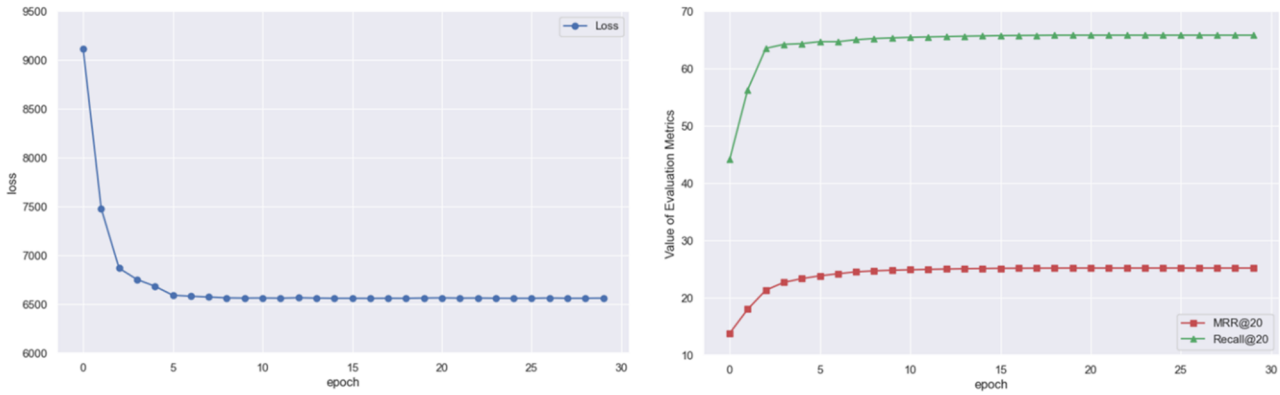


Figure 3. Performance of the Model

4.3 Analysis of Lengths of Sequences

Then the analysis of the effects that sequence length may have on the performance of the model is carried out. Since the average length of session sequences of the dataset is 3.2629, taking the nearest integer, 3, the dataset is split into two partitions: short session sequences, which have lengths smaller than 3, and long session sequences, which have lengths longer than 3. The short partition contains 193,057 sessions, consisting 69.87% of the whole dataset, and the long partition contains 83,246 sessions, consisting 30.13% of the dataset. **Figure.4** depicts the model performance. The result indicates that the model performed better, in terms of both assessment metrics, on the long sessions,

while only to a tiny extent. Moreover, the performance of the model only varies to a small extent as the lengths of sequences change, which shows good adaptability and stability.

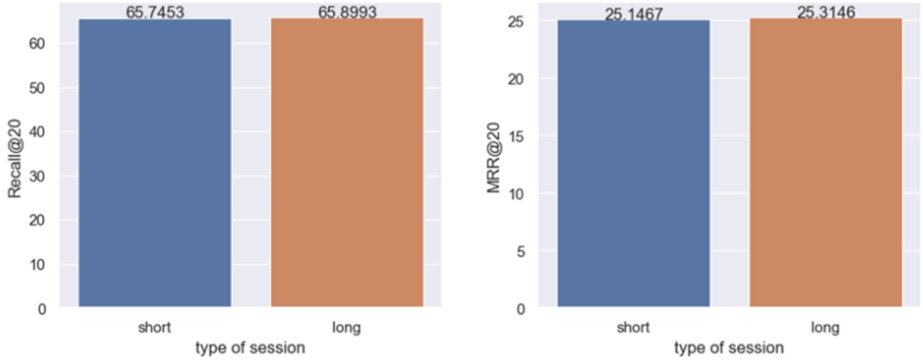


Figure 4. Performance of Model on Short and Long Session Sequences

5. Conclusion

A model for recommendation derived from sessions of purchasing events is examined in this paper by experiments using a real-world dataset, as effective recommendation systems are getting more and more priority in e-commerce when user identity and history preference are not present. After adjusting the parameters, the result showed that the performance of the model has reached a local peak on the dataset used. Moreover, an analysis of how session length would impact the model performance is carried out. The result turned out that model performance is not influenced by the length of sessions significantly, showing a good ability for generalization and adaptability, while the model is better at long sessions, so the future focus would be placed on better performance of the model on the short-length sessions.

References

[1] WU S, SUN F, ZHANG W, et al. Graph neural networks in recommender systems: A survey[J]. ACM Computing Surveys, 2022, 55(5): 1–37.

[2] CHEN T, WONG R C-W. Handling information loss of graph neural networks for session-based recommendation[J]. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020.

[3] BELL R M, KOREN Y. Scalable collaborative filtering with jointly derived neighbourhood interpolation weights[J]. Seventh IEEE International Conference on Data Mining (ICDM 2007), 2007.

[4] COVINGTON P, ADAMS J, SARGIN E, et al. Deep Neural Networks for YouTube recommendations[J]. Proceedings of the 10th ACM Conference on Recommender Systems, 2016.

[5] PANG Y, WU L, SHEN Q, et al. Heterogeneous global graph neural networks for personalized session-based recommendation[J]. Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022.

[6] WU S, TANG Y, ZHU Y, et al. Session-based recommendation with Graph Neural Networks[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33(01): 346–353.

[7] LIU Q, ZENG Y, MOKHOSI R, et al. Stamp[J]. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018.

[8] LI Y, TARLOW D, BROCKSCHMIDT M, et al. Gated graph sequence neural networks [EB/OL]. arXiv.org, 2017-09-22. (2017-09-22) [2022-12-30]. https://doi.org/10.48550/arXiv.1511.05493.

[9] LI J, REN P, CHEN Z, et al. Neural attentive session-based recommendation[J]. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017.

[10] HIDASI B, KARATZOGLOU A, BALTRUNAS L, et al. Session-based recommendations with Recurrent Neural Networks [EB/OL]. arXiv.org, 2016-03-29. (2016-03-29) [2022-12-30]. https:// doi.org/ 10.48550/arXiv.1511.06939.