Learning from History and Present: Next-item Recommendation via Discriminatively Exploiting User Behaviors

Zhi Li[1,2], Hongke Zhao[1], Qi Liu[1,*], Zhenya Huang[1], Tao Mei[3], Enhong Chen[1]

[1]Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China;
[2]School of Software Engineering, University of Science and Technology of China; [3]JD AI Research
{zhili03,zhhk,huangzhy}@mail.ustc.edu.cn; {qiliuql,cheneh}@ustc.edu.cn; tmei@jd.com

ABSTRACT

In the modern e-commerce, the behaviors of customers contain rich information, e.g., consumption habits, the dynamics of preferences. Recently, session-based recommendations are becoming popular to explore the temporal characteristics of customers' interactive behaviors. However, existing works mainly exploit the short-term behaviors without fully taking the customers' long-term stable preferences and evolutions into account. In this paper, we propose a novel Behavior-Intensive Neural Network (BINN) for next-item recommendation by incorporating both users' historical stable preferences and present consumption motivations. Specifically, BINN contains two main components, i.e., *Neural Item Embedding*, and *Discriminative Behaviors Learning*. Firstly, a novel item embedding method based on user interactions is developed for obtaining an unified representation for each item. Then, with the embedded items and the interactive behaviors over item sequences, BINN discriminatively learns the historical preferences and present motivations of the target users. Thus, BINN could better perform recommendations of the next items for the target users. Finally, for evaluating the performances of BINN, we conduct extensive experiments on two real-world datasets, i.e., Tianchi and JD. The experimental results clearly demonstrate the effectiveness of BINN compared with several state-of-the-art methods.

KEYWORDS

Next-item Recommendation, Sequential Behaviors, Item Embedding, Recurrent Neural Networks

1 INTRODUCTION

Recommender system, as an essential component of modern e-commerce websites, tries to predict what the most suitable products or services are of users, based on the users' preferences [30]. With the mechanism development of e-commerce, a massive amount of customer interactions (e.g., browse, click, collect, cart, purchase) have been logged, which imply luxuriant consumption patterns.
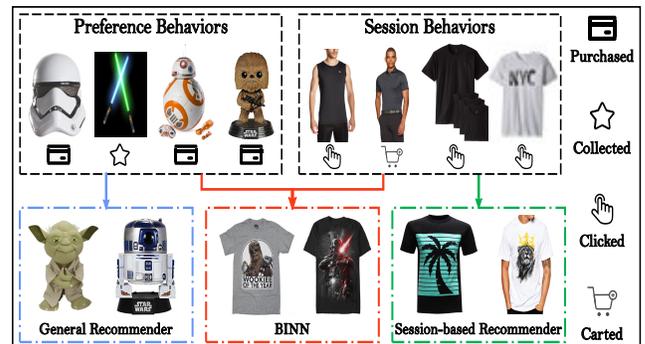
Figure 1: Recommending by integrating preference behaviors and session behaviors.

These information-rich logs provide opportunities for understanding customers' historical stable preferences and also their present consumption motivations, which may further contribute to smarter recommendations.

Along this line, there is a particular interest in understanding interactive behaviors of customers. Existing works can be concluded into two main paradigms. The first paradigm is the general recommenders. These works focus on mining the static relevancy between users and items from interactions, which are represented by the traditional collaborative filtering models [18, 22, 40]. For example, Zhang *et al.* made recommendations through a factorization model with different item semantic representations from the knowledge base [40]. However, most of these works have taken user-item specific relationships into consideration from the static views but neglect the dynamics and evolutions of users' preferences implied in sequential interactions. The other paradigm is recommending next items based on sequential pattern mining [34, 39] or transition modeling [29, 42]. Along this line, researchers recently show more interest in an e-commerce scenario where user profiles are invisible so that recommender systems are developed based on the user interactions in short sessions [15, 19, 27]. These session-based models have provided the comprehension about users' decision-making process in a short term, but the dynamics of preferences [38] and how to perfectly integrate both the historical stable preferences with present consumption motivations are still largely unexplored.

Actually, as a user's interactive behaviors naturally form a behavioral sequence over time, the user's historical preferences from the long-term view and present motivations or demands from the short-term view can be dynamically revealed. For instance, Figure 1 illustrates a typical online shopping scenario. The user's historical interactions imply that this user might be a "Star Wars" fan since

that the user has bought or collected various spin-off products of the "Star Wars". Moreover, we infer that this user would like to buy dark T-shirts because a black shirt is included in the personal cart and the user has browsed many short sleeve shirts. However, following the general collaborative filtering approaches, another spin-off product may be recommended since all the preference behaviors of the user's entire history are exploited in the static manners as shown in the blue chart of Figure 1. By contraries, if we only consider the current session behaviors of this user in accordance with what the session-based models do, another similar or popular shirt would be recommended as shown in the green chart of Figure 1. Actually, by exploiting both the historical stable preferences and present consumption motivations of this user, more attention should be paid to short sleeve shirts and the "Star Wars" graphic T-shirts perfectly match user's tastes. Therefore, we can conclude that a smarter recommender system should not only consider users' historical stable preferences but also take into account the present consumption motivations by discriminatively exploiting different terms or types of user behaviors.

Based on the intuition and observations, we propose a novel solution framework called Behavior-Intensive Neural Network (BINN) to address the next-item recommendation problem. Our BINN framework contains two main components: *Neural Item Embedding* and *Discriminative Behaviors Learning*. Specially, we propose a novel neural item embedding method to obtain a unified item representation space for learning latent vectors which could capture the sequential similarities between items. Different from the traditional item embedding methods which are based on inherent features such as item images or text descriptions, our neural item embedding method generates item representations by means of exploiting users' collaborative sequential interactions over items directly. Then, with the item embedded, we design two different behavior alignments, i.e., *Session Behaviors Learning* and *Preference Behaviors Learning* to respectively model users' present consumption motivations and historical stable preferences by discriminatively exploring interactive behaviors of users. Specific to the alignments, we respectively develop two deep neural network architectures to jointly learn the session behaviors and preference behaviors. Finally, by matching the potentially preferred items in the latent space, BINN generates recommendations for the target users. For evaluating the performances of BINN, we conduct extensive experiments on two real-world datasets. The experimental results clearly demonstrate the effectiveness of BINN compared with several state-of-the-art methods. In summary, the main contributions of this study can be summarized as follows.

- We propose to make item-recommendation by integrating both the historical preferences and present motivations of users, which are all learned from the users' interactive behaviors.
- We propose a novel Behavior-Intensive Neural Network (BINN) which includes embedding items by users' interactions and discriminative behavior alignments accompanied by two applicable neural network architectures.
- We conduct extensive experiments on two real-world datasets. The results show that BINN model outperforms other state-of-the-art methods from various aspects.

2 RELATED WORKS

Recommender systems play an essential role in many Internet-based services [30], such as e-commerces, and have arouse the great attention from both industry and academia. The relevant works of this study can be concluded into two main paradigms: the *General Recommenders* and the *Sequential Recommenders*.

2.1 General Recommenders

Most of the general recommenders focus on mining the static relevancy between users and items from their interactions, which are represented by the traditional collaborative filtering models [18, 22, 25]. More specifically, the most common approaches of collaborative filtering are the *neighborhood methods* [2, 17, 20, 31, 43] and the *factorization models* [4, 13, 14, 25, 41].

Neighborhood Methods are based on the similarities of entities (always users or items) and recommend the nearest neighbor items through the precomputed similarities [2, 17, 31, 43]. For example, Bell *et al.* proposed a new method to simultaneously derive interpolation weights for all nearest neighbors, leading to a substantial improvement of prediction accuracy [2]. Zhao *et al.* considered the expertise of investors to improve the neighborhood methods for the personalized investment recommendations in P2P lending [43]. Wang *et al.* presented a basic probabilistic framework which formalized the learning similarity as a regression problem [35]. Moreover, they introduced a novel multi-layer similarity descriptor which modeled the joint influences of different features to improve the neighborhood methods [35].

Factorization Models treat the recommendation as a user-item matrix reconstruction problem and model the user-item interactions by dot product of latent vectors [21, 25, 41]. Many previous works focused on better representing users or items to improve the qualities of recommender systems. For example, Liu *et al.* considered users' indecisiveness when the customers chose among competing product options and then proposed IMF method to mine the indecisiveness in customer behaviors to improve the performance of recommendation [22]. Zhao *et al.* introduced the Nash Equilibrium into the matrix factorization method to solve the group recommendation problem [41]. In recent years, deep learning has also been applied successfully to the classical collaborative filtering user-item matrix reconstruction problems from different perspectives. For example, many researches incorporated deep learning models for extracting and fusing side features, such as image features [12], text information [8, 33] and multimode data [40]. Moreover, some works developed the deep neural networks to learn the relationships between the users and items [11, 13].

However, most of these works provide recommendations by mining the static relevancy between users and items. The dynamics and evolutions of users' preferences, and also their present consumption motivations are usually not given special attentions.

2.2 Sequential Recommenders

In recent years, researchers start to focus on various sequential recommendation scenarios, such as next-basket [29, 36], session-based [15, 19, 27] and next-item recommendations [6, 39]. Among them, session-based ones are increasingly attractive.

Early works on sequential recommenders are almost based on the sequential pattern mining [34, 39] and the transition modeling [29, 42]. Due to the tremendous success of deep neural networks in the past few years, approaches to sequence data modeling have made significant strides and benefit a broad range of applications, such as NLP [9], social media [37] and recommendations [15, 19, 27]. For example, Hidasi *et al.* firstly applied recurrent neural networks (RNNs) by modeling the whole session for session-based recommendations, which outperformed item-based methods significantly [15]. Quadrana *et al.* focused on some session-based recommendation scenarios where user profiles were readily available and developed a hierarchical recurrent neural networks with cross-session information transfer [27]. Wang *et al.* used a hierarchical representation model to capture both sequential behaviors and users' general tastes by involving transaction and user representations in next-basket prediction [36]. Donkers *et al.* extended RNNs by representing the individual users for the purpose of generating personalized next item recommendations [6].

Although these models have taken the users' sequential information into consideration, there are still largely unexploited in the coherence of customers' sequential behaviors and the dynamics of historical preferences. Comparatively, in this paper, we propose to make item-recommendation by integrating both the historical preferences and the present motivations of users, which are all learned from the users' rich interactive behaviors over items.

3 BINN: BEHAVIOR-INTENSIVE NEURAL NETWORK

In this section, we introduce our proposed model for addressing the personalized next-item recommendations. We first formally define the next-item recommendation task and overview the proposed *Behavior-Intensive Neural Network* (BINN). Then we describe the two main components of BINN in detail, i.e., *Neural Item Embedding* and *Discriminative Behaviors Learning*.

3.1 Preliminaries

Next-item recommendation is the task of predicting what a user would like to access next based on her historical interactions. Here, we give a formulation for the next-item recommendation.

As user interactions naturally form a sequence over time, a log history H of the information system is a set of sequential interactions, i.e., $H = \{S_1, S_2, ..., S_n\}$, where $|H| = n$ denotes the number of users. Each user u has a corresponding interaction sequence $S_u \in H$ which can be represented as $S_u = \{(x_1, b_1), (x_2, b_2), ..., (x_T, b_T)\}$, where x_j denotes the j-th item that user u operates and b_j denotes the behavior type (e.g., click, cart, purchase). Then, the personalized next-item recommendation task can be defined as follow:

Definition 1 (Personalized Next-item Recommendation). Given a target user u with her sequential of interactive behaviors over items $S_u = \{(x_1, b_1), (x_2, b_2), ..., (x_T, b_T)\}$ and also all users' sequential interactions H, the personalized next-item recommendation task is to predict item x_{T+1} that the target user u is most likely to access in her next visit.

In this paper, we address this task with a novel personalized next-item recommendation framework, i.e., Behavior-Intensive Neural

Network (BINN). As shown in Figure 2, BINN contains two main components: Neural Item Embedding and Discriminative Behaviors Learning. Specifically, with a neural embedding model, we obtain a unified item representation space for learning the latent vectors that capture sequential similarities between items. Then we design two interactive behaviors assignments, named *Session Behaviors Learning* (SBL) and *Preference Behaviors learning* (PBL), to exploit the users' present consumption motivations and historical stable preferences over time. Finally, we jointly learn these two interactive alignments on the latent space of item representations and recommend top-k potentially preferred items to each target user.

3.2 Neural Item Embedding

In the first stage of BINN, Neural Item Embedding aims to generate a unified representation for each item by learning the item similarities from a large number of sequential behaviors over items. Previous works of sequential recommenders always use 1-of-N encoding or add an additional embedding layer in the deep learning architecture to represent items [15, 27]. However, for a superb collection of items in the big e-commerce platforms, on one hand, the 1-of-N encoding networks may cost unaffordable time and always cannot to be optimized well because of the high sparsity [3]. On the other hand, adding an additional embedding layer may make networks lose performances to some extend [15]. What's more, both two methods cannot reveal item sequential similarities which is implied in the user interactions. In this case, it is necessary to find an effective representation method to directly learn high-quality item vectors from the users' interaction sequences, with the result that items implied similar attractions tend to be close to each other.

In recent years, the progress in neural embedding has achieved tremendous advances in many domains, such as NLP [3, 24], social networks [5, 10, 26] and recommendations [1]. Among these works, item2vec [1] is one of the significant extensions of Skip-gram with Negative Sampling [24] to produce item embedding for item-based collaborative filtering recommendations.

In this paper, we propose an improved item2vec to capture item similarities and generate item representations by the means of exploiting users' collaborative interactions over items directly. Different from the words in sentences, some items in user interactions have often been frequently accessed. The reason for this phenomenon is that users are usually indecisive in their decision-making process [22], causing a lot of repetitive actions on the same item. In addition, these frequently-operated items also indicate users' main motivations, and other items interspersed these repeats may be very similar or competing to these items. On the other hand, items with low frequency may be aimlessly clicked [19], which will bring noise to the item embedding. Along this line, we take one step further to capture the characteristic of interaction sequences and propose a novel item embedding method, called *w-item2vec*, which considers the frequency of items as a weighted factor.

Inspired by item2vec [1], w-item2vec also uses a Skip-gram model with Negative Sampling method [24]. Specifically, given an item sequence $S_u^{'} = \{x_1, x_2, ..., x_N\}$ of user u from the interactive sequence S_u, the Skip-gram of w-item2vec aims at maximizing the following objective:
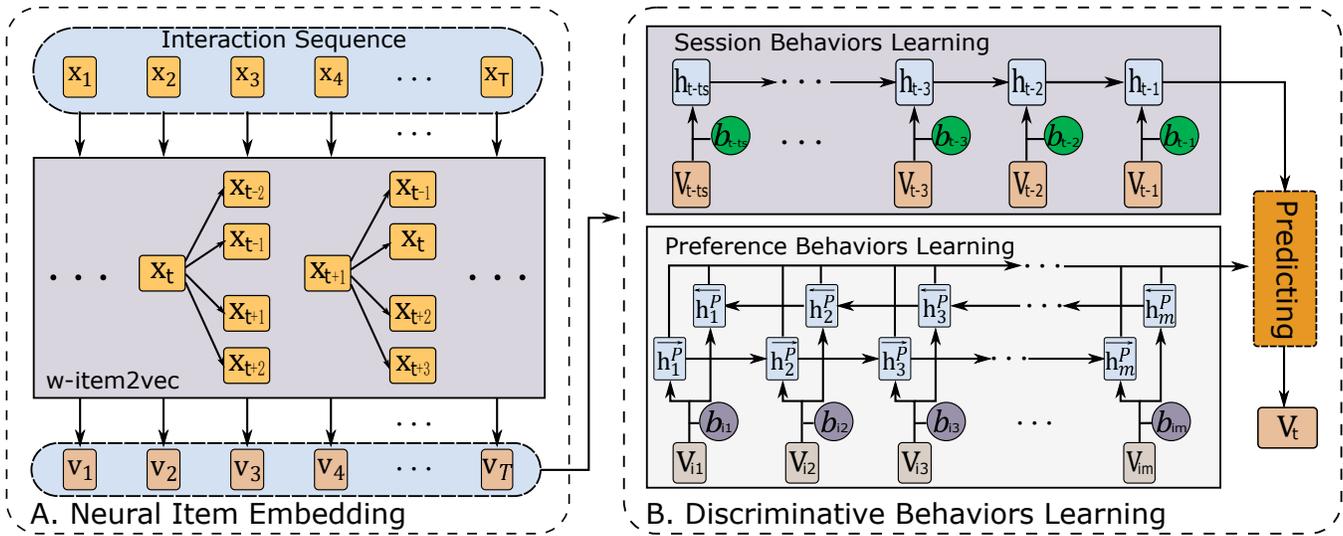
Figure 2: The overview of Behavior-Intensive Neural Network (BINN). A. Neural Item Embedding converts sequential items into a unified embedding space by w-item2vec. B. Discriminative Behaviors Learning constructs two alignments of user behaviors and discriminatively learns behavior information based on two LSTM-based architectures.

$$\arg\max \quad opt = \frac{1}{K} \sum_{i=1}^{K} \sum_{j \neq i}^{K} \log p(x_j|x_i), \quad (1)$$

where K is the length of sequence S_u, and $p(x_j|x_i)$ is defined as the softmax function:

$$p(x_j|x_i) = \frac{exp(w_i^T * v_j)}{\sum_k exp(w_i^T * v_k)}, \quad (2)$$

where w_i and v_i are the latent vectors that correspond to the target and context representations for item x_i. For alleviating computational complexity of the gradient $\nabla(x_j|x_i)$, Eq. (2) is always replaced by negative sampling:

$$p(x_j|x_i) = \sigma(w_i^T * v_j) \prod_{k=1}^{E} \sigma(w_i^T * v_k), \quad (3)$$

where $\sigma(x) = 1/(1 + exp(-x))$, E is the number of negative samples to be drawn per a positive sample.

Then, we improve the negative sampling model of Eq. (3) by considering the item frequency within interaction sequences as the weight of negative sampling process:

$$p(x_j|x_i) = (\sigma^{\Theta(x_i)}(w_i^T * v_j) \prod_k \sigma(w_i^T * v_k))^{\Theta(x_i)}, \quad (4)$$

where $\Theta(x_i)$ is the frequency of item x_i in the sequence. Consequently, the Skip-gram objective in Eq. (1) can be redefined as:

$$\arg\max \quad opt = \frac{1}{K} \sum_{i=1}^{K} \sum_{j \neq i}^{K} \log p(x_j|x_i)$$

$$= \frac{1}{K} \sum_{i=1}^{K} \sum_{j \neq i}^{K} \Theta(x_i)(\Theta(x_i) * \log \sigma(w_i^T * v_j)) \quad (5)$$

$$+ \sum_{k=1}^{E} \log \sigma(w_i^T * v_k)).$$

Finally, we train the w-item2vec by gradient descent, and obtain high-quality distributed vector representations for all the items.

With w-item2vec, we can capture item similarities with the help of user interactions and generate an unified item representation space, in which the representation vectors can reveal similarities and sequential relationships of items. And for each user u, we can generate an interaction sequence with embedding items as $P_u = \{v_1, v_2, ..., v_T\}$, where v_j denotes the d-dimensions latent vector of item x_j.

3.3 Discriminative Behaviors Learning

After obtaining item embeddings, Discriminative Behaviors Learning (DBL) could explore sequential behaviors as prior knowledge to recommend items that the target user is most likely to access in her next visit.

As illustrated in Figure 1, a user's decision-making process is mainly influenced by two factors: her present motivations and historical preferences. More specifically, users' present consumption motivations are dynamic in a short term and the recent fluctuations are also important to reflect the short-term characteristics. Considering that all the recent behaviors (e.g., click, collect, cart, purchase) may imply users' present motivations in a short term, we use all types of recent behaviors to represent the present consumption motivations. On the other hand, as for exploiting users'

historical preferences, not all types of behaviors could depict users' preferences. For example, we can imply that the user do not prefer an item if she just clicks on it without purchasing at last. Therefore, for modeling users' historical preferences, we only remain the behaviors which can clearly depict users' underlying preferences from interaction histories, i.e., purchase behaviors.

In fact, the interaction process of a user is the series of implicit feedbacks over time. Thus, different from traditional recommender systems which explore the user-item interactions from a static manner, we tackle the next-item recommendation by sequential modeling. Specifically, we design two discriminative behavior alignments: Session Behaviors Learning (SBL) and Preference Behaviors Learning (PBL) to discriminatively learn users' present consumption motivations and historical stable preferences. Further, on this basis, we develop two separate deep neural architectures based on LSTM [16] to jointly learn the motivations and preferences from these two alignments of behaviors.

3.3.1 Session Behaviors Learning. As illustrated in the green chart of Figure 1, the session behaviors in a short term can reveal the users' present consumption motivations. The Session Behaviors Learning (SBL) is to model the short-term session behaviors of the target user u. More formally, suppose we already have the previous interaction sequence $S_u = \{(x_1, b_1), (x_2, b_2), ..., (x_{t-1}, b_{t-1})\}$ with embedding items $P_u = \{v_1, v_2, ..., v_{t-1}\}$. The behavior b_i can be represented as an one-hot vector and the length of the vector is the number of interaction types, e.g., click. For determining whether a certain item x_i would be a possible element of the session behaviors, the SBL discrimination function D_{SBL} can be defined as follows:

$$D_{SBL}(x_i, x_t) = \Phi((t - i) \leq ts), \qquad (6)$$

where function $\Phi(a)$ is to compute a discrimination signal that equals to 1 if a is true and equals to 0 otherwise, x_{t-1} is the previous item of the prediction and ts is a controlling indicator to control the length of SBL. Specifically, as for a session-based scenario, ts is the length of the session. And for the non-session scenarios, ts is artificially specified. In this paper, we set ts to 10 as the default.

Aiming at the alignment of session behaviors, we develop a Contextual LSTM (CLSTM) [9] to learn users' present consumption motivations. After the initialization, at j-th interaction step, the hidden state h_j of each interaction is updated by the previous hidden state h_{j-1}, the current item embedding v_j, and the current behavior vector b_j as:

$$
\begin{aligned}
i_j &= \delta(W_{vi}v_j + W_{hi}h_{j-1} + W_{ci}c_{j-1} + W_{bi}b_j + \widehat{b_i}), \\
f_j &= \delta(W_{vf}v_j + W_{hf}h_{j-1} + W_{cf}c_{j-1} + W_{bf}b_j + \widehat{b_f}), \\
c_j &= f_jc_{j-1} + i_jtanh(W_{vc}v_j + W_{hc}h_{j-1} + W_{bc}b_j + \widehat{b_c}), \quad (7) \\
o_j &= \delta(W_{vo}v_j + W_{ho}h_{j-1} + W_{co}c_j + W_{bo}b_j + \widehat{b_o}), \\
h_j &= o_jtanh(c_j),
\end{aligned}
$$

where i_j, f_j and o_j are the input gate, forget gate and output gate at j-th step respectively, v_j is the embedding item vector, b_j is the behavior vector, c_j is the cell memory, $\widehat{b}$ is the bias term, and h_j is the output at j-th step.

We essentially use the final out state h_{t-1} as the representation of the present consumption motivations of user u, i.e., $\Psi_{SBL} = h_{t-1}$. With the above network structures, SBL can naturally model fluctuations of users' session behaviors to obtain representations of present consumption motivations.

3.3.2 Preference Behaviors Learning. As mentioned above, a smarter recommender system should not only consider users' present consumption motivations but also take into account the historical stable preferences. Therefore, in addition to exploiting motivations by SBL in a short term, PBL is used to learn users' stable historical preferences from the preference behaviors in a long term. Actually, only part of behaviors imply users' preferences. Thus, for determining whether a certain interaction $(v_i, b_i) \in S_u$ would be a possible element of the preference behaviors, the discrimination D_{PBL} can be defined as:

$$D_{PBL}(v_i, b_i) = \Phi(b_i \in P), \qquad (8)$$

where P is the preference behavior set which contains the types of preference behaviors i.e., collect, cart and purchase.

Different from SBL, PBL is a global representation of historical preferences with less fluctuations. That may make the architecture of SBL can not work well to obtain users' historical preferences. Inspired by Bidirectional RNN [32], we adapt the CLSTM to a bidirectional architecture, named Bi-CLSTM, to make full use of the contextual long-term representation in both forward and backward directions. Specifically, the cell of Bi-CLSTM is the same as Eq. (7) and it can principally be trained with the same algorithms as a regular unidirectional CLSTM. At each time step s of PBL, the forward layer with hidden state $\overrightarrow{h_s^P}$ is computed based on both the previous hidden state $\overrightarrow{h_{s-1}^P}$ and the current item-behavior pair (v_{is}, b_{is}); while the backward layer updates hidden state $\overleftarrow{h_s^P}$ with the future state $\overleftarrow{h_{s+1}^P}$ and the current item-behavior pair (v_{is}, b_{is}). Therefore, each PBL hidden representation h_s^P can be calculated with the concatenation of the forward state and backward state, i.e., $h_s^P = concatenate(\overrightarrow{h_s^P}, \overleftarrow{h_s^P})$.

After that, we can generate the unified representation of preference behaviors Ψ_{PBL} for user u through an average pooling layer:

$$\Psi_{PBL} = average(h_1^P, h_2^P, ..., h_m^P), \qquad (9)$$

Particularly, taking embedding preference interactions as inputs of above networks, PBL is able to learn and depict the profile of each user. That can help BINN to make a good understanding of users' historical stable preferences.

So far, from Discriminative Behaviors Learning (DBL), we have modeled two behavior alignments: session behaviors learning Ψ_{SBL} and preference behaviors learning Ψ_{PBL}. Then, after an fully connected layer, we can generate the d-dimensions representation $\widehat{v_t}$ of the next item.

3.4 Model Learning and Test Stage

Taking embeddings of sequential items as inputs of networks, DBL is able to learn both users' present motivations and historical preferences by controlling recurrent states update of the two network
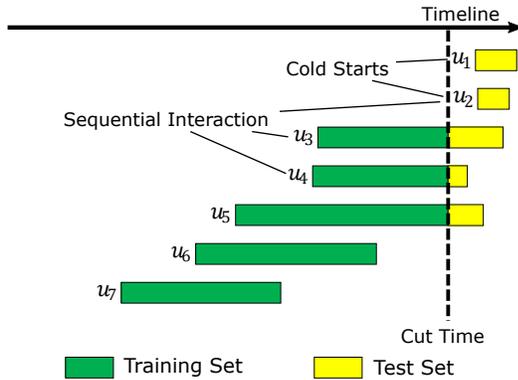
Figure 3: Dataset divided with a cut time.

architectures. After DBL, we can generate the prediction of next item representation $\widehat{v}_t$, which is a d-dimensions vector. In the global learning stage, we use Mean Squared Error (MSE) loss [37] function to learn two behavior alignments jointly from the whole set of sequential interactions H, which can be defined as:

$$\mathcal{L} = \frac{1}{|H|} \sum_{S_u \in H} \frac{1}{(|S_u| - ts - 1)} \sum_{t=ts+1}^{|S_u|} \zeta(\widehat{v}_t, v_t), \qquad (10)$$

where ζ is MSE function, v_t is the item representation that the target user u is access in the next visit, ts is the controlling indicator, $|S_u|$ denotes the length of the interaction sequence $S_u \in H$ and $|H|$ is the number of users.

The Eq. (10) is minimized using Adagrad optimization [7]. More details of settings will be specified in experiments.

So far, we have discussed the whole training stage of BINN. After obtaining the trained BINN model, in the testing stage, given an individual interaction history $S_u = \{(x_1, b_1), (x_2, b_2), ..., (x_{t-1}, b_{t-1})\}$, we could predict item x_t that user u is most likely to access in next visit by the following steps: (1) apply model BINN to fit user interaction process S_u to get the user's states Ψ_{SBL} and Ψ_{PBL} at $t-1$ step for prediction; (2) generate the next-item embedding vector $\widehat{v}_t$ and compute the similarities to all the item candidates in the latent space which we have obtained in Section 3.2; (3) then we can recommend the top-k potentially preferred items in the unified representation space.

4 EXPERIMENTS

In this section, we first describe the experimental setups. Then, we demonstrate the effectiveness of proposed framework from the following aspects: (1) the visualization of embedding comparisons between w-item2vec in BINN and traditional item2vec, (2) the comparisons of overall recommendation performances, (3) the analysis on cold-start scenarios and (4) parameter sensitiveness of user interactions.

4.1 Datasets

Specifically, we conduct experiments on two real-world datasets, i.e., Tianchi dataset and JD dataset.

Table 1: Statistics of datasets after preprocessing.

Statistics	Tianchi	JD
# of users	19,502	102,683
# of items	674,326	24,744
# of behaviors	8,799,573	37,061,992
# of behavior types	4	6
Avg. behaviors per user	451.30	360.94
Avg. behaviors per item	13.05	1,497.82
# of behaviors in training set	7,874,102	31,811,364
# of behaviors in test set	925,471	5,250,646

- **Tianchi**[1] is a public dataset by Alibaba's competition of *Ali Mobile Recommendation Algorithm*, which is based on the real users-commodities behavior data on Alibaba's M-Commerce platforms. It provides 23,291,027 interactions of 20,000 customers on 4,758,484 items within a month. In this dataset, customer behaviors include click, collect, cart and purchase, the corresponding values are 1, 2, 3 and 4, respectively.

- **JD** is provided by a Chinese e-commerce company Jingdong[2], which is one of the top two largest B2C online retailers in China by transaction volume and revenue. Specially, it provides 37,087,895 interactions of 105,180 customers on 28,710 items within 75 days based on the real log data of users-commodities behaviors. In this dataset, customer behaviors include browse, cart, delete-to-cart, purchase, collect and click with the corresponding values 1, 2, 3, 4, 5 and 6, respectively.

For the reliability of experimental results, we make the necessary preprocessing as follows. First, we filter the users whose interaction lengths are less than 10 and items that appear less than 5 times. Then we respectively divide two datasets into training sets and test sets according to cut time, where 90% interactions are chosen into the training set and the remaining interactions are used for testing. Figure 3 illustrates strategy of dataset partitioning. In particular, for Tianchi dataset, we use 27 days data for training and the rest 3 days as test set while for JD dataset, we use 68 days data for training and the rest for test. Considering that collaborative filtering methods can not recommend an item which has not appeared before [15], we filter out interactions from test set where items do not appear in the training set. In the same way, we also remove users from test set who do not appear in the training set, but we take special use of this part for analysis on the cold-start scenarios. The statistics of two datasets after preprocessing are shown in Table 1.

4.2 Baseline Methods

We compare BINN with three traditional methods (i.e., S-POP, BPR-MF, Item-KNN) and two state-of-the-art RNN-based models (i.e., GRU4Rec, HRNN) each of which contains two specific implements (i.e., GRU4Rec, GRU4Rec Concat and HRNN Init, HRNN All).

- **S-POP** recommends the item with the largest number of interactions by the target user. This method works well in the context with high repetitiveness, and the recommendation list changes along with user interactions.
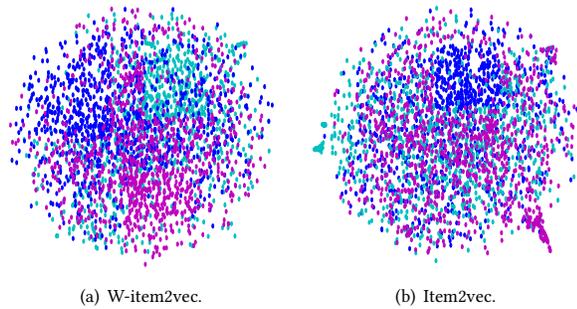
(a) W-item2vec.   (b) Item2vec.

Figure 4: T-SNE embedding for item vectors produced by w-item2vec (a), item2vec (b) on Tianchi dataset. The items are colored according to the categories.

- **BPR-MF [28]** is one of widely used matrix factorization methods, which optimizes a pairwise ranking objective function via stochastic gradient descent.

- **Item-KNN [20]** selects the items which are similar to the previously accessed items to users.

- **GRU4Rec [15]** uses the basic GRU with a TOP1 loss function and session-parallel minibatching.

- **GRU4Rec Concat [15]** is similar with GRU4Rec. Differently, we do not use the session-partition, and the users' interaction sequences are fed to the GRU4Rec independently as a whole.

- **HRNN Init [27]** is a hierarchical RNN for personalized cross-session recommendations, which is based on GRU4Rec and adds an additional GRU layer to model information across the user's sessions for tracking the evolution of the user's interest. It is a state-of-the-art method in next-item recommendations.

- **HRNN All [27]** is similar with HRNN, but the user representation generated by an additional GRU layer is used for initialization and propagated in input as each step of the next session.

For fair comparisons, we set all the hidden units in the RNN-based models as 100, their dropout probabilities and learning rate as 0.1. The embedding vector for each item is 64-dimensional in BINN model. The BINN and all the compared methods are defined and trained on a Linux server with two 2.20 GHz Intel Xeon E5-2650 v4 CPUs and four Tesla K80 GPUs.

4.3 Evaluation Metrics

As recommender systems can suggest few items each time, and the relevant items should be ranked first in the recommendation list. We therefore evaluate the personalized next-item recommendation quality with the following two evaluation metrics.

- **Recall@20**. It is the primary evaluation metric that is the proportion of cases having the desired item amongst the top-20 items in all test cases [15, 19]. Note that Recall@20 does not discriminate between items with different rankings as long as they are amongst the recommended list. In other word, the rank of items in top-20 candidate set do not make a difference.

- **MRR@20**. Another used metric is Mean Reciprocal Rank (MRR), which is the average of reciprocal ranks of the desire items. The same with Recall metric, we set 20 as contributing value, that
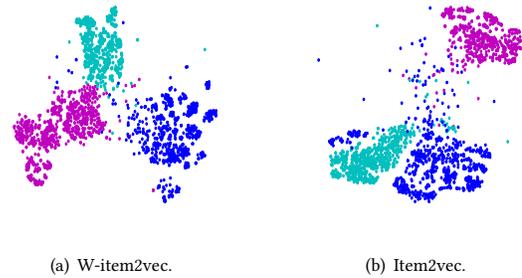

(a) W-item2vec.   (b) Item2vec.

Figure 5: T-SNE embedding for item vectors produced by w-item2vec (a), item2vec (b) on JD dataset. The items are colored according to the categories.

means the reciprocal rank is set to zero if the rank is above 20 [15, 19]. Considering the the order of recommendations matters, MRR takes the rank of each recommended item into consideration.

In summary, the higher both two evaluate metrics are, the better performances the results have.

4.4 Experimental Results

We first visualize the embedding of our proposed w-item2vec competing with item2vec, and then we show performances on next-item recommendation task. Finally, we discuss the cold-start problem of new users in recommendations and we analyze influences of the interaction lengths.
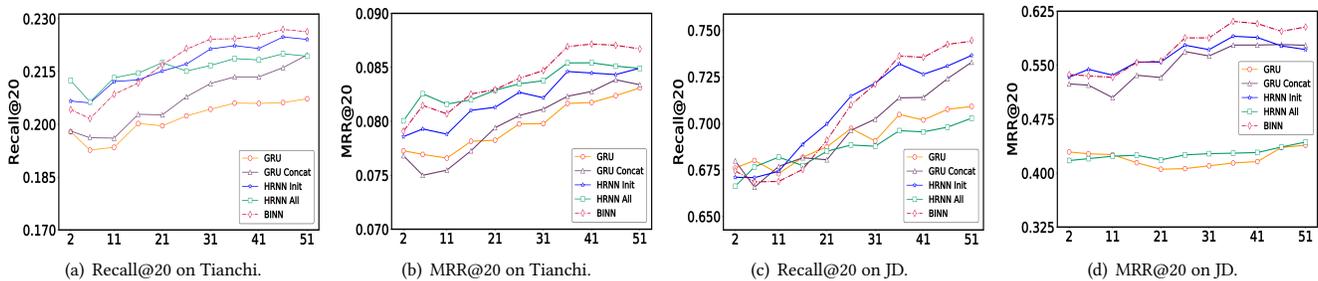
4.4.1 Item Embedding Visualization. We apply w-item2vec for generating the item embedding v_j for each item x_j from Eq. (5), in which item similarities and sequential-behavior relationships over items can be revealed simultaneously. We run the algorithm for 10 epochs and set the negative sampling value $E = 10$ and compare our method with item2vec [1] on both datasets. We apply the same settings for them.

Since an effective representation method can make the items implied similar attractions tend to be close to each other, we use the item categories to visualize whether the latent representations can reveal the similarities of the items. This is motivated by the assumption that a useful representation would cluster similar items in accordance with their category. To this end, we generate embeddings for 3,000 items which are randomly selected from three categories. We apply t-SNE [23] with a squared euclidean kernel to reduce the dimensionality of item embedding vectors to 2. Then we color each item point according to its category.

Figure 4 and 5 present the 2D embedding that are produced by t-SNE, for w-item2vec and item2vec, respectively. As we can see, w-item2vec provides a significantly better clustering on Tianchi and also shows better performance than item2vec on JD since the clustering boundaries in Figure 4(a) and Figure 5(a) are more clear. One possible reason is that w-item2vec takes account of the item frequencies, which makes w-item2vec can generate better representation of items than item2vec. Interestingly, both two methods have shown remarkable results on JD dataset, one possible explanation could be that JD dataset is more dense. As shown in Table 1, JD dataset log much more behavioral interactions on a smaller amount

Table 2: Performance comparisons of BINN with baseline methods on two datasets (The improvements of RNN-based models over the best traditional method have been marked).

Methods	Tianchi		JD	
	Recall@20	MRR@20	Recall@20	MRR@20
P-POP	0.2262	0.0824	0.5854	0.2176
BPR-MF	0.0559	0.0165	0.1873	0.0664
Item-KNN	0.1964	0.0883	0.1246	0.0361
GRU4Rec	0.2025(-10.48%)	0.0861(-2.49%)	0.7034(+20.16%)	0.4198(+92.92%)
GRU4Rec Concat	0.2287(+1.11%)	0.0859(-2.72%)	0.7934(+35.53%)	0.5932(+172.61%)
HRNN Init	0.2305(+1.9%)	0.0897(+1.59%)	0.8073(+37.91%)	0.6098(+180.23%)
HRNN All	0.2167(-4.20%)	0.0893(+1.13%)	0.7762(+32.59%)	0.4335(+99.22%)
BINN	**0.2376(+5.04%)**	**0.0936(+6.00%)**	**0.8430(+44.00%)**	**0.7082(+225.46%)**

(a) Recall@20 on Tianchi.   (b) MRR@20 on Tianchi.   (c) Recall@20 on JD.   (d) MRR@20 on JD.

Figure 6: Recommendation performances of new users cold-start on two datasets.

of items and the average behaviors per item is 1,497.82, which is much larger than Tianchi dataset 13.05 behaviors per item. That makes the model could be trained better. We further observe some outlier items that because many items on either dataset in the same category are not similar to each others.

4.4.2 Recommendation Performances. To demonstrate the practical significance of our proposed model, we compare BINN with the other methods on the next-item recommendation task. The results of all methods on both Tianchi and JD datasets are shown in Table 2. For the convenience of comparing differences between traditional and RNN-based models, we highlight the improvements of RNN-based models over the best traditional method. From the overall views, our BINN model has achieved the best performances on both two datasets.

Firstly, for the results on Tianchi dataset, we have some interesting observations. BINN performs significantly better than all the other methods on both Recall@20 and MRR@20. The result indicates that BINN framework is good at dealing with personalized sequential information from the user interactions. Comparing with the RNN-based models, we can note that the traditional methods provide more competitive results. We guess a possible reason is that users' interactions on Tianchi have a high degree of repetitiveness and this dateset has a large amount of item candidates when making recommendations. That fact makes the generation of "non-trivial" personalized recommendations (i.e., P-POP) very challenging [27]. The comparison among the five RNN-based models highlights the effectiveness to track customers' long-term preferences for next

item recommendations, because models with consideration of personalized information (i.e., BINN, HRNN Init, HRNN All) outperform those methods without that (i.e., GRU4Rec, GRU4Rec Concat) on MRR@20.

Next, we turn to the experiments on JD dataset, which exhibits some different results from those on Tianchi dataset. All the RNN-based models (i.e., GRU, HRNN and BINN) significantly outperform the traditional methods, which indicates that RNN-based models do have better abilities to model users' sequential interactions for next-item recommendations than traditional methods. In addition, the non-personalized RNN-based models GRU4Rec Concat outperforms HRNN All, which indicates that improper personalizing strategy might even make the recommendation performances worse and reveals the importance of the community trends from the short-term sequential behaviors.

Then, we notice that RNN-based models perform much better on JD dataset than that on Tianchi dataset. One possible explanation could be that JD has more interactions of more users but less merchandises than Tianchi, and that may be a strong sequential recommendation scenario. Moreover, the number of users is much larger than the amount of items on JD dataset (i.e., the statistic of items on Tianchi dataset is 674,326, but 24,744 on JD dataset), that naturally makes the predicting candidate set smaller, and therefore, leads to more accurate result. Actually, this scenario is much common in the real world, such as online retailers and B2C platforms.

In summary, BINN achieves the best performances on both two datasets, followed by HRNN Init. Both two methods take user-level representations into account. That clearly demonstrates users' interactive behaviors may follow general short-term community trends
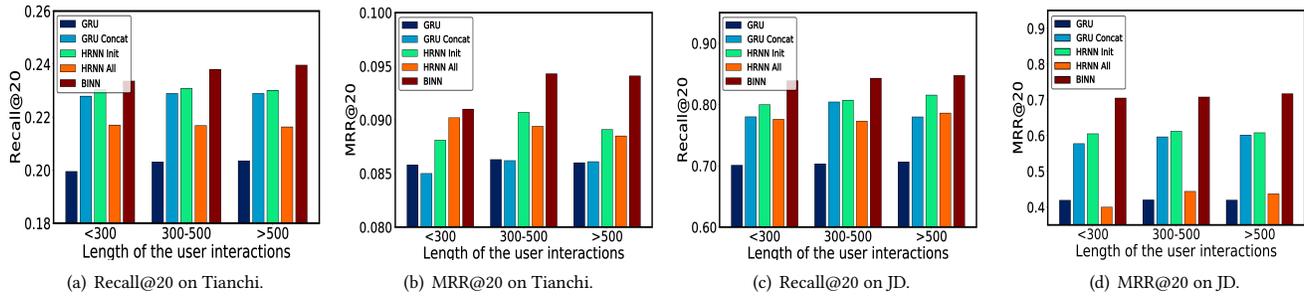
(a) Recall@20 on Tianchi. (b) MRR@20 on Tianchi. (c) Recall@20 on JD. (d) MRR@20 on JD.

Figure 7: Results of next-item recommendation over different history lengths.

and reveal stable long-term preferences. Our BINN discriminatively models the users' historical preferences and present motivations, that leads to superior personalized recommendation quality.

4.4.3 Cold Start of New Users. Cold start is a common problem of recommender systems that new users or items have not yet gathered sufficient information to recommend or be recommended [30]. As we have removed users from test set that are not in the training set on the above experiments, which is shown in Figure 3 shown, here we focus on these users and examine the performances of our model on cold-start problem of new users.

Indeed, new users have no interactions to be pretrained and recommender systems cannot generate user profiles. That makes many user profile-based recommendation methods cannot work well, especially factorization models. However, for the RNN-based next-item recommendations, we can use a trained neural network to fit new users and predict from their second interactions one-by-one and check item rank of the next interaction. Here, we test the recommendation results on fifty items from the beginning of the second ones in the interactions of the target new users. Please note that, we do not change any training process and just select cold-start users for testing, thus all the testing do not need retraining. For better illustration, we report the results of all RNN-based models, using both metrics, respectively.

The results are shown as Figure 6. In most cases, BINN performs better than the other models on both datasets. At the beginning of the user interactions, our model BINN has deteriorated to CLSTM because of the absence of personalized preferences. Then, with the number of users' interactions growing, our model shows great improvement on recommendation performances. That can indicate the effectiveness of modeling the long-term historical preferences in BINN. What's more, all the deep learning models have shown strong capacity to face the cold-start challenges of new users. Thus, we can conclude that all the RNN-based models can work well for new users. Consequently, the results indicate the effectiveness of BINN structure.

4.4.4 Analysis on the User History Length. Here, we take further analysis on performances of our BINN model and other RNN-based models. This allows to evaluate personalized recommendation methods under different amounts of historical information and reveal the capacity of users' long-term preference representations. Since we argue the length of the user history has an impact on the recommender system performances, we divide the evaluation by the

length of user interactions. Specially, we use the both datasets to make the analysis and partition users into three groups: the historical interactions less than 300, between 300 and 500, and more than 500. On account of our purpose for measuring the impact of the complex long-term preference dynamics used in BINN and other RNN-based models, we respectively record performances on these three user groups.

Figure 7 shows the performances on both datasets. Firstly, we pay attention to improvements over the length of user behaviors grows on Tianchi dataset. We can notice that our proposed BINN has a significantly improvement on Recall@20 as the history lengths grow. Then, we turn to the results on JD dataset. For users with largest amount of interactions, our proposed BINN performs best with at least 3.92% improvement compared to other RNN-based models. Interestingly, BINN and both two HRNN models have an improvement with the history length growing, but GRU4Rec and GRU4Rec Concat do not show continuous improvement between 300-500 and larger than 500 scales. In summary, the length of the user interactions does have an impact on the performances of recommender systems. These results clearly demonstrate the effectiveness of exploiting personalized strategies, i.e., users' historical stable preferences, to improve the recommendation performances.

5 CONCLUSIONS AND FUTURE WORKS

In this paper, we proposed a novel solution framework, the Behavior-Intensive Neural Network, to address the problem of personalized next-item recommendations. As a user's behaviors naturally form a interactive sequence over time, the user's historical preferences from the long-term view and present motivations from the short-term view can be dynamically revealed. Along this line, we first introduced a w-item2vec method to generate item representations by considering the sequential similarities of the superb items. Then we discriminatively exploited user behaviors and proposed two alignments of the behaviors. Specific to each alignment, we respectively developed LSTM-based neural networks to learn personal historical preferences and present consumption motivations. Finally, we conducted extensive experiments on two industrial datasets. The experimental results clearly demonstrated the effective of our proposed model in personalized next-item recommendations.

In the future, we plan to study the impact of different types of user behaviors to generate user representations and improve the next-item recommendation even further. We also plan to investigate our model in other domain, such as advertisements.

ACKNOWLEDGMENTS

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on.* IEEE, 1–6.

[2] Robert M Bell and Yehuda Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on.* IEEE, 43–52.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[4] Peng Cui, Fei Wang, Shaowei Liu, Mingdong Ou, Shiqiang Yang, and Lifeng Sun. 2011. Who should share what?: item-level social influence prediction for users and posts ranking. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval.* ACM, 185–194.

[5] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2017. A Survey on Network Embedding. *arXiv preprint arXiv:1711.08752* (2017).

[6] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17).* ACM, New York, NY, USA, 152–160.

[7] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[8] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15).* 278–288.

[9] Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. 2016. Contextual lstm (clstm) models for large scale nlp tasks. *arXiv preprint arXiv:1602.06291* (2016).

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 855–864.

[11] Huifeng Guo, Ruiming TANG, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17.* 1725–1731.

[12] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback.. In *AAAI.* 144–150.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17).* 173–182.

[14] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval.* ACM, 549–558.

[15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 426–434.

[18] Yehuda Koren. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53, 4 (2010), 89–97.

[19] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17).* 1419–1428.

[20] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.

[21] Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. 2011. Personalized travel package recommendation. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on.* IEEE, 407–416.

[22] Qi Liu, Xianyu Zeng, Hengshu Zhu, Enhong Chen, Hui Xiong, Xing Xie, et al. 2015. Mining indecisiveness in customer behaviors. In *Data Mining (ICDM), 2015 IEEE International Conference on.* IEEE, 281–290.

[23] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems.* 3111–3119.

[25] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems.* 1257–1264.

[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 701–710.

[27] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17).* 130–137.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence.* AUAI Press, 452–461.

[29] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web.* ACM, 811–820.

[30] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. 2015. *Recommender systems handbook.* Springer.

[31] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web.* ACM, 285–295.

[32] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[33] Shuo Shang, Ruogu Ding, Bo Yuan, Kexin Xie, Kai Zheng, and Panos Kalnis. 2012. User Oriented Trajectory Search for Trip Recommendation. In *Proceedings of the 15th International Conference on Extending Database Technology (EDBT '12).* ACM, New York, NY, USA, 156–167.

[34] Shuo Shang, Ruogu Ding, Kai Zheng, Christian S Jensen, Panos Kalnis, and Xiaofang Zhou. 2014. Personalized trajectory matching in spatial networks. *The VLDB Journal* 23, 3 (2014), 449–468.

[35] Jun Wang and Qiang Tang. 2016. A probabilistic view of neighborhood-based recommendation methods. In *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on.* IEEE, 14–20.

[36] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning Hierarchical Representation Model for NextBasket Recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15).* ACM, New York, NY, USA, 403–412.

[37] Bo Wu, Wen-Huang Cheng, Yongdong Zhang, Qiushi Huang, Jintao Li, and Tao Mei. 2017. Sequential Prediction of Social Media Popularity with Deep Temporal Context Networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17.* 3062–3068.

[38] Bo Wu, Tao Mei, Wen-Huang Cheng, Yongdong Zhang, et al. 2016. Unfolding Temporal Dynamics: Predicting Social Media Popularity Using Multi-scale Temporal Decomposition.. In *AAAI.* 272–278.

[39] Ghim-Eng Yap, Xiao-Li Li, and Philip Yu. 2012. Effective next-items recommendation via personalized sequential pattern mining. In *Database Systems for Advanced Applications.* Springer Berlin/Heidelberg, 48–64.

[40] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining.* ACM, 353–362.

[41] Hongke Zhao, Qi Liu, Yong Ge, Ruoyan Kong, and Enhong Chen. 2016. Group Preference Aggregation: A Nash Equilibrium Approach. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on.* IEEE, 679–688.

[42] Hongke Zhao, Qi Liu, Hengshu Zhu, Yong Ge, Enhong Chen, Yan Zhu, and Junping Du. 2017. A sequential approach to market state modeling and analysis in online p2p lending. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2017).

[43] Hongke Zhao, Le Wu, Qi Liu, Yong Ge, and Enhong Chen. 2014. Investment recommendation in p2p lending: A portfolio perspective with risk management. In *Data Mining (ICDM), 2014 IEEE International Conference on.* IEEE, 1109–1114.