2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) | 978-1-7281-1056-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/ASONAM49781.2020.9381352

Semantics Embedded Sequential Recommendation for E-Commerce Products (SEMSRec)

Mahreen Nasir
School of Computer Science
University of Windsor,
Windsor, ON N9B 3P4
Email: nasir11d@uwindsor.ca

C. I. Ezeife*
School of Computer Science
University of Windsor,
Windsor, ON N9B 3P4
Email: cezeife@uwindsor.ca

***Abstract*—In Collaborative Filtering methods, tailored recommendations cannot be obtained when the user-item matrix is sparse (i.e., has low user-item interactions such as item ratings or purchases). Conventional recommendation systems (ChoiRec12, HPCRec18, HSPRec19) utilizing mining techniques such as clustering, frequent and sequential pattern mining along with click and purchase similarity measures for item recommendation cannot perform well when the user-item interactions are less, as the number of items keep increasing rapidly. Additionally, they have not explored the integration of semantic information of products extracted from customers' purchase histories into the item matrix and the pattern mining process.**

To address this problem, this paper proposes (SEMSRec) which integrates semantic information of E-commerce products extracted from purchase histories into all phases of recommendation process (pre-processing, pattern mining and recommendation). This is achieved by i) learning semantic similarities between items from customers' purchase histories using prod2vec model, ii) leveraging this information to mine semantically rich sequential purchase patterns and, iii) enriching the item matrix with semantic and sequential product purchase information before applying item based collaborative filtering. Thus, SEMSRec can provide Top-K personalized recommendations based on semantic similarities between items without the need for users' ratings on items. Experimental results on publically available E-commerce data set show that SEMSRec provides more relevant recommendations over other existing methods.

***Index Terms*—Collaborative filtering, data mining, electronic commerce, recommender systems, semantics, sequential model.**

I. Introduction

A. Background

E-Commerce recommendation systems facilitate customers' purchase decisions by recommending products of interest [1]. Collaborative Filtering (CF) is a common matrix-completion based recommendation technique, which inputs a user-item rating matrix representing user-item interactions [1] and outputs top item recommendations for each target user, by finding similarities among users or items [1] [3]. Finding these interactions is important to infer users' preferences and items' characteristics. CF suffers from (1) sparsity (users' interact with limited products) and (ii) cold start (an item cannot be recommended if no interaction exists). Furthermore, to learn customers' purchase behaviors for next purchase prediction, frequent sequential purchase patterns can be extracted by applying sequential pattern mining techniques (SPM) [7] on purchase sequences. However, SPM lack (1) contextual information: i.e., for rules, the chocolate products "Ferrero Rocher", and "Ferrero Rondnoir" are totally different as they cannot learn implicit and inherent similarities between products according to their contextual co-occurrence and features such as brand, title and description which have an influence on succeeding purchases and diverse product recommendation, (2) personalization: rules are not targeted for a specific customer, as they infer decisions based on a global view of sequences. Recently, Neural Language Models (NL) word2vec [10] have been extended to recommender systems. Prod2vec [6] uses the terminology from word2vec [10] and learns vector representations of products to find similarities between products by considering a purchase sequence as a sentence and the products in the sequence as words. In principle, the products occurring in the same context (neighborhood) are similar in semantics (meaning).

*This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an operating grant (OGP-0194134) and a University of Windsor grant.

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

B. Contributions

The main contributions are to: (i) extract semantic knowledge of items from customers' purchase histories to compute item to item similarities, (ii) use items' semantic information to mine sequential purchase patterns from purchase sequences of customers with similar buying behaviors and (iii) enrich the item-item matrix by integrating items' semantic and sequential information for collaborative filtering without the need for user ratings, thereby improving the quality of recommendations and providing personalized recommendations.

C. Problem Description

This paper focuses on the task of Top-K personalized recommendations formally stated as, consider a set of users $U = \{u_1, u_2, \ldots, u_n\}$ and a set of products $P = \{p_1, p_2, \ldots, p_n\}$. Each user u is associated with a sequence of some items from P such that $S_u = (S_1, S_2, \ldots..S_n)$. Given all users' purchase sequences S_u, the system aims to recommend each user u a list of Top–K tailored products from the candidate set based

on the semantic and sequential relationship between products instead of u's interactions (ratings) on products.

The paper is structured as follows. We discuss related work in Section 2, the proposed SEMSRec method in Section 3, and experimental studies in Section 4. The paper is concluded in Section 5 along with future work.

II. Related work

A. General Recommendation

Collaborative Filtering [1], matrix factorization [12], and rule based approaches [2] [14] are traditional recommendation methods. Recommendation tasks involves either implicit feedback (clicks, purchases) or explicit feedback (ratings, reviews). For implicit feedback, BPR [11] utilizes a pairwise ranking loss in a Baysian framework and optimizes the latent factor model. HPCRec18 [14] system used normalized purchased frequency matrix to improve the quality of ratings and then mine the consequential bond between clicks and purchases in each session to predict the ratings for next possible purchase. However, methods under this category do not consider the order of actions, hence they cannot capture sequential patterns.

B. Sequential Recommendation

Sequential recommendation predicts the future item(s) that a user will be interested in given her past item interactions. Some previous works on sequential recommendation using sequential pattern mining [5] [14] and explicit sequential association rules based on statistical co-occurrences [2] have limitations of a huge search space and suffer from suitable threshold settings which may lead to large number of rules most of which are redundant. [3] [14]. HSPRec19 [3] augmented the item rating matrix with sequential purchase patterns of customers to quantitatively and qualitatively enrich the matrix for collaborative filtering. However, none of the systems utilized semantic knowledge about items while computing the consequential bond between click and purchase sequences and during the sequential purchase pattern mining process. Other line of work proposed factorized personalized Markov chains (FPMC) [12] by integrating matrix factorization and Markov chains, for next basket recommendation. More recently, deep learning have shown great progress, and many new techniques such as RNN [8] and CNN [13] have been adapted to sequential recommendation. In RNN based methods, user's preferences are represented by embedding historical interactions into a latent vector. RUM [4] proposes to use memory network to improve sequential recommendation by explicitly capturing item and feature level sequential patterns. Convolutional Sequence Embeddings (Caser) [13], a CNN-based method captures high order Markov chains by applying convolutional operations on the embedding matrix of the L most recent items. A self-attention based sequential model (SASRec) [9] captures long term user preferences by using attention mechanism and makes its predictions based on relatively few actions.

III. Proposed Semantic Embedded Sequential Recommendation(SEMSRec)

The proposed SEMSRec system integrates items' semantic and sequential information extracted from customers' purchase histories to compute item similarities for personalized recommendations without using item ratings or purchase frequency. It also integrates semantic knowledge to mine semantically rich frequent sequential purchase patterns. Thus, SEMSRec takes purchase sequences (PS), number of Top-N users with similar purchase behavior (N), number of Top-K recommendation items (K), minimum support(s), minimum semantic similarity (m_sem) as input and recommends set of items to target user that are similar in semantics and purchased frequently in sequential order according to his purchase history, (see Algorithm 1).

A. Learning Products' Semantic Representation

The similarity or correlation between products is a good indicator of the potential similarities between users' purchase behaviors. Inspired by the Skip-gram model [10] used in prod2vec [6], we use co-occurrence relationships between products in each user's purchase sequence to learn product embeddings. Accordingly, the model learns vector representations that are good at predicting the nearby products given the current product and aims to maximize the objective function over the entire set S of purchase sequences, as in Eq.(1) from [6],

$$\mathcal{L} = \sum_{s \in S} \sum_{pi \in s} \sum_{-c \leq j \leq c, j \neq 0} log\mathbb{P}(p_{i+j}|p_i) \tag{1}$$

Probability of $P(p_{i+j})|p_i)$ is the probability that product p_{i+j} co-occurs with the current product p_i and is defined using the soft-max [6] function as in Eq. (2),

$$\mathbb{P}(p_{i+j}|p_i) = \frac{exp(v_{pi}^T v_{pi+j}^{/})}{\sum_{p=1}^{p} exp(v_{pi}^T v_{pi}^{/})} \tag{2}$$

where v_p and $v^/p$ are the input and output vector representations of product p, c is the length of the context (i.e., with similar neighboring purchases) for product sequences, and P is the number of unique products in the vocabulary.

An Example of Learning Product Representations Using Prod2vec. For a customer, given the input purchase sequence ['20674', '21242', '20675', '21245', '20677', '20655'] represented as a sentence with words (products) as p_1= 20674, p_2 = 21242, p_3 = 20675, p_4= 21245, p_5 = 20677, p_6 = 20655. Consider the product '20675' as center product (word), the goal is to train the model to predict the neighboring (context) products which are ['20674' , '21242' , '21245', '20677', '20655'] by learning vector representations. The model works as explained below.

Input:sentences (purchase sequences), Input layer size – [1 x V], Input hidden weight matrix X – [V x N], dimension of embedding vector (hidden layer) – N, Hidden-Output weight matrix Y – [N x V], Output layer size – C [1 x V].

Output: Vector representation of products in V across N.

Steps:

Algorithm 1: Semantic Embedded Sequential Recommendation(SEMSRec)

Input : purchase sequences (PS), number of Top-N users with similar purchase behavior (N), number of Top-K recommendation items (K), minimum support(s),minimum semantic sim (m_sem)

Output : set of Top-K recommended items (RS)

Intermediates: matrix (PV) of features x products, matrix (M_1) of item x item similarity, purchase sequence vector of user u ($\overrightarrow{PS_u}$), similar purchase sequence data base (SPSDB), semantic similar sequential patterns(SSP)

```
Matrix (PV) of features x items ← computed using section 3A ;
Similarity Matrix (M) of item x item ← computed using cosine similarity between item features from PV using equation (4) in section 3B ;
for each purchase sequence (PS) do
    PS_u ← calculate by taking average of vectors of all products in the purchase sequence using matrix PV in section 3C;
for purchase sequence vector of each target user PS_ut do
    for purchase sequence vector of each user PS_u do
        compute similarity as similar users(PS_ut, PS_u) using section 3D ;
    sort results from the previous step ;
    N← store Top-N users with semantically similar purchases ;
SPSDB ← Extract purchase sequences of Top-N similar users from PS ;
SSP ← Generate semantic sequential patterns from SPSDB using section 3E;
M_1 ← Modified with semantic similarity and sequential score of products using equation (5) in section 3F ;
for each item i in user profile u_p of target user u_t do
    for each item j in M_1 do
        RS ← retrieve score (i,j) from M_1;
RS ← sort the results from RS and retrieve set of Top-K recommendations ;
```

1) Create a vocabulary of size (V). Here, V =$\{1 : 20674, 2 : 21242, 3 : 20675, 4 : 21245, 5 : 20677, 6 : 20655\}$.
2) Create weight matrices, X $\in R^{N\times V}$ and Y $\in R^{V\times N}$.
3) Represent each product as one hot vector (p), e.g., product '21242' will be encoded as [010000].
4) Get embedded product vector by taking product of the one hot vector with the input weight matrix X.
5) Multiply the result vector with the output weight matrix Y to get embedded vectors for context products. For example, embedded vector for products p_i =p_3='20675'and p_j=p_1='20674' will be the third and the fifth columns respectively in the PV matrix in Fig.1. To compute the probability that product p_j occurs with product p_i, use the softmax function Eq.(2). The probability scores of context products ['20674', '21242', '21245', '20677', '20655'] are 0.89, 0.65, 0.67, 0.85, 0.31 respectively showing that products '20674'[Green Polka Dot Bowl] and '20677'[Pink Polka Dot Bowl] are more closer to the product '20675'[Blue Polka Dot Bowl] in semantics.

$$PV = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ 0.24 & -0.13 & 0.21 & 0.17 & 0.16 & 0.14 \\ 0.22 & 0.31 & 0.26 & 0.27 & -0.21 & -0.15 \\ -0.32 & 0.13 & -0.36 & 0.15 & 0.24 & -0.24 \\ 0.12 & -0.23 & 0.46 & 0.37 & 0.31 & 0.14 \end{bmatrix}$$

Fig. 1: Matrix PV of size (NxV) for Vector Representation of Products

$$M = \begin{bmatrix} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ p_1 & 1.0 & 0.818 & 0.873 & 0.844 & 0.847 & 0.021 \\ p_2 & 0.818 & 1.0 & 0.806 & 0.863 & 0.778 & 0.018 \\ p_3 & 0.873 & 0.806 & 1.0 & 0.809 & 0.877 & 0.760 \\ p_4 & 0.844 & 0.863 & 0.809 & 1.0 & 0.761 & 0.650 \\ p_5 & 0.847 & 0.778 & 0.877 & 0.767 & 1.0 & 0.012 \\ p_6 & 0.021 & 0.018 & 0.760 & 0.650 & 0.012 & 1.0 \end{bmatrix}$$

Fig. 2: Item to Item Similarity Matrix

B. Integrating Semantic Information into Item-Item Matrix

Next, compute products' semantic similarity using Eq. (3) on product vectors in the PV matrix (Fig.1).

$$CosineSimilarity(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \quad (3)$$

where x_i and y_i are components of vectors of products x and y respectively. For example, cosine similarity (p_2,p_4) is 0.86 and (p_2,p_5) is 0.77, showing that product p_2 is closer to product p_4 in the vector space than product p_5. Next, entries in the item to item similarity matrix M (Fig.2) are populated using Eq.(4) where each entry $R_{x,y}$ represents semantic similarity between products x and y in the vector space.

$$M_{x,y} = \begin{cases} 1, & if \quad x = y \\ CosineSimilarity(x, y), & otherwise \end{cases} \quad (4)$$

C. Aggregated Vector Representation of Purchase Sequences

Next, we utilize products' semantic similarity information to compute vector representation of each purchase sequence (PS). This is obtained by computing row wise mean of vectors of products that are in the purchase sequence using PV matrix and then taking the transpose of the result. For our example, the purchase sequence has vector representation as [0.166 0.076 0.053 0.040].

D. Top-N neighbors

Next, Top-N neighbors with semantically similar sequences are identified by finding cosine similarity using Eq.(3) between purchase sequence vector in the test data (target customer) and each purchase sequence vector in the training data. For example, consider two purchase sequence in test and train data as ([‘20677’, ‘20674’, ‘21244’, ‘21239’, ‘21242’]) and ([‘85116’, ‘22375’, ‘71477’, ‘22492’, ‘22771’, ‘22772’]) with vector representations as $\overrightarrow{PS}_{ut}$ = [-0.03 0.21 0.13 0.03] and $\overrightarrow{PS}_{u}$ = [0.13 0.22 -0.02 0.002]) respectively. So, the cosine similarity between $\overrightarrow{PS}_{ut}$ and $\overrightarrow{PS}_{u}$ is 0.61. Similarly, cosine similarity between $\overrightarrow{PS}_{ut}$ in test data is computed with other $\overrightarrow{PS}_{u}$ in the training data. The results of cosine similarity are then sorted in decreasing order to select Top-N customers with similar purchase behaviors.

E. Sequential Pattern Extraction

Purchase sequences of Top-N neighbors are extracted from the training data set and are then converted into Sequential Historical Database (SHOD) [3] format for extracting frequent sequential patterns. For example, a sequence will be represented as $< 20674\ -1\ 20674\ 21245\ -1\ 21239\ -1\ 21242\ -2 >$ where -1 indicates the end of item and -2 indicates the end of sequence.

Example to extract frequent sequences using Prefixspan [7]. Next, frequent sequential patterns from these sequences containing semantically similar products are extracted using PrefixSpan [7]. The goal is to find products which are purchased in sequential order with a minimum support of 1.0% and have a semantic similarity score of $>= 0.5$. It is here in the mining process that we integrate semantic information of products from matrix M to prune sequences of products having a similarity score less than the specified similarity threshold in addition to the traditional method of removing sequences based on support count. This gives us frequent sequential purchase patterns of products which are similar in semantics. Some of the semantic rich frequent sequences along with their support count are : $< 20674 > SUP : 47, < 20674, 20675 > SUP : 23, < 20674, 20676 > SUP : 26, < 20674, 20677 > SUP : 20, < 20674, 21080 > SUP : 19, < 20674, 20676, 20677 > SUP : 25$. The sequence $< 20674, 20676, 20677 >$ suggests that these products are semantically similar and are purchased frequently in sequential order. So, if a customer has purchased products ‘20674’ and ‘20676’, the next product for recommendation which is semantically similar and purchased more frequently in sequential order to these two products will be product ‘20677’.

F. Semantically and Sequentially Rich Item-to-Item Matrix

Each entry $R_{x,y}$ in the semantically rich item to item similarity matrix M is now updated with the sequential information about products by computing a score using Eq.(5),

$$\text{Score(x,y)= CosineSimilarity(x,y)*Confidence(x,y)} \quad (5)$$

$$M_1 = \begin{bmatrix} & p1 & p2 & p3 & p4 & p5 & p6 \\ p1 & 1.0 & 0.33 & 0.42 & 0.28 & 0.36 & 0.02 \\ p2 & 0.22 & 1.0 & 0.32 & 0.29 & 0.25 & 0.01 \\ p3 & 0.39 & 0.28 & 1.0 & 0.25 & 0.35 & 0.03 \\ p4 & 0.44 & 0.67 & 0.40 & 1.0 & 0.59 & 0.01 \\ p5 & 0.35 & 0.24 & 0.37 & 0.23 & 1.0 & 0.11 \\ p6 & 0.02 & 0.01 & 0.12 & 0.11 & 0.01 & 1.0 \end{bmatrix}$$

Fig. 3: Semantic and Sequentially Rich Item to Item Similarity Matrix

where CosineSimilarity(x,y) is already computed using Eq.(3) and Confidence (x,y) is computed using Eq.(6) ,

$$\text{Confidence(x,y)} = \frac{\text{Support(x,y)}}{\text{Support(x)}} \quad (6)$$

where support(x,y) is the measure of how both products x and y occur together frequently in sequential order in all sequences and confidence(x,y) is the measure of how frequently x and y occur together in sequential order given all sequences in which x occurs. For example, cosine similarity between p_1 = 20674 and p_5= 20677 is 0.84. Confidence(x,y) can be calculated as: $Confidence(p_1, p_5) = \frac{\text{Support}(p_1, p_5)}{\text{Support}(p_1)} = \frac{20}{47}$ =0.42. So, final score computation between product p_1 and p_5 using Eq. (5) will be; Score (p_1,p_5) = 0.84*0.42=0.35. Fig.3 shows semantic and sequentially rich updated item to item matrix M_1 after score calculations for sample products. This semantically and sequential rich item-item rating matrix can now be used by CF to recommend Top-K personalized items to users. For our example, the set of Top-K (K=5) recommended products $\hat{p}_i$ which are semantically similar and purchased sequentially is $\{\hat{p}_1, \hat{p}_2, \hat{p}_3, \hat{p}_4, \hat{p}_5\}$ where $\hat{p}_1$=20677,$\hat{p}_2$=20676,$\hat{p}_3$=21238,$\hat{p}_4$=21243,and $\hat{p}_5$=21239.

IV. Experiments

A. Dataset and Implementation Details

- Online Retail[1]: Contains eight-month purchase records of 3k users and 4k items from a UK based company.
- Amazon[2]: Data about ratings of items purchased at Amazon. Experiments were performed on a set of 10M transactions between 2013 and 2014 out of 142.8M available transactions.

Data set was partitioned using (i) leave one out (taking last sequence of each user for testing and all remaining sequences for training) and (ii) temporal user splitting with train and test splits of (a)70%, 30% and (b) 80%, 20%. SEMSRec performed well on a split of 80% and 20% indicating that including more historical user interactions in training better capture users’ interest. We implemented SEMSRec with python and used open source data mining library SPMF[3] for mining sequential patterns using PrefixSpan [7] and the github repository[4] for comparison with some sequential models. For SEMSRec, settings for the embedding dimension d is determined by

[1]https://archive.ics.uci.edu/ml/datasets/online+retail
[2]http://jmcauley.ucsd.edu/data/amazon/
[3]https://www.philippe-fournier-viger.com/spmf/index.php
[4]https://github.com/mquad/sars_tutorial

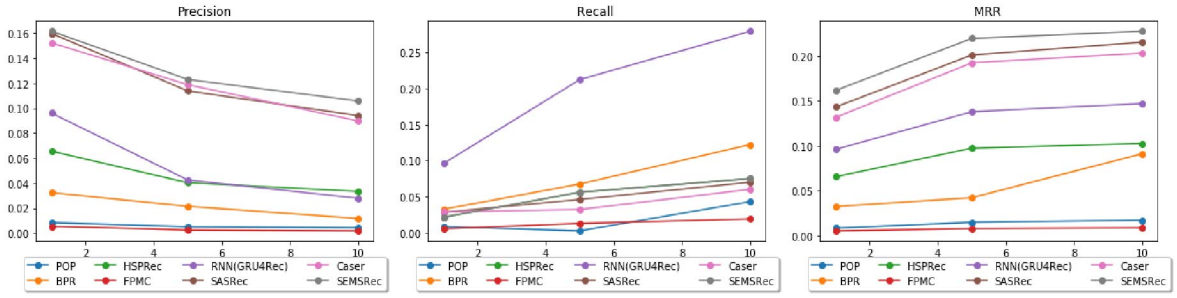

Fig. 4: Performance comparison of SEMSRec with other methods on Online Retail Data set

grid search in the range {10,20,30,40,50,100}, number of Top users with similar behavior N as {5,10,15}, number of Top Recommendation items K in {1,5,10,20,50,100} and minimum support s (%) for mining sequential patterns as {1,2,3}. Optimal performance was with d=100, N=5, s=1, m_sem=0.5. Followed by [13] [9] [4], the model was evaluated on a variety of metrics including Precision@K, Recall@K, Mean Reciprocal Rank (MRR), Hitrate@K and Normalized Discounted Cumulative Gain (NDCG@K).

B. Comparison Methods

We compared SEMSRec with baselines in (i) non-sequential recommenders **(Popularity based and BPR [11])** and (ii) sequential recommenders including sequential pattern mining based **(HSPRec [3])**, Matrix Factorizaton and Markov Chain based **(FPMC [12])** and deep learning models **(GRU4Rec [8], Caser [13], SASRec [9]).**

C. Prediction Performance

The results presented in Fig.4 exhibit that SEMSRec has better performance on all K tested by a good margin. Among the methods, the sequential recommenders such as HSPRec19 [3] and RNN [8] usually outperform non sequential recommenders such as POP and HPCRec18 [14] suggesting the importance of considering sequential information. On the other hand, FPMC [12] gave the least performance among all, showing that it is unable to completely capture the item semantics and sequences in online retail data set. Next, we examine the impact of the parameters on model's performance.

1) Influence of Top-N customers (N): By increasing the value of N, gradual decrease in model performance was noticed. This is because increase in the number of Top-N customers results in increased similar purchase sequences to recommend products from, however, those similar products are not necessarily purchased in sequence (which is important to capture users' long and short term behaviors) which lowers the recommender's performance.

2) Influence of the minimum support value (s) : By increasing the support values, gradual decrease was examined in the number of sequences and thus lower model performance was examined. The dominant term in the computational complexity of SEMSRec is $O(n^2)$ mainly due to computing similarity at item level and sequence level.

V. Conclusion

We present Semantic Embedded Sequential Recommender system (SEMSRec) to provide Top-K personalized recommendation by integrating E-commerce products' semantic and sequential relationships extracted from customers' purchase histories into CF's item similarity matrix. Experiments on publically available E-commerce data sets showed that SEMSRec, exhibited improved performance in terms of providing personalized recommendations. For future work, we intend to enhance the item matrix by extracting semantic and sequential information from other data sources such as customer's wish list, item reviews and their social networks.

References

[1] C. C. Aggarwal, "An introduction to recommender systems," in *Recommender systems*. Springer, 2016, pp. 1–28.

[2] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215, 1994, pp. 487–499.

[3] R. Bhatta, C. Ezeife, and M. N. Butt, "Mining sequential patterns of historical purchases for e-commerce recommendation," in *International Conference on Big Data Analytics and Knowledge Discovery*. Springer, 2019, pp. 57–72.

[4] X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin, and H. Zha, "Sequential recommendation with user memory networks," in *Proceedings of the eleventh ACM international conference on web search and data mining*, 2018, pp. 108–116.

[5] K. Choi, D. Yoo, G. Kim, and Y. Suh, "A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis," *electronic commerce research and applications*, vol. 11, no. 4, pp. 309–317, 2012.

[6] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 1809–1818.

[7] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *proceedings of the 17th international conference on data engineering*. Citeseer, 2001, pp. 215–224.

[8] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[9] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," *arXiv preprint arXiv:1205.2618*, 2012.

[12] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 811–820.

[13] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 565–573.

[14] Y. Xiao and C. Ezeife, "E-commerce product recommendation using historical purchases and clickstream data," in *International Conference on Big Data Analytics and Knowledge Discovery*. Springer, 2018, pp. 70–82.