Sequential Recommender System based on Hierarchical Attention Network

submitted for blind review

Abstract

With a large amount of user activity data accumulated, it is crucial to exploit user sequential behavior for sequential recommendations. Conventionally, user general taste and recent demand are combined to promote recommendation performances. However, existing methods often neglect that user long-term preference keep evolving over time, and building a static representation for user general taste may not adequately reflect the dynamic characters. Moreover, they integrate user-item or item-item interactions through a linear way which limits the capability of model. To this end, in this paper, we propose a novel two-layer hierarchical attention network, which takes the above properties into account, to recommend the next item user might be interested. Specifically, the first attention layer learns user long-term preferences based on the historical purchased item representation, while the second one outputs final user representation through coupling user long-term and short-term preferences. The experimental study demonstrates the superiority of our method compared with other state-of-the-art ones.

1 Introduction

With the emergence of platform economy, many companies like Amazon, Yelp, and Uber, are creating self-ecosystems to retain users through interaction with products and services. Users can easily access these platforms through mobile devices in daily life, as a result large amounts of behavior logs have been generated. For instance, 62 million user trips have been accumulated in July 2016 at Uber, and more than 10 billion check-ins have been generated by over 50 million users at Foursquare. With such massive user sequential behavior data, sequential recommendation, which is to recommend the next item user might be interested, has become a critical task for improving user experience and meanwhile driving new value for platforms.

Different from traditional recommender systems, there are new challenges in sequential recommendation scenarios. First, user behaviors in the above examples only reflect their implicit feedbacks (e.g., purchased or not), other than explicit feedbacks (e.g., ratings). This type of data brings more noises because we cannot differentiate whether users dislike unobserved items or just do not realize them. Therefore, it is not appropriate to directly optimize such one-class score (i.e., 1 or 0) through conventional latent factor model [Bayer *et al.*, 2017]. Second, more and more data is originated from sessions or transactions, which form user's sequential pattern and short-term preference. For instance, users prefer resting at hotels than sporting after they leave the airport, while after buying a camera, customers choose purchasing relevant accessories rather than clothes. However, previous methods mainly focus on user general taste and rarely consider sequential information, which leads to repeated recommendations [Hu *et al.*, 2017; Ying *et al.*, 2016; Zhang *et al.*, 2016].

In the literature, researchers usually employ separate models to characterize user's long-term preference (i.e, general taste) and short-term preference (i.e., sequential pattern), and then integrate them together [Rendle *et al.*, 2009; Feng *et al.*, 2015; He and McAuley, 2016]. For example, Rendel et al. [Rendle *et al.*, 2010] propose factoring personalized Markov chains for next basket prediction. They factorize observed user-item matrix to learn user's long-term preference and utilize item-item transitions to model sequential information, and then linearly add them to get final scores. However, these models neglect the dynamics of user general taste, which means user's long-term preference keep evolving over time. It is not adequate to learn a static low-rank vector for each user to model her general taste. Moreover, they mainly assign fixed weights for user-item or item-item interactions through linear modeling, which limits the model capability. It has been shown that nonlinear models can better model the user-item interaction in user activities [He and Chua, 2017; Xiao *et al.*, 2017; Cheng *et al.*, 2016].

To this end, we propose a novel approach, namely Sequential Hierarchical Attention Network (SHAN), to solve the next item recommendation problem. The attention mechanism can automatically assign different influences (weights) of items for user to capture the dynamic property, while the hierarchical structure combines user's long- and short-term preferences. Specifically, we first embed users and items into low-dimensional dense spaces. Then an attention layer is employed to compute different weights of items in user long-term set and then compress item vectors with weights to gen-

erate user long-term representation. After that, we use another attention layer to couple user sequential behavior with long-term representation. User embedding vector is used as context information in both attention networks to compute different weights for different users. To learn the parameters, we employ the Bayesian personalized ranking optimization criterion to generate a pair-wise loss function [Rendle *et al.*, 2009]. From the experiments, we can observe that our model outperforms state-of-the-art algorithms on two datasets. Finally, our contributions are summarized as follows:

- We introduce the attention mechanism to model user dynamics and personal preferences for sequential recommendations.
- Through the hierarchical structure, we combine user's long- and short-term preferences to generate a high-level hybrid representation of user.
- We perform experiments on two datasets which show our model consistently outperforms state-of-the-art methods in terms of Recall and Area Under Curve.

2 Related Work

To model user's individual and sequential information jointly, Markov chains have been introduced by previous work for traditional recommendations. [Rendle *et al.*, 2010] combined factorization method to model user general taste and Markov chains to mine user sequential pattern. Following this idea, researchers have utilized different methods to extract these two different user preferences. [Chen *et al.*, 2012] and [Feng *et al.*, 2015] used metric embedding to project items into points in a low-dimension Euclidean space for play list prediction and successive location recommendation. [Liang *et al.*, 2016] utilized word embedding to extract information from item-item co-occurrence to improve matrix factorization performance. However, these methods have limited capacity on capturing high-level user-item interactions, because the weights of different components are fixed.

Recently, researchers turn to graphical models and neural networks in recommender systems. [Liu *et al.*, 2016] proposed a bi-weighted low-rank graph construction model, which integrates users' interests and sequential preferences with temporal interval assessment. [Cheng *et al.*, 2016] combined wide linear models with cross-product feature transformations and employed deep neural network to learn highly nonlinear interactions between feature embeddings. However, this model needs feature engineering to design cross features, which can be rarely observed in real data with high sparsity. To deal with this problem, [He and Chua, 2017] and [Xiao *et al.*, 2017] designed B-Interaction and attentional pooling layers, respectively, to automatically learn second-order feature interaction based on traditional factorization machine technology. [Hidasi *et al.*, 2015] and [Wu *et al.*, 2017] employed recurrent neural network (RNN) to mine dynamic user and item preferences in trajectory data. However, items in a session may not follow rigidly sequential order in many real scenarios, e.g., transactions in online shopping, where RNN is not applicable. Beyond that, [Wang *et al.*, 2015] and [Hu *et al.*, 2017] learned user hierarchical representation to combine user long- and short-term preferences. Our work follows this pipeline but contributes in that: (1) Our model is built on hierarchical attention networks, which can caputure dynamic long- and short-term preferences. (2) Our model utilizes nonlinear modeling of user-item interactions. It is able to learn different item influences (weights) of different users for the same item.

3 Sequential Hierarchical Attention Network

In this section, we first formulate our next item recommendation problem and then introduce the details of our model. Finally, we present the optimization procedures.

3.1 Problem Formulation

Let $\mathcal{U}$ denote a set of users and $\mathcal{V}$ denote a set of items, where $|\mathcal{U}|$ and $|\mathcal{V}|$ are the total numbers of users and items, respectively. In this work, we focus on extracting information from implicit, sequential user-item feedback data (e.g., users' successive check-ins and purchase transaction records). For each user $u \in \mathcal{U}$, his/her sequential transactions (or sessions) are denoted as $L^u = \{S_1^u, S_2^u, ..., S_T^u\}$, where T is the total number of time steps and $S_t^u \subseteq \mathcal{V}$ ($t \in [1, T]$) represents the item set corresponding to the transaction of user u at time step t. For a fixed time step t, the item set S_t^u can reflect user u's short-term preference at time t, which is an important factor for predicting the next item he/she will purchase. On the other hand, the set of items purchased before time step t, denoted as $L_{t-1}^u = S_1^u \cup S_2^u \cup ... \cup S_{t-1}^u$, can reflect user u's long-term preference (i.e., general taste). In the following, we name L_{t-1}^u and S_t^u the long- and short-term item sets w.r.t time step t, respectively.

Formally, given users and their sequential transactions L, we aim to recommend the next items users will purchase based on long- and short-term preferences learned from L.

3.2 The Network Architecture

We propose a novel approach based on hierarchical attention network, as shown in Figure 1, according to the following characteristics of user preference. 1) User preference is dynamic at different time steps. 2) Different items have different influences on the next item that will be purchased. 3) For different users, same items may have different impacts on the next item prediction.

The basic idea of our approach is to generate a hybrid representation for each user through jointly learning the long- and short-term preferences. More specifically, we first embed sparse user and item inputs (i.e., one-hot representations) into low-dimensional dense vectors, which endow each user or item an informative representation instead of the basic index. After that, a hybrid representation for each user is learned through a two-layer structure, which combines both the long- and short-term preferences. To capture the long-term preference before time step t, we learn a long-term user representation, which is a weighted sum over the embeddings of items in the long-term item set L_{t-1}^u, while the weights are inferred by an attention-based pooling layer guided by the user embedding. To further incorporate the short-term preference, the final hybrid user representation combines the long-term user representation with the embeddings of items in the short-term

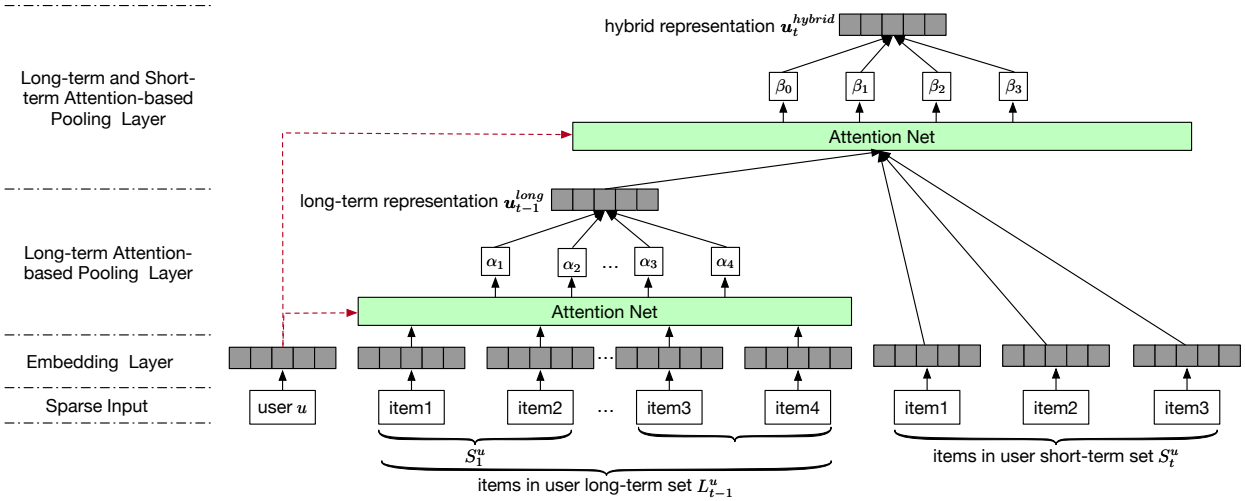


Figure 1: The architecture of our model. A hybrid user representation is learned by a sequential hierarchical attention network, which combines both the long- and short-term user preference.

item set, where the weights are learned by another attention-based pooling layer.

As shown above, our model simultaneously considers the dynamic long- and short-term user preferences. Moreover, they have different influences on the next item to be purchased by using the different learned weights. Finally, it is worthy pointing out that impacts of same items are also different on different users, since the leaning procedure of weights in attention layers is guided by the user embedding. Next we introduce each part of our model in details.

Embedding Layer. Similar to discrete word symbols in natural language processing, the original user and item IDs have very limited representation capacity. Therefore, our model first employs a fully connected layer to embed user and item IDs (i.e., one-hot representations) into two continuous low-dimensional spaces. Formally, let $\boldsymbol{U} \in \mathbb{R}^{K \times |\mathcal{U}|}$ and $\boldsymbol{V} \in \mathbb{R}^{K \times |\mathcal{V}|}$ be two matrices consisting of the user and item embeddings, respectively, where K is the dimensionality of the latent embedding spaces. Theoretically, traditional matrix factorization is equivalent to a two-layer neural network, which constructs low-rank embeddings for users and items in the first layer and employs inner product operation in the second layer. However, embeddings through matrix factorization only capture low-level, bi-linear and static representation, which limits the representation capability [Liang *et al.*, 2016]. Differently, our model learns the dynamic and high-level user representations based on these basic embeddings as will be explained later.

Long-term Attention-based Pooling Layer. In sequential recommender systems, long- and short-term preferences correspond to users' general taste and sequential behavior, respectively [Rendle *et al.*, 2010]. Since the long-term item set of a user usually changes over time, learning a static long-term preference representation for each user cannot fully express the dynamics of long-term user preference. On the other hand, reconstructing long-term user representations from the up-to-date long-term item set is more reasonable. Moreover, we argue that the same items might have different impacts on different users. For instance, assume that user a buys item x for himself because of interest, while user b buys item x as a gift for others. In such a case, it is reasonable to infer that item x has different weights or attentions on users a and b when predicting their next items.

To meet the above requirements, we propose to use the attention mechanism which has been successfully applied in many tasks, such as image question answering [Yang *et al.*, 2016a], document classification [Yang *et al.*, 2016b] and recommendation [Xiao *et al.*, 2017]. It first computes the importance of each item in the long-term item set of a given user, and then aggregates the embedding of these items to form the long-term user preference representation. Formally, the attention network is defined as:

$$\boldsymbol{h}_{1j} = \phi(\boldsymbol{W}_1 \boldsymbol{v}_j + \boldsymbol{b}_1), \tag{1}$$

$$\alpha_j = \frac{\exp(\boldsymbol{u}^\top \boldsymbol{h}_{1j})}{\sum_{p \in L_{t-1}^u} \exp(\boldsymbol{u}^\top \boldsymbol{h}_{1p})}, \tag{2}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{K \times K}$ and $\boldsymbol{b}_1 \in \mathbb{R}^{K \times 1}$ are model parameters. We assume that the items are consecutively labeled from 1 to $|\mathcal{V}|$, and $\boldsymbol{v}_j$ represents the dense embedding vector of item j. We first feed the dense low-dimensional embedding of each item $j \in L_{t-1}^u$ through a multi-layer perceptron (MLP) to get the hidden representation $\boldsymbol{h}_{1j}$. Function $\phi(\cdot)$ is the activation function and we utilize RELU to enhance nonlinear capability. Unlike traditional attention models that use the same context vectors for each input, we put the embedding $\boldsymbol{u}$ of user u as the context vector and measure the attention score α_j as the normalized similarity between $\boldsymbol{h}_{1j}$ and $\boldsymbol{u}$ with the softmax function, which characterizes the importance of item j for user u. Finally, we compute the long-term user representation $\boldsymbol{u}_{t-1}^{long}$ as a sum of the item embeddings weighted by the attention scores as follows:

$$\boldsymbol{u}_{t-1}^{long} = \sum_{j \in L_{t-1}^u} \alpha_j \boldsymbol{v}_j \tag{3}$$

Long- and Short-term Attention-based Pooling Layer. In addition to user's general taste, sequential behavior (i.e.,

short-term preference) is also incorporated. Short-term preference is important for predicting next items, and there has been studies on combining long- and short-term preference for sequential recommendation [Rendle *et al.*, 2010; He and McAuley, 2016]. However, the interaction of short- and long-term preference remains linear and items are assigned with the same weights in earlier work, which cannot reflect the characteristics of item impacts on next item prediction and, hence, limit the model performance. Similar to modeling user long-term preference, we also turn to attention networks, assigning weights to long-term representations and embeddings of items in the short-term item set, to capture the high-level representation of user u. Formally,

$$\boldsymbol{h}_{2j} = \phi(\boldsymbol{W}_2 \boldsymbol{x}_j + \boldsymbol{b}_2), \tag{4}$$

$$\beta_j = \frac{\exp(\boldsymbol{u}^\top \boldsymbol{h}_{2j})}{\sum_{p \in S_t^u \cup \{0\}} \exp(\boldsymbol{u}^\top \boldsymbol{h}_{2p})}, \tag{5}$$

where $\boldsymbol{W}_2 \in \mathbb{R}^{K \times K}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{K \times 1}$ are model parameters, and $\boldsymbol{x}_j$ represents the embedding of item $j \in S_t^u$ when $j > 0$ and $\boldsymbol{x}_j = \boldsymbol{u}_{t-1}^{long}$ when $j = 0$. Similarly, the user embedding is applied as the context vector to achieve the goal of personality (i.e., assigning different weights of same items to different users). After obtaining the normalized attention scores, the hybrid user representation is calculated as follows:

$$\boldsymbol{u}_t^{hybrid} = \beta_0 \boldsymbol{u}_{t-1}^{long} + \sum_{j \in S_t^u} \beta_j \boldsymbol{v}_j, \tag{6}$$

where β_0 is the weight of long-term user preference.

To conclude, $\boldsymbol{u}_t^{hybrid}$ considers not only the dynamic properties in long- and short-term preferences, but also differentiate contributions of items for predicting the next item. Moreover, two hierarchical attention networks can capture the nonlinear interaction between users and items. Note that [Wang *et al.*, 2015] can also achieve the goal of nonlinearity by using max pooling to aggregate final representations. However, it loses much information in the meanwhile. We will experimentally demonstrate that our model can achieve better performance than [Wang *et al.*, 2015].

3.3 Model Inference

After computing user hybrid representation $\boldsymbol{u}_t^{hybrid}$, we employ the traditional latent factor model to compute his preference score of item j as follows:

$$R_{ujt} = \boldsymbol{u}_t^{hybrid} \boldsymbol{v}_j. \tag{7}$$

However, user transaction records are a type of implicit data. It is difficult to directly optimize the preference score R_{ujt} because of the data sparsity problem and the ambiguity of unobserved data [Pan *et al.*, 2008].

The goal of our model is to provide a ranked list of items given the long- and short-term item sets of a user at time t. Therefore, we are more interested in the ranking order of items rather than the real preference scores. Following BPR optimization criterion [Rendle *et al.*, 2009], we propose a pair-wise ranking objective function for our model. We assume that users prefer the next purchased items than other unobserved items, and define a ranking order $\succ_{u, L_{t-1}^u, S_t^u}$ over items j and k as follows:

$$j \succ_{u, L_{t-1}^u, S_t^u} k \Leftrightarrow R_{ujt} > R_{ukt}, \tag{8}$$

where j is the purchased next items by user u at time step t, and k is an unobserved item generated by bootstrap sampling. For each observation (u, L_{t-1}^u, S_t^u, j), we generate a set of pairwise preference orders $D = \{(u, L_{t-1}^u, S_t^u, j, k)\}$. Then we train our model by maximizing a posterior (MAP) as follows:

$$\begin{aligned} \arg\min_{\Theta} \sum_{(u, L_{t-1}^u, S_t^u, j, k) \in D} & -\ln \sigma(R_{ujt} - R_{ukt}) \\ & + \lambda_{uv} ||\Theta_{uv}||^2 + \lambda_a ||\Theta_a||^2, \end{aligned} \tag{9}$$

where $\Theta = \{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2\}$ is the set of model parameters, σ is the logistic function, $\Theta_{uv} = \{\boldsymbol{U}, \boldsymbol{V}\}$ is the set of embeddings of users and items, $\Theta_a = \{\boldsymbol{W}_1, \boldsymbol{W}_2\}$ is the set of weights in attention networks, and $\lambda = \{\lambda_{uv}, \lambda_a\}$ is the regularization parameters. The detailed learning algorithm is presented in Algorithm 1.

Algorithm 1: SHAN Algorithm

Input: long-term item set L, short-term item set S, learning rate η, regularization λ, number of dimensions K
Output: model parameters Θ

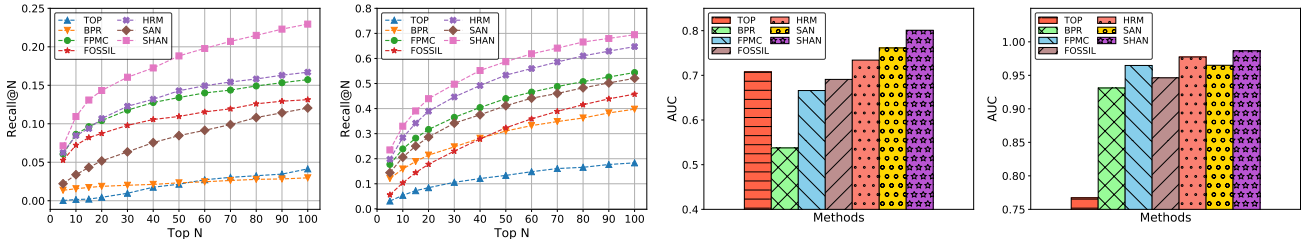
Draw Θ_{uv} from Normal Distribution $N(0, 0.01)$;
Draw Θ_a from Uniform Distribution $[-\sqrt{\frac{3}{K}}, \sqrt{\frac{3}{K}}]$;
repeat
  shuffle the set of observations $\{(u, L_{t-1}^u, S_t^u, j)\}$
  for *each observation* (u, L_{t-1}^u, S_t^u, j) **do**
    Randomly draw an unobserved item k from $\mathcal{V} \backslash L_{t-1}^u$
    compute $\boldsymbol{u}_{hybrid}$ according to Equation (1) - (6)
    compute R_{ujt}, R_{ukt} according to Equation (7)
    update Θ with gradient descent
until *convergence*
return Θ

4 Experiments

In this section, we conduct experiments to answer the following questions: 1) what's the performance of our model as compared to other state-of-the-art methods? 2) what's the influence of long- and short-term preferences in our model? 3) how do the parameters affect model performance, such as the regularization parameters and the number of dimensions?

4.1 Experimental Setup

Datasets. We perform experiments on two real-world datasets, Tmall [Hu *et al.*, 2017] and Gowalla [Cho *et al.*, 2011], to demonstrate the effectiveness of our model. Tmall dataset accumulates user behavior logs in the largest online shopping site in China (i.e, Tmall.com), while Gowalla dataset records the time and point-of-interest information of check-ins from users in the location-based social networking



(a) Recall@N on Tmall (b) Recall@N on Gowalla (c) AUC on Tmall (d) AUC on Gowalla

Figure 2: Performance Comparison of Methods at Tmall and Gowalla Dataset.

Table 1: Statistics of datasets

Dataset	Tmall	Gowalla
#user	20,648	15,171
#item	25,129	13,193
avg. session length	2.73	2.97
#train session	71,892	129,225
#test session	3,534	3,635

site, Gowalla. We focus on the data generated in the last seven months on both datasets. Items which have been observed by less than 20 users during this period are removed. After that, user records in one day are treated as a session (i.e., a transaction) to represent the short-term preference, and all singleton sessions (i.e., contain only one item) are removed. Similar to [Hu *et al.*, 2017], we randomly select 20% of sessions in the last month for testing, and the rest are used for training. We also randomly hold out one item in each session as the next item to be predicted. After preprocessing, basic statistics of both datasets are summarized in Table 1.

Metrics. To evaluate the performance of each method for sequential recommendation problem, we employ two widely used metrics Recall@N and AUC. The first metric evaluates the fraction of ground truth items that have been rightly ranked over top-N items in all testing sessions, while the second metric evaluates how highly ground truth items have been ranked over all items. Note that larger metric values indicate better performances.

Baselines. We compare our model with the following baseline algorithms, including traditional classic next item recommendation method and one hierarchical representation method: (1) **TOP.** The top rank items based on popularity in training data are taken as recommendations for each session in test data. (2) **BPR.** BPR is a state-of-the-art framework for implicit user feedback data through pairwise learning to rank, and we choose matrix factorization as internal predictor. [Rendle *et al.*, 2009]. (3) **FPMC.** This method models user preference through matrix factorization and sequential information through first-order Markov chain simultaneously, and then combine them by linear way for next basket recommendation [Rendle *et al.*, 2010]. (4) **FOSSIL.** This method integrates factored item similarity with Markov chain to model user's long- and short-term preference. Note that we set η_u and η as single scalar since the length of each session is variable [He and McAuley, 2016]. (5) **HRM.** This method generates a user hierarchical representation to capture sequential information and general taste. We use max pooling as the aggregation operation because this reaches the best result [Wang *et al.*, 2015]. (6) **SHAN.** This is our proposed model, which employs two attention networks to mine long- and short-term preferences. We also show the performance of our simplified version, i.e., **SAN**, which ignores the hierarchical construction and computes the weights of items from long- and short-term sets through a single attention network. For fair comparison, all model-based methods optimize a pair-wise ranking objective function based on the BPR criterion.

4.2 Comparison of Performance

Figure 2 shows performances of all methods under the metric of recall from Top-5 to Top-100 and AUC in Tmall and Gowalla datasets. From the figure, we can observe that:

1. SHAN consistently outperforms all other methods under all measurements on Tmall Dataset with a large margin. Specifically, SHAN improves 33.6% and 9.8% at Recall@20 compared with the second best method (i.e., HRM) on Tmall and Gowalla datasets, respectively. This indicates that our model captures more high-level complicated nonlinear information for long- and short-term representations through attention network, while HRM may lose much information through hierarchical max pooling operation. In addition, the performance of SHAN is better than SAN, possibly because the number of items in long-term set is much more than that in short-term set. Therefore, it is hard for a single attention network to assign appropriate weights to fewer but more important items belong to short-term set.
2. HRM outperforms FPMC in most cases under the two measures generally. More specifically, the relative performance improvement by HRM is 6.7% and 4.5% in terms of Recall@50 on Tmall and Gowalla datasets, respectively. It demonstrates that interactions among multiple factors can be learned through max pooling operation [Wang *et al.*, 2015]. Furthermore, introducing nonlinear interaction will promote model capability despite through simple max pooling. Finally, our model achieves better performance than HRM, which indicates that attention network is more powerful than max pooling at modeling complex interactions.
3. All hybrid models (i.e., FPMC, FOSSIL, HRM and SHAN) outperform BPR on both datasets under different metrics in most cases. Taking AUC as an ex-

ample, the relative performance improvements continuously keep at a very high level. This demonstrates that sequential information is very important for our task, where BPR ignores it. Moreover, our model gets the best performance in these hybrid models.

4. Surprisingly, TOP method surpasses BPR when N increases from 50 in terms of recall and even performs better than FPMC under AUC. This phenomenon can be explained that users may tend to buy popular items in online shopping. Therefore, TOP method can reach better performance when N is large enough. On the contrary, because user check-in behavior is more personalized at Gowalla dataset, TOP method achieves much worse performance than other methods. Finally, our model can outperform all the baselines in terms of different Ns.

Table 2: Influence of components at AUC

Dataset	Method	AUC	Recall@20
Tmall	SHAN-L	0.701	0.026
	SHAN-S	0.763	**0.156**
	SHAN	**0.801**	0.143
Gowalla	SHAN-L	0.947	0.232
	SHAN-S	0.982	0.407
	SHAN	**0.987**	**0.439**

4.3 Influence of Components

To evaluate the contribution of each components for forming final user hybrid representation, we conduct experiments to analyze each component in Table 2. SHAN-L means only user's general taste is modeled, while SHAN-S only considers user's short-term preference.

SHAN-L achieves better performance compared with BPR which also only models long-term preference. For example, the Recall@20 values of BPR are 0.019 and 0.204 at Tmall and Gowalla datasets, respectively. This indicates that modeling general taste through the dynamic way is better than using a fixed embedding vector. SHAN-S outperforms SHAN-L in a large margin, which demonstrates that short-term sequential information is more important on predicting next item task. Surprisingly, SHAN-S outperforms HRM on both dataset. The AUC values of HRM are 0.734 and 0.966 at Tmall and Gowalla datasets, respectively. The reason may be that the basic user embedding vector on SHAN-S, fusing user basic preference, is learned for computing each weight of items in short-term set. Therefore, SHAN-S also considers user's static general taste and sequential information to generate hybrid representation. The result also indicates one-layer attention network is better than two-layer max pooling operation. SHAN-S performs a few better than SHAN at Recall@20 on Tmall dataset. This indicates that user click behaviors in previous session do not have much impacts for the next clicked item in current session. Finally, SHAN performs better than two single component models in most cases. It demonstrates that adding dynamic user's general taste to SHAN-S is helpful to predict next items. Because SHAN-S just combines user basic fixed preference and sequential behavior.

Table 3: Influence of different regularization at Recall@20

Dataset	λ_{uv} \ λ_a	0	1	10	50
Tmall	0.01	0.082	0.124	0.143	0.142
	0.001	0.074	0.120	0.137	0.140
	0.0001	0.071	0.116	0.127	0.132
Gowalla	0.01	0.261	0.334	0.339	0.343
	0.001	0.356	0.434	0.418	0.437
	0.0001	0.362	0.432	0.429	0.439

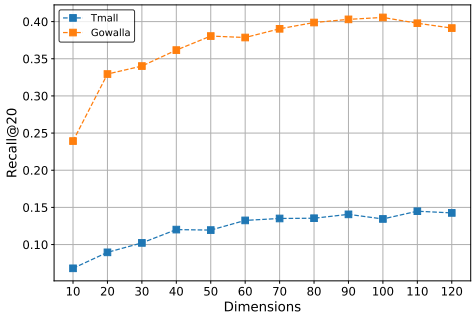


Figure 3: the impact of dimension size

4.4 Influence of Hyper-parameters

In this subsection, we study the influence of regularization and embedding size in our model. Due to space limitation, we just show the results under the metric of Recall@20.

In our model, we utilize user and item embedding regularization λ_{uv} and attention network regularization λ_a to avoid overfitting problem. Table 3 shows the influence of different regularization values at Recall@20. As shown in this table, the performance will be greatly improved when $\lambda_a > 0$. This also indicates that attention network is helpful for our task. We further investigate the impact of dimension size K, which is relevant to not only user and item embedding sizes, but also MLP parameters in attention network. For simplicity, user and item embedding sizes are the same. We can observe that high dimensions can embed better for user and item, and will be more helpful to build high-level factor interaction through attention network. This phenomenon is similar to the traditional latent factor model. In the experiments, we set the size to 100 for the trade-off between computation cost and recommendation quality for both datasets.

5 Conclusion and Future Work

In this paper, we proposed a hierarchical attention network for recommending next item problem. Specifically, we first embedded users and items into low-rank dimension spaces, and then employed a two-layer attention network to model user's dynamic long-term taste and sequential behavior. Our model considered not only dynamic properties in user's long- and short-term preferences, but also high-level complex interactions between user and item factors, item and item factors. From the experiments, we observed that our model outperformed the state-of-the-art methods on two real-world datasets in terms of Recall and AUC.

References

[Bayer *et al.*, 2017] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th International Conference on World Wide Web*, 2017.

[Chen *et al.*, 2012] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *SIGKDD*. ACM, 2012.

[Cheng *et al.*, 2016] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 2016.

[Cho *et al.*, 2011] Eunjoon Cho, Seth A Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*. ACM, 2011.

[Feng *et al.*, 2015] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. Personalized ranking metric embedding for next new poi recommendation. In *IJCAI*, 2015.

[He and Chua, 2017] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *SIGIR*, 2017.

[He and McAuley, 2016] Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In *Proceedings of the 16th International Conference on Data Mining*. IEEE, 2016.

[Hidasi *et al.*, 2015] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *Proceedings of the fourth International Conference on Learning Representations*, 2015.

[Hu *et al.*, 2017] Liang Hu, Longbing Cao, Shoujin Wang, Guandong Xu, Jian Cao, and Zhiping Gu. Diversifying personalized recommendation with user-session context. In *IJCAI*, 2017.

[Liang *et al.*, 2016] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M Blei. Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Proceedings of the 10th conference on recommender systems*. ACM, 2016.

[Liu *et al.*, 2016] Yanchi Liu, Chuanren Liu, Bin Liu, Meng Qu, and Hui Xiong. Unified point-of-interest recommendation with temporal interval assessment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1015–1024. ACM, 2016.

[Pan *et al.*, 2008] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Proceedings of International Conference on Data Mining*. IEEE, 2008.

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

[Rendle *et al.*, 2010] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World Wide Web*. ACM, 2010.

[Wang *et al.*, 2015] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International SIGIR conference on Research and Development in Information Retrieval*. ACM, 2015.

[Wu *et al.*, 2017] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *Proceedings of International Conference on Web Search and Data Mining*. ACM, 2017.

[Xiao *et al.*, 2017] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *IJCAI*, 2017.

[Yang *et al.*, 2016a] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE, 2016.

[Yang *et al.*, 2016b] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, 2016.

[Ying *et al.*, 2016] Haochao Ying, Liang Chen, Yuwen Xiong, and Jian Wu. Collaborative deep ranking: a hybrid pair-wise recommendation algorithm with implicit feedback. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2016.

[Zhang *et al.*, 2016] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *SIGKDD*. ACM, 2016.