2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

User Preference Translation Model for Recommendation System with Item Influence Diffusion Embedding

Hao-Shang Ma
Institute of Computer and Communication Engineering
Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan
Email: ablove904@gmail.com

Jen-Wei Huang
Institute of Computer and Communication Engineering
Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan
Email: jwhuang@mail.ncku.edu.tw

***Abstract*—Recommendation systems which are designed to understand and predict user interest based on user preferences play an important role in the era of information explosion. We propose the item influence embedding which adopts the social influence diffusion concept to model the item relations. We can learn the activation paths in items-item relation graph. In addition, for generating top-k items, most of recommendation systems calculate the similarity between user embedding and embedding of all items. The calculation costs too much time when number of users and items are huge. Therefore, we propose the User Preference Translation Model (UPTM) to recommend the Top-k items based on the language translation technology. UPTM directly generates the recommendation items based on translating the user preference. We can avoid to calculate the similarity of user embedding and item embedding. From the experimental results, UPTM not only outperforms the compared methods but also save the time in real large datasets.**

***Index Terms*—Recommendation Systems, Translation-based Recommendation Model, Item Influence Embedding**

I. INTRODUCTION

Recommendation systems which are designed to understand and predict user interest based on user preferences play an important role in the era of information explosion. Generally, recommendation systems are trying to learn the low dimensional representation of users and items. Many features can be adopted in recommendation systems, for example, user-item interactions, user features, item features, and other information such as the temporal factor.

Some approaches focus on learning the item embedding to realize the relations between items [11] [13] [2]. Nevertheless, the relationship between different items are limited in the common text or co-occurrence. In real world, when we explore the e-shopping website, we search for some specific items which we want to buy. The system usually recommends other items to users such as the similar items and the items which are watched by similar users. Sometimes, we will buy many related items if we are attracted by a target item. We would like to discover the target items which can trigger users to buy as much related items as possible. Therefore, We adopt the social influence diffusion [7] concept to help us to learn the relationship between items. On social network, people spread influence to their neighbors and receive influence from their neighbors at the same time. A user is activated by another user since they have same opinion tendency. Some social recommendation approaches learn the influence propagation process between people in the network, leading to the similar interests among the connecting user on the diffusion path. Same as the social network, assume an item-item network is formed from users' sequential interaction behavior with item. The item influence propagation represents when a user interacts with an target item, the probability that user interacts with the connecting items after interacting with the target item. According to this concept, we learn the item relations through modeling the item influence diffusion on the item-item network.

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

In this work, we propose a User Preference Translation model with item influence embedding (UPTM) to recommend items. Inspired by some session-based item recommendation approaches uses recurrent neural networks (RNN). These works try to learn the users and items embedding based on RNN and calculate the similarity between users and items to predict the next items [8] [15]. However, when the number of users and items are large, the similarity which is usually calculated by the dot product costs much time. We would like to adopt the concept of translation model in natural language processing. The translation model is designed for modeling the sequential text or sentences. The users' preference and behaviors can be seen as the sequential data. After training a User Preference Translation model, we input the user's preference, then the model gives us the recommendation items directly. In UPTM, first, the item-item relationship is modeled from sampling the influence diffusion path on item-item network. From the generated influence diffusion path, the relations include which items can triggler users to interact with more items and the sequential interaction information can be realized. Then, we use the skip-gram model to learn the item influence embedding. Finally, we propose the translation model to translate the users' preference into users' future behaviors with the item influence embedding. For evaluating the performance of proposed model, we compare UPTM model with other existing model on real four datasets.

We summarize the contributions of this work as follows:

1) We propose to apply the influence diffusion on item-item network to simulate the item relations and embed the item influence relations into item representations.
2) We propose the translation model to translate the users' preference into users' future behavior for recommendation to avoid the similarity calculating between users and items.
3) From the experimental results, our proposed method outperforms the existing methodologies for top-k recommendation.

II. RELATED WORKS

In this section, we introduce research related to recommendation systems. Recommendation systems aims to learn the user and item latent representation and calculate the user-item similarity to predict the items which users will have interest.

A. Item embedding-based Recommendation

In item embedding-based recommendation, the proposed methods aims to consider the information of items, for example, the co-occurrence of different items, context of items, context relation of items. Wang et al. [11] propose the Item Concept Embedding, ICE, which build the item-text network to learn the item relations via the common text information of items. In LearnSuc [12], Wang et al. propose the multi-type item embedding to learn the context items' representations collectively from the itemset structure.

B. Collaborative Filtering with Deep Neural Network

Collaborative filtering solves the recommendation problem by assuming that users with similar behaviors exhibit similar preferences for items. He et al. [5] propose NeuMF model that utilize the non-linearity of multi-layer perceptron to replace dot products of matrix factorization. They combine the multi-layer perceptron and matrix factorization to learn the user and item embedding. Wang et al. [14] propose NGCF which is based on the graph neural network. They encodes the collaborative signal which represents the high-order connectivities by performing embedding propagation. Another kind of Generative Adversarial Networks-based methods try to apply GAN to recommendation. They try to obtain more satisfactory recommendation accuracy by adopting adversarial training methods instead of optimizing pair-wise learning functions. Chae et al. [1] suggest a new direction of vector-wise adversarial training and propose the GAN-based CF framework.

Generally, recommendation systems utilize any possible information of users, items, and historical behaviors to learn the user embedding and item embedding. For top-k recommendation, recommendation systems calculate the similarity of users and items and rank the score of items to recommend. However, the similarity calculation limits the capability of recommendation system since the dot-product calculation costs much time when number of users and items are large.

III. METHODOLOGY

In this section, we will formulate the translation problem for recommendation. Then, we will introduce the procedure of learning item influence embedding and the **User Preference Translation model (UPTM)** in detail.

A. Problem Formulation

The recommendation task takes the users' past behavior as the preference, then predict the items which user will have interest in the future. Suppose the $U = \{u_1, u_2, ...u_m\}$ and $I = \{i_1, i_2, ..., i_n\}$ are denoted the user and item set. Each user u has the preference $P_u = (i_{u,1}, i_{u,2}, ...i_{u,t})$ where t is the interactive order. Given a user's preference record P_u, we would like to recommend the Top-k items $R_u = \{i_{u,t+1}, i_{u,t+2}, ..., i_{u,t+k}\}$ which user u will interact.

B. The Model Architecture

We develop a new translation-based recommendation model, User Preference Translation Model with item influence embedding, abbreviated as UPTM. As shown in Figure 1, the model contains the simulation of the item influence embedding and the translation of users' preference. First, we generate the item influence diffusion paths using the social influence paths sampling from the item-item relation graph. UPTM learns the item influence embedding according to the influence paths. Then, UPTM encodes the users' preference based on the item influence diffusion embedding and learns the parameters in the hidden layer to output the item embedding. Finally, UPTM generates the recommendation list from the decoder of translation module, and applies the softmax function and top-k sampling.

C. Item Influence Diffusion Embedding

We propose the item influence diffusion embedding to embed the information of item influence diffusion as shown in Figure 2. First, we need to construct the item-item relation graph from transactions. In the transactions, users interact with different items sequentially. A user may interact with more items after an item since the the item trigger the user. Therefore, according to the order of items in a transaction, there is a relation between two consequent items. We define the item-item relation graph can be consisted of the items and the directed relations. The weight on the relation edges is defined to the number of relation occurs in the transactions. Finally, the influence probability of an edge (u, v) can be formulated as $p(u, v) = \frac{w(u,v)}{\sum_{v_o \in Out(u)} w(u,v_o)}$, where $Out(u)$ is the set of out-neighbors of user u

1) Maximum Influence Path (MIP) and Maximum Influence In-Arborescence(MIIA): For simulating the item influence propagation paths, we adopt the concept proposed in [3]. Assume the influence from one node to another in G can only be transmitted along the maximum influence path between the

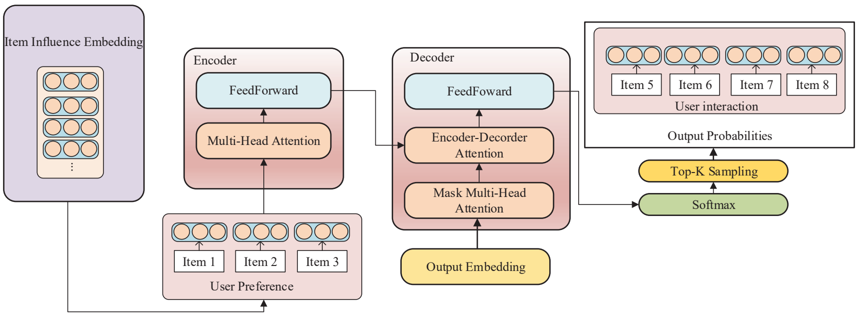


Fig. 1: The Framwork of User Preference Translation model with Item Influence diffusion Embedding

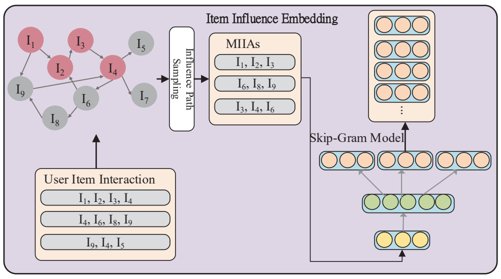


Fig. 2: Item Influence Diffusion Embedding Procedure

two nodes. Therefore, $MIP_G(u, v)$ has the maximum propagation probability. $MIP_G(u, v) = \arg\max_P\{pp(P) \,|\, \forall P \in \varphi(G, u, v)\}$, where $\varphi(G, u, v)$ is the set of all the possible paths on G extending from u to v; and $pp(\cdot)$ is the *propagation probability* of given path $P =< u = p_1, p_2, \ldots, p_m = v >$, which is defined as $pp(P) = \prod_{i=1}^{m} pp(p_i, p_{i+1})$.

We generate the maximum influence in-arborescence(MIIA) from MIP, which consider the received influence from the in-edges. The MIIA of node $v \in V$, *MIIA*(v, θ) can be formulated as follow:

$$MIIA(v, \theta) = \cup_{u \in V, pp(MIP(u,v)) > \theta} MIP_G(u, v), \qquad (1)$$

where $MIP_G(u, v)$ is the maximum influence path from u to v and the influence pruning threshold θ. The pruning threshold is set to ignore the edges with too small influence probability. Generally, *MIIA*(v, θ) scoping the influence region of node v in G. *MIIA*(v, θ) takes the view from the receiver's perspective indicating the potential nodes that could propagate its influence to v.

2) Item embedding with MIIA: After generating MIIA from the graph, we have a set of $MIIAs$ which is denoted as M, where each $MIIA$ is defined as $m = (i_1, ... i_l)$. We adopt the skip-gram model to learn the item representation vector v_{i_l}. In this model, we define a window size w in a sequence to determine the context items and central item. Then skip-gram model assumes the context items are related to central item and aims to learn the item representation that related items locate nearby in the embedding space. The objective function $\mathcal{L}$ can be formulated as follow:

$$\mathcal{L} = \sum_{m \in M} \sum_{i_l \in m} (\sum_{-w \geq j \leq w, j \neq 0} \log \mathcal{P}(i_{l+j} | i_l)), \qquad (2)$$

where the $\mathcal{P}(i_{l+j}|i_l)$ is the probability of predicting context items from the central target item. The probability can be defined as a softmax function $\mathcal{P}(i_{l+j}|i_l) = \frac{exp(v_{i_l}^{\top} v'_{i_{l+j}})}{\sum_{i=1}^{|I|} exp(v_{i_l}^{\top} v'_i)}$, where the v_{i_l} is the input vector of item i_l and $v'_{i_{l+j}}$ is the output vector of the j-th context items. Finally, we embed the information of item influence diffusion into the item embeddings. The embedding size is set to 64 in our implementation.

D. *User Preference Translation Model*

In sequence-to-sequence language translation models, they usually have a encode-decoder structure. We adopt the idea of Transformer architecture and follow the notations in that work [10]. The encoder and decoder are consisted of several identical layers. In encoder, each layer contain two sub-layers, multi-head attention layer and feed forward layer.

1) Multi-Head Attention Layer: Different with the scaled dot-product attention, the multi-head attention projects the queries, keys, and values into h subspace with different and learned linear projected function. The dimensions of queries, keys, and values are d_k, d_k and d_v. Then, the transformer performs the attention function in parallel and generate the output values corresponding to the dimensions. The formula is defined as follow:

$$MultiHead(Q, K, V) = Concat(head_1, head_2, ..., head_h)W^O$$
$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V), \qquad (3)$$

where the projection matrices are corresponding to $W_i^Q \in \mathbb{R}^{d_k \times d_k/h}$, $W_i^K \in \mathbb{R}^{d_k \times d_k/h}$, $W_i^V \in \mathbb{R}^{d_v \times d_v/h}$. The $Attention$ function is the scaled dot-product attention.

2) Feed Forward Layer: In each identical layer of encoder and decoder, there is a feed forward layer which is fully connected to each unit of the network. The fully connected layer enables the model learn the non-linearity and interaction between different units in the network. We apply the Gaussian

TABLE I: Statistics of the datasets

Dataset	**ML-1M**	**ML-20M**	**AmazonBook**	**Yahoo**
# Users	5953	138475	323186	19843
# Items	11847	22187	90398	144868
# Records	394486	9587046	10502385	642868
Avg. Records/User	66.26	69.23	32.49	32.39
Density	0.559%	0.312%	0.035%	0.022%

Error Linear Unit (GELU) [6] activation function which has better performance especially in transformer.

3) Softmax and Top-k sampling layer: In order to output the probabilities of items, we use the softmax function to transform the output of decoder to predict the most suitable item in the itemset. The softmax is defined as $Softmax(x_i) = \frac{exp(x_i)}{\sum_{i=0}^{|I|} exp(x_i)}$. However, using a sequence-to-sequence model for language generation task often generates the duplicate words or sentences. We use the top-k sampling [4] to solve the repetition problem when generating the predict items. The main idea is selecting the items with top-k probabilities. The k most likely next items are filtered and the probability distribution is recalculated among only those k items.

IV. Experiments

A. Experiment Setting

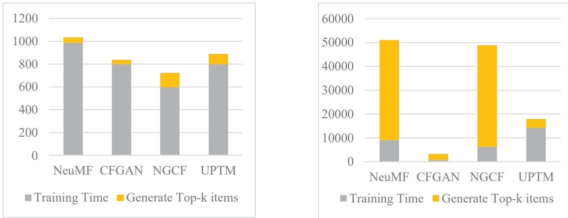
1) Datasets Description: Table I summarizes the detailed statistics of these datasets. For evaluating performance with UPTM, we split the datasets into training set and testing set by sequential splitting. We sort the users' interactions by the timestamp column, and take the first 80% user-item interactions of each user as the training set and remaining 20% as the testing set. We use the testing set to valid the performance of UPTM and tuning hyper-parameters.

2) Compared Methods: We compare UPTM with the following methods:

- **BPR-MF**: [9] BPR adopts the matrix factorization to learn the representation vectors of users and items.
- **NeuMF**: [5] NeuNF is generic and can express and generalize matrix factorization under its framework.
- **CFGAN**: [1] CFGAN is the state-of-the-ark GAN-based recommendation system. They propose the new direction, real-valued vector-wise adversarial training, to solve the problem of applying GAN on recommendation.
- **NGCF**: [14] NGCF is the state-of-the-ark CF-based recommendation system. This work exploits the user-item graph structure and design an embedding propagation layer to enhance the high-order connectivity information.

B. Overall Performance Comparison

Table II describes the performance comparison of all methods. We report the precision@K, recall@K, and ndcg@K in the results. Overall, UPTM has achieved the best performance in most cases. In some cases, we observe that NGCF and CFGAN are defeated by NeuMF, e.g. the precision in MovieLen 20M and the NDCG in MovieLen 20M. That may be caused by the sampling methods for splitting the dataset into



(a) MovieLen 1M dataset. (b) MovieLen 20M dataset.

Fig. 3: Costing time comparison of each Model.

the training part and the testing part. For comparing to our UPTM, we split the dataset using the sequential sampling instead of random sampling. We split the first 80% of each user's behaviors as the training and remaining 20% part as the testing.

From the results of MovieLen 1M and MovieLen 20M, in compared methods, we can observe that the larger number of items and users usually lead to the recommendation systems performs bad. In contrast, UPTM has more information to learn and makes the performance better. In addition, the Amazon book dataset has the largest number of users, and each user has a few interactions with items. Even we remove the unfrequent users and items, there still a lot of users and the dataset is very sparse. That makes the performance of all recommendation is lower than 10%. The Yahoo E-commerce dataset is a special dataset since we collect the data from the user's view behavior. Therefore, there are many duplicate items in users' preference so that all recommendation systems are hard to generate the recommendation list correctly.

C. Ablation Study of UPTM

In this section, we verify the effectiveness of the proposed item influence diffusion embedding. We use the MovieLen 1M and MovieLen 20M as the datasets to test the performance. We only report the precision@20, recall@20, and NDCG@20 in table III. From the experimental results, the item influence diffusion embedding actually helps the model to learn the more complex item relations in the users' preference.

D. Costing Time Comparison

We would like to verify the time efficiency of UPTM since UPTM does not need to calculate the similarity between users and items. When the number of users and items are large, the calculation costs too much time. The results are shown in Fig. 3. The unit of time is the second.

In MovieLen 1M dataset, all methods cost the most time on training model and cost few seconds to calculate the similarity and generate the top-k recommendation list for all users. In MoveLen 20M dataset, NeuMF and NGCF cost a lot of time to generate the top-k recommendation list since the number of users and items are large. CFGAN is the GAN-based method so that the model also does not need to calculate the similarity. The recommendation results are generated by the

TABLE II: Overall performance comparison

Methods	Movielens 1M			Movielens 20M			Amazon Book			Yahoo E-commerce		
	Pre@5	Pre@10	Pre@20	Pre@5	Pre@10	Pre@20	Pre@5	Pre@10	Pre@20	Pre@5	Pre@10	Pre@20
BPR-MF	0.0765	0.0697	0.0884	0.0687	0.0568	0.0489	0.0086	0.0111	0.0094	0.0098	0.0092	0.0086
CFGAN	0.0811	0.0868	0.0922	0.0711	0.0668	0.0622	0.0156	0.0126	0.0117	0.0138	0.0122	0.0117
NeuMF	0.1165	0.1051	0.0906	0.1149	0.1103	0.1035	0.0136	0.0141	0.0137	0.0116	0.0104	0.0095
NGCF	0.1194	0.1070	0.0918	0.0944	0.0871	0.0787	0.0183	0.0164	0.0144	0.0251	0.0217	0.0195
UPTM	**0.2262**	**0.2205**	**0.1853**	**0.2391**	**0.2384**	**0.2163**	**0.0321**	**0.0314**	**0.024**	**0.0264**	**0.0224**	**0.0208**
Methods	Movielens 1M			Movielens 20M			Amazon Book			Yahoo E-commerce		
	Rec@5	Rec@10	Rec@20	Rec@5	Rec@10	Rec@20	Rec@5	Rec@10	Rec@20	Rec@5	Rec@10	Rec@20
BPR-MF	0.0256	0.0358	0.0487	0.0254	0.0348	0.0683	0.0032	0.0071	0.0123	0.0039	0.0057	0.0108
CFGAN	0.0331	0.0427	0.0674	0.0334	0.0427	0.0774	0.0053	0.0076	0.0167	0.0051	0.0069	0.0134
NeuMF	0.0365	0.0686	0.1046	0.0402	0.0744	0.1044	0.004	0.0083	0.0159	0.0048	0.0072	0.0129
NGCF	0.0356	0.0672	0.1147	0.0456	0.0772	0.1086	0.0078	0.0146	0.0249	0.0057	0.0089	**0.0201**
UPTM	**0.0438**	**0.0812**	**0.1302**	**0.0543**	**0.1086**	**0.1862**	**0.0095**	**0.0168**	**0.0263**	**0.0082**	**0.0148**	0.0197
Methods	Movielens 1M			Movielens 20M			Amazon Book			Yahoo E-commerce		
	ND@5	ND@10	ND@20	ND@5	ND@10	ND@20	ND@5	ND@10	ND@20	ND@5	ND@10	ND@20
BPR-MF	0.086	0.0812	0.0901	0.0786	0.0864	0.0916	0.0118	0.0107	0.0129	0.0138	0.0141	0.0153
CFGAN	0.1066	0.0962	0.0916	0.0856	0.0986	0.09	0.0166	0.0152	0.0188	0.0152	0.0154	0.0178
NeuMF	0.0945	0.1033	0.1131	0.1193	0.1168	0.1231	0.0135	0.0141	0.0166	0.0144	0.0164	0.0177
NGCF	0.1168	0.1157	0.1235	0.0931	0.0921	0.098	0.0191	0.0176	0.0184	0.0156	0.0187	**0.0215**
UPTM	**0.1888**	**0.1996**	**0.1946**	**0.2001**	**0.2185**	**0.2327**	**0.0268**	**0.0283**	**0.0245**	**0.0189**	**0.0201**	0.0197

TABLE III: The performance comparison of UMTP and UMTP without Item Influence Diffusion Embedding

Methods	MovieLen 1M			MovieLen 20M		
	Prec@20	Rec@20	ND@20	Prec@20	Rec@20	ND@20
UPTM without Item Influence Diffusion Embedding	0.1735	0.1194	0.1828	0.2033	0.1792	0.2019
UPTM	0.1853	0.1302	0.1947	0.2163	0.1862	0.2327

generator in the GAN. However, in our experiment, UPTM has outperformed CFGAN in all test datasets.

V. Conclusion

In this work, we adopt the social influence spreading to model the trigger relation between items in the item-item network. In addition, we propose User Preference Translation Model to translate users' preference into users' future interactions. The model also can avoid the time-comsuming calculation of similarity between users and items. From the experimental results, UPTM outperforms the compared methods in most of cases.

References

[1] D.-K. Chae, J. Kang, J.-S. Kang, S.-W. Kim, J. Lee, and J.-T. Lee. Cfgan: A generic collaborative filtering framework based on generative adversarial networks. In *Proceedings of the 27th ACM on CIKM*, pages 137–146, 2018.

[2] W. Chen, X. Mao, X. Li, Y. Zhang, and X. Li. Pne: Label embedding enhanced network embedding. In *Proceedings of PAKDD*, pages 547–560, 2017.

[3] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD*, pages 1029–1038, 2010.

[4] A. Fan, M. Lewis, and Y. Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 889–898, 2018.

[5] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the International World Wide Web Conference*, pages 173–182, 2017.

[6] D. Hendrycks and K. Gimpel. Bridging nonlinearities and stochasticregularizers with gaussian error linear units. In *CoRR abs/1606.08415*, 2016.

[7] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD*, pages 137–146, 2003.

[8] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang. Stamp: Short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD*, pages 1831–1839, 2018.

[9] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI'09*, 2009.

[10] A. Vaswani, N. S. adn Niki Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of NIPS*, pages 5998–6008, 2017.

[11] C.-J. Wang, T.-H. Wang, H.-W. Yang, B.-S. Chang, and M.-F. Tsai. Ice: Item concept embedding via textual information. In *Proceedings of the 40th International ACM SIGIR*, pages 85–94, 2016.

[12] D. Wang, M. Jiang, Q. Zeng, Z. Eberhart, and N. V. Chawla. Multi-type itemset embedding for learning behavior success. In *Proceedings of the 24th SIGKDD*, pages 2397–2406, 2018.

[13] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu. Attention-based transactional context embedding for next-item recommendation. In *Proceedings of AAAI*, pages 2532–2539, 2018.

[14] X. Wang, X. He, M. Wang, F. Feng, and T.-S. 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.

[15] H. Ying, F. Zhuang, F. Zhang, Y. Liu, G. Xu, X. Xie, H. Xiong, and J. Wu. Sequential recommender system based on hierarchical attention networks. In *Proceedings of the 27th IJCAI*, pages 3926–3932, 2018.