Utilizing Graph Neural Network to Predict Next Items in Large-sized Session-based Recommendation Data

Tianqi Wang*
CSE Department, University at Buffalo
Buffalo, NY, USA

Zhongfen Deng†
CS Department, University of Illinois at Chicago
Chicago, IL, USA

Hou-Wei Chou
Rakuten Institute of Technology
Boston, MA, USA

Lei Chen‡
Rakuten Institute of Technology
Boston, MA, USA
lei.a.chen@rakuten.com

Wei-Te Chen
Rakuten Institute of Technology
Boston, MA, USA

ABSTRACT

Users now spend increasingly longer online time than before. This new trend makes session-based recommendation (SBRec) play an important role in fulfilling users' media/product consumption intentions. In the data challenge organized by SIGIR'21 e-Commerce workshop, Coveo provides a brand new large-scaled industry data to promote more active research on the SBRec topic. We tested several leading methods ranging from simple vector-similarity search to modern solutions using deep learning on this new data. We found that deep learning methods are more accurate on predicting the next items than the conventional methods. Using a graph neural network (GNN) to model items' dependency relationships generates the best performance in our experiments (Mean Reciprocal Rank (MRR) to be 0.165 on the leader board 2).

ACM Reference Format:

Tianqi Wang, Zhongfen Deng, Hou-Wei Chou, Lei Chen, and Wei-Te Chen. 2021. Utilizing Graph Neural Network to Predict Next Items in Large-sized Session-based Recommendation Data. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR eCom'21).* ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

Session-based Recommendation (SBRec) refers to a special type of recommendation system (RS) that uses a session, i.e., a sequence of actions on various items, as an input. Different to the general purpose RS that is based on global user-item relations, the SBRec provides a context based on input sessions to enable the RS to utilize recent users' preferences for making more accurate recommendations. In some RS tasks, e.g., recommendation of next-to-play media files in either Spotify or Netflix platforms, SBREc has been become a key technology. In e-commerce, with more users who repetitively purchase, online shops now can access massive shopping sessions and are at tipping point to utilize the SBRec technique to better support users' future shopping.

As shown in a recent survey paper on the SBRec [14], in the past decade, there have been substantial increases in related publications. In this paper, we report on our work[1] when participating the SBRec research topic organized by SIGIR'21 e-Commerce workshop data challenge [2].

Using the real shopping data provided by Coveo, which will be described in more detail in Section 3, we compared several SBRec methods ranging from conventional ones to more recent deep learning (DL) based methods. We found that Graph Neural Network (GNN) based SRGNN [16] outperforms other DL methods. Later, using the provided metadata dense vectors, we further explored the benefit of adding items' metadata in the SBRec task. Coveo dataset is directly from industry practices and exposes several real challenges, e.g., the session context could be empty to give the SBRec system a real "cold start" stress-test and metadata could be missing as a high rate. Clearly, these real challenges brought a closer-to-reality scenario to enable us to measure several SBRec methods from an industry usage perspective.

2 RELATED WORKS

SBRec is solved by both conventional machine learning methods and more increasingly by deep learning (DL) methods. Regarding the conventional SBRec, one effective method is based on K Nearest Neighbour (KNN). Treating all items appearing in sessions to be "words", prod2vec [5] computes the items' dense vector representations. Then, based on a similarity search, e.g., cosine similarity between vectors, the next item can be recommended.

As a widely used DL approach to process sequence data, recurrent neural network (RNN) has been used in SBRec to encode session contexts. [11] is built on Gated Recurrent Unit (GRU) RNN to encode session context c. The hidden state from the last timestamp serves as the entire session's representation to predict the next item. Later, following the successful application of Transformer [15] in the natural language processing (NLP) area on effectively and

*Equal contribution; intern at RIT Boston
†Equal contribution; intern at RIT Boston
‡Correspence author

SIGIR eCom'21, July 15, 2021, Virtual Event, Montreal, Canada

[1] Our team name is beantown
[2] https://sigir-ecom.github.io/data-task.html

efficiently processing sequence data, **self-attention** (SA) mechanism was used in building SBRec systems. [6] proposed using a self-attention based sequential model (SASRec) to combine the benefits from both RNN based model and Markov Chains (MCs) model's advantages. When testing on several e-commerce data sets, SASRec shows its higher performance than ordinary RNN based methods. Another direction of replacing RNN, which is limited on its running speed due to the difficulty of paralleling recurrent operations, is using Convolutional Networks (CNNs). Inspired by the dynamic CNN architecture from both NLP and Computer Vision (CV) research areas, [12] proposed DynamicRec model that can learn optimal CNN kernel sizes.

Following the trend of pretraining models using self-supervised learning, e.g., BERT [4], a model **Prod2BERT** [2] has been proposed and showed encouraging performance on the next-item prediction (NEP) task. [16] used Graph Neural Network (GNN) to obtain item representations. By generating a graph connecting all items from historical sessions, GNN can be viewed as going beyond a simple left-to-right one directional sequence to better capture dependencies among items that are treated as nodes in a graph.

Beyond item-item level dynamic information, content features on items also play important roles in the SBRec research. For example, when items do not frequently appear in historical sessions, the item level dependence pattern may not be reliably estimated. In this case, using items' metadata, e.g., categories, description texts, and even visual patterns, can improve the representations. [17] extended the SASRec [6] to the FSDA model by adding one more self-attention branch to capture feature level changes among items. The FSDA model shows its superior performance than the models purely using item cooccurrences. [13] extended the SASRec in another way. It used a 1D CNN model based on actions (like shopping) and items' categories to explicitly model users' local intentions. The obtained user intentions were used together with long-range item preferences learned by the SASRec to provide more personalized recommendations.

3 TASK AND DATA INSIGHTS

The Coveo released a session-based shopping data set consisting of the following three files:

- **browsing interactions**: More than four million sessions containing fine-grained shopping behaviors are provided. Each row reports a separate interaction that could be a page view or a product interaction, i.e., detailing, adding-into-cart, and buying. Product SKUs stored in a hash format, action timestamps, and hashed URL are provided.
- **search interactions**: More than $800k$ search-based interactions are provided. Each row reports on the query (represented in a dense vector), and product SKUs have been shown to and are clicked by users.
- **product metadata**: From the entire dataset including both train and test subsets, products' metadata, i.e., category labels, text descriptions, product images, and prices are provided. Category labels are represented by their hash codes and texts/images are represented in dense vectors computed by pre-trained models and dimension reduction techniques. Price information is provided as a 10-quantile integer.

According to Coveo, the shopping data was recorded from *Shop Z*, a mid-sized shop with an Alexa ranking between $25k$ and $200k$. Training and testing sets were sampled from disjoint but adjacent time periods. Compared with a set of public data sets for the SBRec research, the Coveo data provides more comprehensive behavior tracking and fine-grained details. Also, compared with other data sets with rich details, the data size has been dramatically increased about 10 times. Clearly, the introduction of the Coveo dataset will provide more boosting to the entire SBRec research area.

We did a set of exploration data analysis (EDAs) to better understand the challenge data. For example, Figure 1 depicts the sessions' length histogram and we can find that most of the sessions are very short. More than half of sessions have no more than three events and most of the sessions include no more than ten events. Figure 2

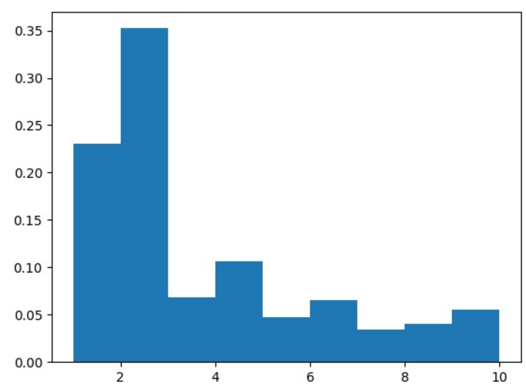

Figure 1: Histogram of session lengths. X-axis represents the length of sessions and Y-axis denotes the probability density.

depicts the items' frequency histogram. Note that a large portion of items only appears in the Coveo data with a low frequency. This impacts learning item representations based on their occurrences. Figure 3 shows a t-SNE plot of all of the items text descriptions that were provided as dense vectors in the training set. All items look evenly distributed on the embedding space and there has no clear subgroup pattern on items.

In the data challenge, two session-based tasks were tackled, i.e., **session-based recommendation task** and **cart-abandonment task**. We have been focusing on the first task and it requires to predict the next interaction between shoppers and products based on previous product interactions and search queries. Regarding evaluation, a session's starting portion was provided to challenge participants' models and future interactions are needed to be filled. The predicted interactions are evaluated using the two metrics. Focusing on the *immediate next item*, **Mean Reciprocal Rank** (MRR) is used. Focusing on *all subsequent items*, up to a maximum of 20 after the current event, **F1 score** from precision@20 and recall@20 is used.

4 MODEL

Given a sequence of products $p_0, p_1, p_2, \ldots, p_{n-1}$, the next item recommendation task is predicting the following product p_n. The

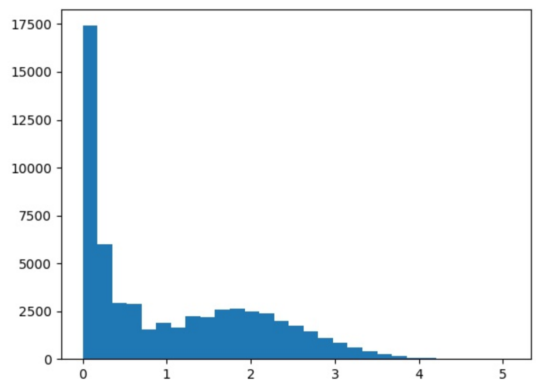

Figure 2: The histogram of the count of products. X-axis represents the frequency of the counts of products ($log10$ scale) and Y-axis represents the counts of products

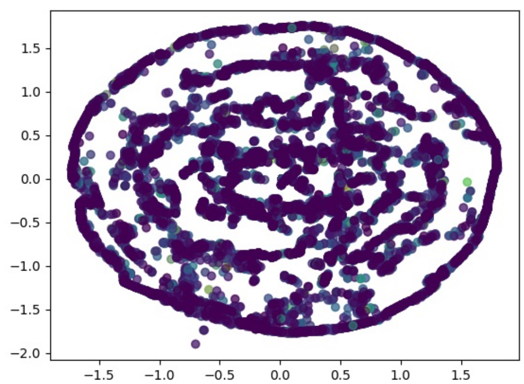

Figure 3: t-SNE plot of items' descriptions embedded as dense vectors. Colors are based on categories. No any clustering pattern shows among all of items.

Coveo data also provides the action sequence $a_0, ...a_{n-1}$ where a_i is one of four actions. Moreover, for some products, their corresponding category c_i, text description dense vector t_i, and image dense vector v_i are also provided to be available inputs. For example, approximately half of the products (37,822 out of 66,386) have a text description dense vector.

The following models are tried in this data challenge.

- **KNN**: following the word2vec [9] embedding training, prod2vec vectors for all items are first learned in a unsupervised way. Then, we average the embeddings of all items in a session to obtain the session representation and find its closest item to be p_n.
- **s-KNN**: this method is an extension of KNN in the SBRec situation. For a session $\mathbf{s}$, s-KNN first finds $N(\mathbf{s})$ similar sessions and calculate scores for candidate items in them. Then the candidate item with highest score is returned as p_n.
- **SASRec**: a Transformer is used to encode item sequence. As shown in Figure 4 top panel, self-attention layer is used to process sequence $p_0, p_1, \dots p_{i-1}$ and then a feed-forward network (FFN) is used to map encoding to next item p_i.
- **DynamicRec**: 1D Convolution on the temporal order with a varying kernel size is used to encode item sequence. For a product at $i-1$ timestamp, the kernel size K is estimated by a FFN. Then, the Conv operation with the K-sized kernel is applied from p_{i-K} to p_{i-1} to obtain the sequence encoding. At last, the sequence encoding is mapped to the next item using a FFN. More details can be found in Figure 4 bottom panel
- **SRGNN**: A graph is used to map global relations among all products in the training sessions. As shown in Figure 5, the items in the three separate sessions are placed in a graph. Then, the standard GNN computation, e.g., graph convolution network (GCN) [7] is used to obtain all items (nodes in the graph) embedding vectors. For a session, the last timestamp's product embedding serves as a local representation s_l and an global representation s_g is computed from all timestamps. These two representations are concatenated and are converted into the entire session's representation s. At the final step, the similarities between s and the learned product vectors are computed to find the most similar product to the next item.

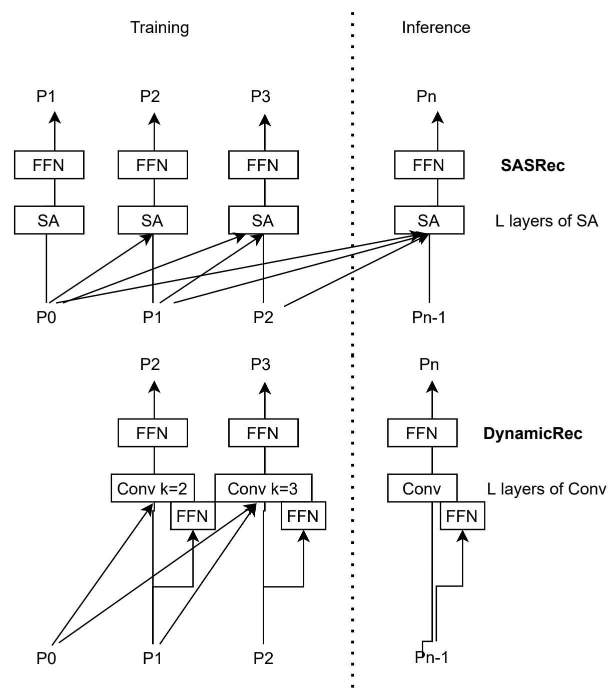

Figure 4: Session-based recommendation solved by SASRec model using self-attention (SA) layers to encode product sequence and by DynamicRec model using convolution (Conv) layers with varying kernel sizes K that is estimated from the last product.

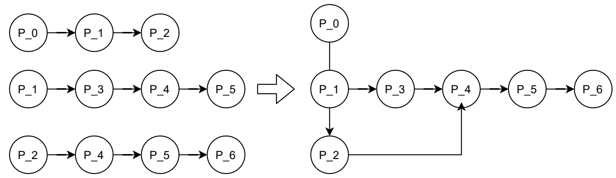

Figure 5: SRGNN model places all of products appearing in various sessions into a graph and then uses GNN to encode all nodes to obtain items' embedding vectors.

5 EXPERIMENT

5.1 Setups

The Coveo data provided is massive in records. To process it, we started from cleaning browsing events by removing page-view events that are not associated with any product. At the same time, we also cleaned the searching events by removing the events without any product click. After these cleaning steps, we processed both records by the following steps: (1) grouping all events based on session IDs, (2) sorting the events in the same session by the ascending order of the server timestamp, and (3) extracting the product SKU sequence by considering all clicked products for a search event and keeping the clicked order in the sequence. To mimic the official test data of phase 1, we randomly selected 96,108 sessions from the training set as our own evaluation set (approximately 3% of all sessions) and the remaining sessions are used for building SBRec models. Among the selected 96, 108 test sessions, 48, 730 of them only contain one product, we further removed those from our own evaluation set. For the rest of the selected sessions, we randomly split each session into two parts, the first part is used as input, and the second part is used as the ground truth of the corresponding session. Doing so, we can rule out the possibility that interactions occurred later could be used to wrongly predict current items. Finally, our own-eval set contains 47, 378 sessions. Note that the input of our own-eval set contains sessions with at least one context product. This is different to the official test data in the leader board (LB). The LB data (phase 2) contains about 1/3 (104,492 out of 332,247) sessions without any context product to reflect the fact that an industry SBRec model needs work in a real "cold-start" condition and the performance in such a tough scenario plays an important role in real business.

Both KNN and s-KNN apply the dense item embeddings to find the neighbors where distances are measured by the cosine similarity. Those embeddings are learned using prod2vec method via Gensim [10]. For s-KNN, for each test session, we first find 200 most similar sessions in the training set and obtain a set of candidate items including all the items in the 200 most similar sessions. Then, for each candidate item, we calculate its score by summing the similarity between the sessions it occurs and the test session.

The DynamicRec model is based on the code released with the paper [3], it adopts the product embeddings generated by prod2vec as initial embeddings (its size is 50) instead of learning from scratch. Two dynamic convolutional layers are utilized in DynamicRec and sessions with length less than 5 are removed from the training set. And the maximum length of sessions is set to 30. Both SASRec and SRGNN codes are from the Rec Bole package [18] [4]. For SASRec, the maximum sequence length is 50, the layers of self-attention (SA) is 2 and embedding size is 64. For SRGNN, the embedding size is 64. When facing zero-length session in the test set, we simply fill the most frequent item in the train set based on the maximal likelihood principle. Clearly, there are many other solutions to better handle this situation, e.g., [1]. However, we have no time to explore other "advanced" methods in the data challenge and will put this to be our future focus.

5.2 Result

Table 1 reports on several SBRec models' experiment results. From conventional ML models, we found that session specific s-KNN model does not show a higher MRR value than the simple item-KNN model in both our own-eval set and the LB set. Note that our own-eval set does not contain sessions containing zero context, therefore, the MRR values are much higher than the values obtained on the LB data. One of the possible explanations to the reason that s-KNN is worse than KNN is that the sessions are quite short and sequence patterns used in s-KNN may not be sufficient. Comparing the DL based methods with KNN method, we can find all DL models show significantly higher MRR on our own-eval set. However, on the official LB set, the DynamicRec model does not show a noticeable gain than the item-KNN model. In a contrast, the SASRec model shows about 0.03 MRR increase. Among all of the three DL methods, the SRGNN model shows the highest MRR values. This may be due to its proper usage of all items in all sessions.

Model	MRR on own-eval	MRR on LB2
KNN	0.326	0.115
s-KNN	0.285	0.068
DynamicRec	0.37	0.116‡
SASRec	0.42	0.144
SRGNN	0.46	0.165

Table 1: Several session-based recommendation models were compared to predict immediate next items. MRR metrics on our own-eval set and the official LB2 are reported. ‡: this is our team's final submission to the challenge before the deadline.

Table 2 reports on next item recommendation by considering product metadata. We tried the simplest approach of adding metadata features by replacing item embedding with a concatenation of item embedding and metadata embedding directly. Note that for products without the provided dense embedding vectors, we simply initialize their metadata embedding by using the average embedding from the train data. In the SASRec model, adding both text and image vectors does not bring noticeable gains on our own-eval set. However, on the official LB set, 0.02 MRR gains are achieved. We applied the same metadata inclusion approach on the SRGNN model but could not observe any further MRR gain. In [2], adding category information on top of pure item interactions did not show any further recommendation performance gain. It looks that how to integrate items' metadata into nowadays sophisticated neural models is still an open question and worth more in-depth investigations.

[3]https://github.com/Mehrab-Tanjim/DynamicRec

[4]https://recbole.io/

Model	Features	MRR on own-eval	MRR on LB2
SASRec		0.42	0.144
SASRec	text	0.43	0.166
SASRec	text+img.	0.42	0.167

Table 2: Adding products' metadata features into the session-based recommendation models that predict immediate next items. MRR metrics on our own-eval set and official LB2 set are reported.

6 DISCUSSION

SBRec has attracted increasing interest from both academic and industry research teams and a set of new models have been proposed in recent years. In the SIGIR'21 e-Commerce workshop data challenge, Coveo provides a new dataset to support this active research area. We started from a quick survey of existing SBRec techniques based on reading survey papers like [14] and topic-focused code packages like [18]. Then, we focused on trying several competitive SBRec models ranging from s-KNN to more recent DL based models. With the availability of products' metadata, we also explored adding products' text descriptions or image representations into the SBRec model. The experiment results provided valuable insights for future R&D on the SBRec topic.

From the experiment results, we made the following observations. Although conventional models are suggested to be competitive to the DL based models in a set of recent SRBec studies [3], on the Coveo data, we observed that DL methods, i.e., SASRec, DynamicRec, and SRGNN, outperform the KNN model. This shows that the sessions contain useful clues to support a more accurate next item recommendation. Different from both SASRec and DynamicRec that process product sequences appearing in all sessions one by one, the SRGNN uses a GNN to explicitly connect all sessions. As a result, the item vectors learned by considering the graph connections show the best performance in the next item recommendation. Previous research, e.g., [17] already suggests that including products' metadata can provide additional helps on top of items' interaction representations. Following this suggestion, we tried the simplest solution of including products' metadata by concatenating metadata vectors with item vectors in both SASRec and SRGNN models. In the SASRec model, inclusion of metadata features brought performance gains. However, on the SRGNN model, we could not observe the similar gains. It looks that a more sophisticated way of including metadata features, e.g., using a separate network to model features directly as used in [13], need be explored.

Regarding the next step research on using the Coveo data to build a more accurate SBRec model, there are several directions. First, due to the limited time in the challenge, we had no chance to use several types of features, including query vectors in the search and category and prices in the product catalog. For obtaining higher performance, a comprehensive usage of available data resources is needed. Second, our solution of including products' metadata is still in an infancy stage. It is worth exploring more advanced integration methods. Third, to better deal with recommendations based on few or zero context interactions, we need improve our data generation plan, such as creating such training instances and including some zero-context sessions in our test set. Moreover, we believe that data augmentation (e.g., [8]) may play some critical roles to allow the SBRec model to behave better in the "cold-start" cases.

7 ACKNOWLEDGEMENTS

The authors thank Hao Gong from RIT Tokyo and Yandi Xia from RIT Boston for their helpful discussions during this work.

REFERENCES

[1] Federico Bianchi, Jacopo Tagliabue, Bingqing Yu, Luca Bigon, and Ciro Greco. 2020. Fantastic Embeddings and How to Align Them: Zero-Shot Inference in a Multi-Shop Scenario. In *Proceedings of the SIGIR 2020 eCom workshop, July 2020, Virtual Event, published at http://ceur-ws.org*. https://arxiv.org/abs/2007.14906

[2] Federico Bianchi, Bingqing Yu, and Jacopo Tagliabue. 2021. Bert goes shopping: Comparing distributional models for product representations. In *Proceedings of the ECNLP-4 workshop, Aug. 2021.*

[3] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 101–109.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]* (May 2019). http://arxiv.org/abs/1810.04805 arXiv: 1810.04805.

[5] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1809–1818.

[6] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[7] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[8] Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S Yu. 2021. Augmenting Sequential Recommendation with Pseudo-Prior Items via Reversely Pre-training Transformer. *arXiv e-prints* (2021), arXiv–2105.

[9] Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting Similarities among Languages for Machine Translation. *arXiv preprint arXiv:1309.4168* (2013). http://arxiv.org/abs/1309.4168

[10] Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic* 3, 2 (2011).

[11] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-Based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems* (Boston, MA, USA) *(DLRS 2016)*. Association for Computing Machinery, New York, NY, USA, 17–22. https://doi.org/10.1145/2988450.2988452

[12] Md Mehrab Tanjim, Hammad A Ayyubi, and Garrison W Cottrell. 2020. DynamicRec: A Dynamic Convolutional Network for Next Item Recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2237–2240.

[13] Md Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian McAuley. 2020. Attentive Sequential Models of Latent Intent for Next Item Recommendation. In *Proceedings of The Web Conference 2020*. 2528–2534.

[14] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Defu Lian. 2019. A survey on session-based recommender systems. *arXiv preprint arXiv:1902.04864* (2019).

[15] A Waswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, AN Gomez, L Kaiser, and I Polosukhin. 2017. Attention is all you need. In *NIPS*.

[16] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 346–353.

[17] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation.. In *IJCAI*. 4320–4326.

[18] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Kaiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2020. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. *arXiv preprint arXiv:2011.01731* (2020).