

Hierarchical User Profiling for E-commerce Recommender Systems

Yulong Gu

Data Science Lab, JD.com
guyulongcs@gmail.com

Shuaiqiang Wang

Data Science Lab, JD.com
wangshuaiqiang1@jd.com

Zhuoye Ding

Data Science Lab, JD.com
dingzhuoye@jd.com

Dawei Yin

Data Science Lab, JD.com
yindawei@acm.org

ABSTRACT

Hierarchical user profiling that aims to model users' real-time interests in different granularity is an essential issue for personalized recommendations in E-commerce. On one hand, items (i.e. products) are usually organized hierarchically in categories, and correspondingly users' interests are naturally hierarchical on different granularity of items and categories. On the other hand, multiple granularity oriented recommendations become very popular in E-commerce sites, which require hierarchical user profiling in different granularity as well. In this paper, we propose HUP, a **Hierarchical User Profiling** framework to solve the hierarchical user profiling problem in E-commerce recommender systems. In HUP, we provide a Pyramid Recurrent Neural Networks, equipped with Behavior-LSTM to formulate users' hierarchical real-time interests at multiple scales. Furthermore, instead of simply utilizing users' item-level behaviors (e.g., ratings or clicks) in conventional methods, HUP harvests the sequential information of users' temporal finely-granular interactions (micro-behaviors, e.g., clicks on components of items like pictures or comments, browses with navigation of the search engines or recommendations) for modeling. Extensive experiments on two real-world E-commerce datasets demonstrate the significant performance gains of the HUP against state-of-the-art methods for the hierarchical user profiling and recommendation problems. We release the codes and datasets at https://github.com/guyulongcs/WSDM2020_HUP.

CCS CONCEPTS

• **Information systems** → **Personalization; Recommender systems.**

KEYWORDS

User profiling; Recommender systems; Hierarchical user profiling; Pyramid Recurrent Neural Networks; E-commerce

ACM Reference Format:

Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical User Profiling for E-commerce Recommender Systems. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20)*, February 3–7, 2020, Houston, TX, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3336191.3371827>

1 INTRODUCTION

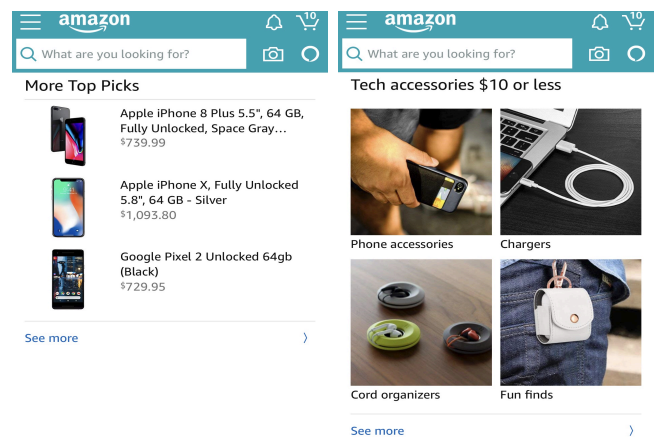


Figure 1: Hierarchical recommendations in Amazon

In the era of Internet, recommender systems are playing crucial roles in various applications such as E-commerce portals (e.g. Amazon, JD.com, Alibaba), social networking websites like Facebook, video-sharing sites like Youtube, visual discovery sites like Pinterest and so on. In practice, *User Profiling* [5, 11, 18, 24, 33, 38] is one of the most important phases in recommender systems. It yields *profile vectors*, which formally represent users' interests by deeply understanding their historical interactions, can be used for candidate generation [31, 42], click-through rate prediction [4, 39, 40], conversion rate prediction [3, 16] and long-term user engagement optimization [34–37, 44–46].

Recently, modeling users' hierarchical real-time interests is emerging to be a crucial issue in E-commerce recommender systems. Firstly, items (i.e. products) in E-commerce sites are typically organized in hierarchical catalogue. Correspondingly, users' interests naturally lie hierarchically on multiple granularity of items and categories. Secondly, different granularity of recommendations (e.g. item, topic and category) become very popular in E-commerce sites, and such scenarios require hierarchical user profiling in different

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WSDM '20, February 3–7, 2020, Houston, TX, USA

© 2020 Association for Computing Machinery.

ACM ISBN 978-1-4503-6822-3/20/02...\$15.00

<https://doi.org/10.1145/3336191.3371827>

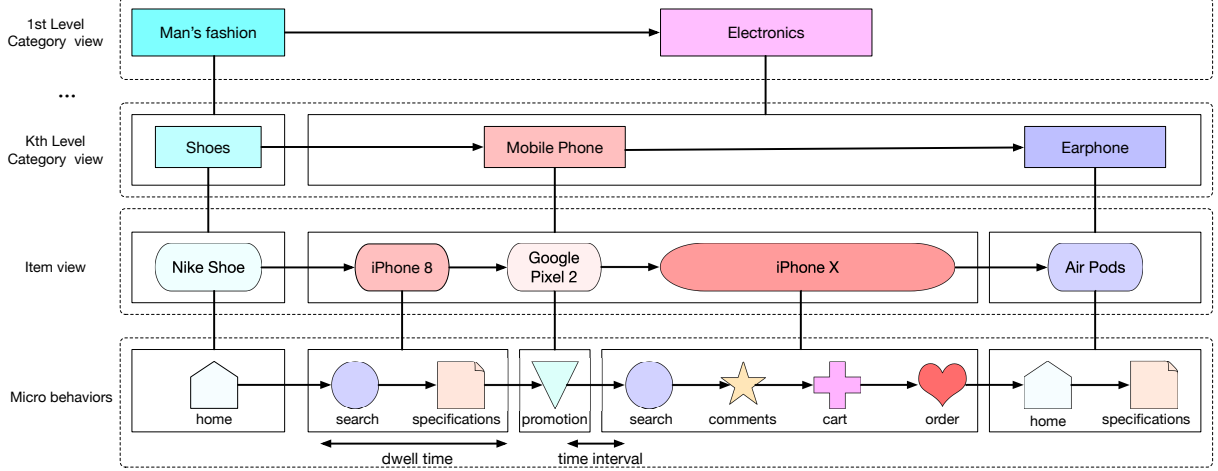


Figure 2: Hierarchical views of Micro behaviors of a user in JD.com

granularity as well. For instance, Figure 1 illustrates a real example of hierarchical recommendations in Amazon. The left side of the figure recommends some items (mobile phones) to a user, while the right side shows a list of recommendations on the categories of “phone accessories”, “chargers” and so on. Category recommendation can help the recommender systems quickly figure out the main interest of the user and make better recommendations.

Existing user profiling methods mainly focus on item recommendations, usually based on users’ item-level responses like ratings [20] or clicks [14]. Among existing methods, latent factor modeling is a popular branch, including matrix factorization [13, 20, 38], neural embedding [8, 10], etc. Generally they learn a unified embedding for the target user to represent her interests on the items based on her historical behaviors. Recently, recurrent neural networks (RNN) have achieved state-of-the-art performance in session-based recommendations [14, 29].

Existing methods have the following limitations. First, when facing different granularity of recommendation tasks, most of them usually need to run a similar algorithm multiple times on different granularity of item organizations, where each run builds users’ certain level profile vectors for the corresponding recommendation task, i.e., item-level profiles for item recommendations and category-level profiles for category recommendations. Correspondingly, the training process of each level’s profile vectors is completely independent from the others. However, users’ multiple-level interests are closely correlated. Figure 2 illustrates a user’s hierarchical interests, including an item level and two category levels, with her historical behaviors. Resulting from the correlations between items and categories, improvement on one recommendation task might benefit others. However, to the best of our knowledge, such privilege has not been explored in existing methods.

Second, only harvesting the signals of users’ item-level interactions like ratings and clicks is insufficient. In most of the E-commerce portals, users provide finely-granular responses such as clicking and browsing different modules (e.g., comments and pictures) of items, adding to shopping carts and purchases, which are referred to as “micro-behaviors” [30, 41]. For example, the bottom layer of Figure 2 presents a user’s historical micro-behaviors

in JD.com (one of the largest e-commerce site in the world), including browsing a pair of Nike shoes from the homepage, searching and reading specifications of iPhone 8, browsing Google Pixels 2 from the promoting page, searching iPhone X, reading comments and adding it into the shopping cart for purchasing, etc. Obviously, in comparison with users’ item-level responses, micro-behaviors provide more detailed information, and preliminary studies [30, 41] have demonstrated the advantage of modeling such detailed behaviors. However, to our best knowledge, none of existing methods has leveraged such advantages to improve the performance of multiple-level user profiling.

Third, generally users’ interests are dynamic and continuously shifting. Some state-of-the-art methods like Time-LSTM [43] usually incorporate time intervals to track the interests shifting. However, we argue that besides the time intervals, the types of behaviors and their dwell time are also extremely important. As shown in Figure 2, we know that iPhone X is preferable to others, since various micro-behaviors are performed on iPhone X with long dwell time. We also observe that triggered by making an order on iPhone X, the user’s interests on mobile phones drop sharply. Neglecting to model behavior types and dwell times, Time-LSTM would be in trouble to capture users’ detailed preferences and interests shifting.

To cope with these challenges, we present HUP, a hierarchical user profiling framework to precisely formulate users’ real-time interests on multiple organizations of items, targeting significant performance gains in recommendation accuracy. In particular, it models users’ multiple-level interests with a Pyramid Recurrent Neural Networks, which typically consist of a micro layer, an item layer, and multiple category recurrent neural network layers. The micro layer harvests the detailed behavioral information and passes it to the higher layers, which could abstract users’ hierarchical interests on the corresponding levels of the item organizations simultaneously. Furthermore, to sensitively track users’ real-time interests, we introduce Behavior-LSTM in each layer, where a *behavior gate* is designed to model the types and dwell time of behaviors. Extensive experiments for item recommendation and category recommendation tasks have been conducted on two large-scale real e-commerce datasets to demonstrate the effectiveness of our proposed approach.

To sum up, our major contributions are listed as follows:

- We formulate a novel hierarchical user profiling problem, which aims to precisely model users’ multiple level interests simultaneously in E-commerce recommender systems.
- We present HUP, which exploits a Pyramid Recurrent Neural Networks for hierarchical user profiling based on users’ historical micro-behaviors.
- We propose Behavior-LSTM, which utilizes a behavior gate to model the types and dwell time of behaviors for effectively formulating users’ real-time interests.
- We conduct extensive experiments and prove that our method outperforms state-of-the-art baselines greatly on both item recommendation and category recommendation tasks.

2 RELATED WORK

2.1 User Profiling for Recommendations

Recommender systems [1] can recommend potentially interested items to users for tackling the information overload problem. Existing works mainly fall into either content-based technology [26] or collaborative filtering [23]. In both of them, user profiling plays a critical role in formulating users’ interests or characteristics [5] based on their behaviors in the past [18, 24, 33, 35, 38]. Classic collaborative filtering techniques like matrix factorization [20] learn users’ static profiles from their rating preferences for estimation of users’ interests in the future [38]. Furthermore, the evolutionary user profiling can learn users’ dynamic profiles along time based on the time changing factor model [19], vector autoregression [24], dynamic sparse topic model [8], etc. However, these methods mainly focus on the item recommendation problem, where neither the sequential information of users’ behaviors nor the hierarchy of the user profiles could be considered.

2.2 RNN-based User Profiling

In recommender systems, recurrent neural networks (RNN) have shown impressive advantages by modeling user’s sequential behaviors [14, 15, 17, 29]. For example, Hidasi et al. [14] introduced the concept of session-based recommendations, and firstly proposed an RNN-based framework to process user’s click sequences on items in a session. Tan et al. [29] further improved its performance by considering the data augmentation and temporal shift issues. Hidasi et al. [15] integrated some content features extracted from images and text into parallel RNN architectures, which demonstrated their significant performance improvements over baselines. Li et al. [22] proposed a neural attentive recommendation machine that can identify users’ main purpose of their current session targeting the performance gains. Beyond behaviors within a session, Quadrana et al. [27] leveraged an additional GRU layer to model users’ cross-session activities for session-based recommendations. Recently, it has been found that the temporal information and users’ finely-granular interactions are significantly helpful for recommendations. Wu et al. [32] leveraged timestamps of behaviors with a long short-term memory (LSTM) autoregressive method. Zhu et al. [43] proposed Time-LSTM, which used the time gates to model the time intervals between behaviors. Wan and McAuley [30] exploited the effectiveness of the relations among users’ different types of behaviors in recommendations. Zhou et al. [41] trained a

single layer RNN model with the micro-behaviors for product recommendation. However, this method only models user’s interests in items and just exploits micro behaviors information as additional input, which might lead to inferior performance. Our method uses multi-layer Behavior-LSTM cells and attentions to explicitly model the micro-behaviors information, which can solve both the item recommendation and the hierarchical categories recommendation problems.

In a word, most existing RNN-based methods fail to address the hierarchical user profiling problem. In addition, to the best of our knowledge, there are no explorations that could leverage the types, dwell time and time intervals of the behaviors simultaneously in an RNN framework for user profiling.

3 PROBLEM FORMULATION

In this section, we firstly introduce the background, notations and definitions in this paper, and then formulate our problem formally.

3.1 Background

Hierarchical categories organize products of the E-commerce sites in different granularity. The hierarchy is generally a tree structure, where each lower level category is an element of a higher level one, and products are usually hung onto the finest categories as the leaf nodes of the tree. For example, the first level category “Electronics” might include some second level categories like “Telephone” and “Accessory”, and “Mobile Phone” is a category in the third and finest level belonging to “Telephone”.

Micro-behaviors are detailed unit interactions (e.g. reading the detail comments, carting) of users with recommender systems. They can provide rich information for indicating users’ timely interests, including the type of behavior that a user conducts on an item, how long a user dwells on an item and move to the next one [30, 41]. In this paper, we consider 10 types of micro behaviors, which are shown in Table 1.

Micro behaviors	Description
Home2Product	Browse the product from the homepage
ShopList2Product	Browse the product from the category page
Sale2Product	Browse the product from the sale page
Cart2Product	Browse the product from the carted page
SearchList2Product	Browse the product from the searched results
Detail_comments	Read the comments of the product
Detail_specification	Read the specification of the product
Detail_bottom	Read the bottom of page of the product
Cart	Add the product to the shopping cart
Order	Make an order

Table 1: List of micro-behaviors

3.2 Hierarchical User Profiling

Definition 3.1 (Hierarchical User Profiling). Hierarchical user profiling aims to generate the micro-level, item-level and hierarchical category-level profile vectors p_{m_u}, p_{i_u} and $p_{c_u} = \{p_{c_u}^{(1)}, \dots, p_{c_u}^{(K)}\}$ respectively based on her micro-behaviors, which represent each target user u ’s interests in corresponding granularity.

Definition 3.2 (Hierarchical Recommendations). Let U be a set of users, V be a set of items, and $C^{(1)}, C^{(2)}, \dots, C^{(K)}$ be the K levels hierarchy of the categories. The hierarchical recommendations task aims to recommend a set of items \hat{V}_u and K set of categories $\hat{C}_u^{(1)}, \dots, \hat{C}_u^{(K)}$ to each target user u by maximizing the relevance between u and her recommendations in different granularity.

4 HUP: A HIERARCHICAL USER PROFILING FRAMEWORK

In this section, we introduce HUP, a hierarchical user profiling framework for hierarchical recommendations. As illustrated in Figure 3, HUP utilizes a Pyramid Recurrent Neural Networks to extract users' hierarchical interests from micro-behaviors at multiple scales.

4.1 The Input and Embedding Layers

Given a target user u , the input of our model is a sequence of her micro-behaviors $X = \langle x_1, x_2, \dots, x_N \rangle$. The i th element $x_i = (t_i, v_i, c_i, b_i, d_i, g_i)$ indicates that u performs a micro-behavior of type b_i on the item v_i at the time t_i , where v_i belongs to multiple-level categories $c_i = \{c_i^{(1)}, c_i^{(2)}, \dots, c_i^{(K)}\}$, the dwell time is d_i , and the time interval between x_i and x_{i+1} is g_i . Here both dwell time and time interval are real numbers. As previous work did [41], we discretize them into several buckets respectively for embedding. For each micro-behavior x_i , the embedding layer firstly uses embedding tables of items, categories, behavior types, dwell time buckets and time intervals to transform v_i, c_i, b_i, d_i, g_i into low-dimensional dense vectors (i.e., $e_{v_i}, e_{c_i}, e_{b_i}, e_{d_i}, e_{g_i}$) respectively and then concatenates these vectors into a single embedding vector e_i . The embedding tables are initialized as random numbers.

4.2 Pyramid Recurrent Neural Networks

Most of previously recurrent neural networks (RNN)-based recommendation methods [5, 14, 15, 29, 41] use a single-layer RNN to generate user profile vectors, which might not be capable of capturing user's hierarchical interests in different levels. To solve this problem, inspired by the Spatial Pyramid Pooling-net (SPP-net) [12], we propose a Pyramid Recurrent Neural Networks, which contains a micro-level, an item-level and several category-level RNN layers to abstract users' hierarchical interests at multiple scales simultaneously.

The micro-level RNN layer aims to model users' finest level interests. The input at the time stamp i of this layer x_{M_i} comes from the embedding layer, and the output of this layer Y_M is forwarded to the item-level RNN layer for further calculations. The hidden state is updated after taking each micro-behavior as input. The formulations of the Micro-level RNN layer are defined in Equation 1.

$$\begin{aligned} X_M &= [x_{M_i}] = [e_i], & i &= 1, 2, \dots, N \\ Y_M &= [y_{M_i}] = \text{RNN}_M(X_M), & i &= 1, 2, \dots, N \end{aligned} \quad (1)$$

The item-level RNN layer models users' item-level interests. The input at the time stamp i of this layer x_{I_i} is the concatenation of the item embedding e_{v_i} and the output of the micro-level layer. The hidden state is only updated after a user have transferred her focuses from one item to another. Its output Y_I is forwarded to the

category-level RNN layers. The formulations of the Item-level RNN layer are defined in Equation 2.

$$\begin{aligned} X_I &= [x_{I_i}] = [e_{v_i}; y_{M_i}] \\ Y_I &= [y_{I_i}] = \text{RNN}_I(X_I) \end{aligned} \quad (2)$$

The category-level RNN layers formulate users' category-level interests. In the K th category layer (the finest granularity of categories), the input at the time stamp i is $x_{C_i}^{(K)}$, which is the concatenation of the category embedding $e_{c_i}^{(K)}$ and the output of the item-level RNN layer calculated on items under this category. For other higher-level category layers, the input at the time stamp i of the k th level category layer is $x_{C_i}^{(k)}$, which is the concatenation of the category embedding $e_{c_i}^{(k)}$ in this layer and the output of the $(k-1)$ th level category layer. In each layer, the hidden state is only updated after a user has moved her focuses from one category to another in this layer. The formulations of the category-level RNN layers are defined in Equation 3.

$$\begin{aligned} X_C^{(k)} &= [x_{C_i}^{(k)}] = \begin{cases} [e_{c_i}^{(k)}; y_{I_i}], & k = K \\ [e_{c_i}^{(k)}; y_{C_i}^{(k-1)}], & k = 1, \dots, K-1 \end{cases} \\ Y_C^{(k)} &= [y_{C_i}^{(k)}] = \text{RNN}_C^{(k)}(X_C^{(k)}), \quad k = 1, \dots, K \end{aligned} \quad (3)$$

4.3 Behavior-LSTM Cell

Generally users' interests are dynamic and continuously shifting. Time-LSTM [43] is a state-of-the-art method that incorporates time intervals between users' sequential purchases to address the interest shifting problem. However, it cannot model the behavior type and the dwell time information, which may lead to inferior performance. We here propose Behavior-LSTM, a novel RNN layer that provides an additional behavior gate to process the types and dwell time of the behaviors, enabling HUP to track users' real-time interests more precisely. In particular, it is described in Figure 4 and formulated in Equation 4:

$$\begin{aligned} I_t &= \sigma(W_I[\mathbf{h}_{t-1}, x_t] + b_I) & \mathcal{F}_t &= \sigma(W_{\mathcal{F}}[\mathbf{h}_{t-1}, x_t] + b_{\mathcal{F}}) \\ \mathcal{T}_t &= \sigma(W_{\mathcal{T}}[x_t, \Delta_t] + b_{\mathcal{T}}) & \mathcal{A}_t &= \sigma(W_{\mathcal{A}}[x_t, a_t] + b_{\mathcal{A}}) \\ \tilde{C}_t &= \tanh(W_C[\mathbf{h}_{t-1}, x_t] + b_C) & C_t &= \mathcal{F}_t \odot C_{t-1} + I_t \odot \mathcal{T}_t \odot \mathcal{A}_t \odot \tilde{C}_t \\ O_t &= \sigma(W_O[\mathbf{h}_{t-1}, x_t] + b_O) & \mathbf{h}_t &= O_t \odot \tanh(C_t) \end{aligned} \quad (4)$$

where $I, \mathcal{F}, \mathcal{T}, \mathcal{A}$ and O are the input, forget, time, behavior and output gates, C and \mathbf{h} are the cell state and hidden state vectors, $W_I, W_{\mathcal{F}}, W_{\mathcal{T}}, W_{\mathcal{A}}, W_C$ and W_O are weight matrices, $b_I, b_{\mathcal{F}}, b_{\mathcal{T}}, b_{\mathcal{A}}, b_C$ and b_O are the biases, respectively. The input of the Behavior-LSTM is a tuple (x_t, a_t, Δ_t) , where x_t is the embedding vector of the input at the time stamp t , a_t is the embedding vector of the behavior type or dwell time information, and Δ_t is the embedding vector of time interval between current behavior and the next one.

In Behavior-LSTM, the time gate \mathcal{T} estimates how much information that should maintain or pass to the next state, and the behavior gate \mathcal{A} calculates the importance of current behavior with the meta information of the behavior. In particular, such meta information of the behaviors involves two aspects: their types and users' dwell time. In particular, the behavior gate actually only processes the types of micro-behaviors in the micro level RNN layer. It is because

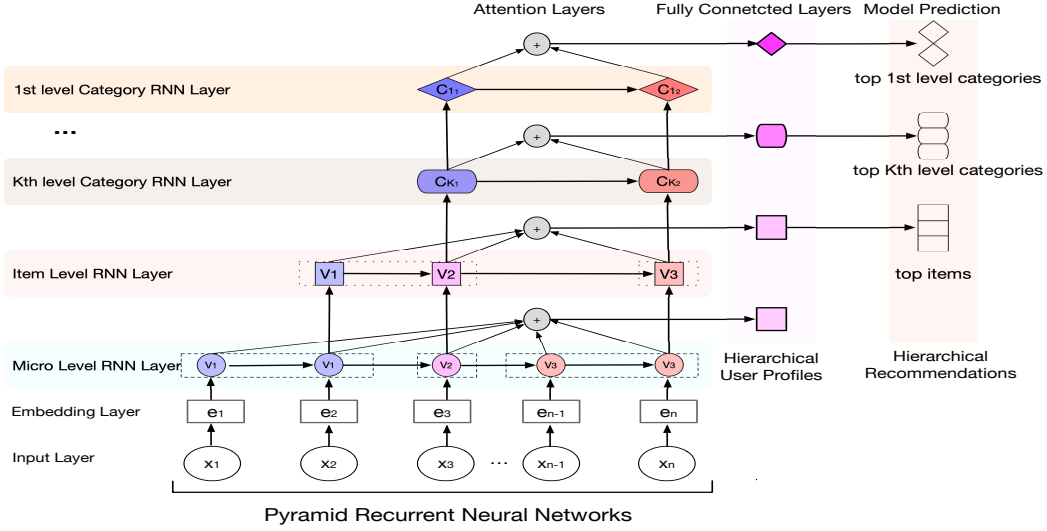


Figure 3: The architecture of the HUP. It uses a Pyramid Recurrent Neural Networks, which is consisted of a micro layer, an item layer, and hierarchical category recurrent neural networks layers, to extract users' hierarchical profile at multiple scales. The profiles represent users' real-time interests in items and hierarchical categories, based on which the most relevant categories and items can be recommended to users.

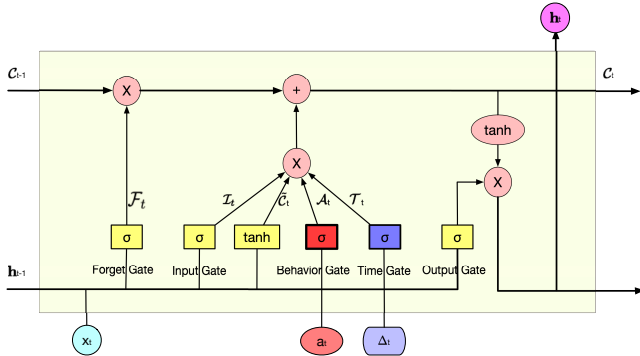


Figure 4: The architecture of the Behavior-LSTM. It has a behavior gate \mathcal{A} and a time gate \mathcal{T} , where \mathcal{A} models users' behavior information in micro behaviors, and \mathcal{T} captures the time intervals between users' micro behaviors.

most of micro-behaviors are instant responses and we could not get their dwell time, but their types are extremely important for users' interest modeling. In the item-level and hierarchical category-level RNN layers, this gate models the dwell time on the items or categories. That is because the dwell time varies significantly in items and categories and is very informative in presenting users' interests.

4.4 The Attention Layers

The attention mechanism [2] is a common technique in deep learning. Usually, it is able to mitigate long-term dependency issues as well as provide interpretations, which is extremely important in real-world recommender systems. In particular, an attention layer

takes the output sequence $Y = [y_1, y_2, \dots, y_T]$ of an RNN as input and return a context vector s . Let y_i be a user's interests at time stamp i . The context vector s of each attention layer is calculated as a weighted sum of the interests vectors among all the time stamps, which is formulated formally in Equation 5.

$$s = \sum_{i=1}^T \alpha_i y_i; \quad \alpha_i = \frac{\exp(e_i)}{\sum_{k=1}^T \exp(e_k)}; \quad e_i = f(y_i, y_T, a_i) \quad (5)$$

HUP has multiple attention layers, where each is directly followed by its corresponding RNN layer and therefore referred to as micro, item and category level attention layers respectively. The context vectors from these attention layers are denoted as s_m , s_i and $s_c = \{s_c^{(1)}, s_c^{(2)}, \dots, s_c^{(K)}\}$ respectively. The attention signal a_i represents the type of micro-behaviors in micro-level attention layer, and the dwell time in both the item and the category level attention layers. f is an alignment model, which scores the importance of y_i based on the hidden state y_i , last hidden state y_T and attention signal a_i . In order to achieve abundant expressive ability, we design the alignment model f as two-layers feedforward neural networks, which is jointly trained in the model.

4.5 The Fully Connected Layers

The fully connected neural network layers transform users' context vectors from the attention layers into hierarchical user profiles. Specifically, they transform users' micro-level, item-level and category-level context vectors s_m , s_i and $s_c = \{s_c^{(1)}, s_c^{(2)}, \dots, s_c^{(K)}\}$ into real-time user profile vectors p_m , p_i and $p_c = \{p_c^{(1)}, p_c^{(2)}, \dots, p_c^{(K)}\}$ in corresponding levels.

4.6 Loss Function

Deep learning models like convolutional neural networks [21] and recurrent neural networks [9] usually use softmax as the last layer for prediction. However, in real-world recommendation scenarios, the possible items can be millions or billions, and thus such calculations on all items is prohibitively expensive. Given a user u and her sequential activities X_u , we try to maximize the cosine similarity between the user's real-time profile vectors (i.e. p_m , p_i and $p_c = \{p_c^{(1)}, \dots, p_c^{(K)}\}$) and the embedding of the ground-truths, on which the target user will act in the next time stamp $N + 1$ (i.e., the next item v_{N+1} for micro and item level layers or the next hierarchical categories $c_{N+1} = \{c_{N+1}^{(1)}, \dots, c_{N+1}^{(K)}\}$ for category-level layers). Similar strategy has achieved success in recommendation systems [14, 29, 41]. Let L_{M_u} , L_{I_u} and $L_{C_u} = \{L_{C_u}^{(1)} \dots L_{C_u}^{(K)}\}$ be the losses of the micro-level, item-level and category-level layers for the target user u . The loss of the micro-level layers L_{M_u} can be calculated as Equation 6, where $e_{v_{N+1}}$ is the embedding of the ground-truth item v_{N+1} . The losses of the item and category levels can be calculated similarly.

$$L_{M_u} = \text{cosine_proximity}(p_m, e_{v_{N+1}}) = -\frac{p_m \cdot e_{v_{N+1}}}{\|p_m\| \|e_{v_{N+1}}\|} \quad (6)$$

The total loss L is the weighted sum of losses in micro-level, item-level and category-level layers of all users. Formally it is defined as follows:

$$L = \lambda_{I_M} \sum_{u \in U} L_{M_u} + \lambda_{I_I} \sum_{u \in U} L_{I_u} + \lambda_{I_C} \sum_{u \in U} \sum_{k=1}^K L_{C_u}^{(k)} \quad (7)$$

where λ_{I_M} , λ_{I_I} and λ_{I_C} are the coefficients of the losses in the micro-level, item-level and multiple category-level layers respectively.

5 EXPERIMENTAL SETTINGS

5.1 Hierarchical Recommendations

We evaluate our proposed HUP method on two tasks: item recommendations and category recommendations. Given a target user u and a sequence of her micro-behaviors, HUP generates a hierarchical profile vectors for u , which represent the user's interests in items and hierarchical categories respectively. At the same time, the embedding vectors of the items and multiple-level categories can be learned from HUP as well during the training stage. The item recommendation process is as follows. At each recommendation stage, as previous work did [41], we first retrieve a set of candidate items, which are similar to at least one of users' browsed items in terms of cosine similarity on embeddings. We then calculate the cosine similarity between each candidate item embedding and the user's item-level profile vector p_i as ranking score. Finally, we rank the candidate items and select top items for recommendations. The category recommendations are performed in a similar manner.

5.2 Dataset

To evaluate the effectiveness of HUP, we utilize the benchmark "JD Micro Behaviors Datasets" [41], which are collected from a large e-commerce site JD.com. The datasets contain users' micro-behaviors in two product categories "Appliances" and "Computers", where each line is a sequence of a user's micro behaviors in a session.

The statistics of the datasets are shown in Table 2. In each dataset, we sort all the sessions in chronological order, and use 70%, 10%, 20% sessions as the training, validation and testing set respectively. As previous work did [41], the last item and the corresponding finest category in each session are used as ground truth.

Dataset	JD-Appliances	JD-Computers
Users	6,166,916	3,191,573
Products	169,856	419,388
Categories	103	93
Number of Micro behaviors	176,483,033	88,766,833

Table 2: Statistics of the Datasets

5.3 Baseline Methods

We make a comparative study of our approach HUP with the following methods, where the last three are state-of-the-art RNN-based methods that have demonstrated excellent performance recently.

- **POP** recommends the most popular items to each user. This simple method is a common mechanism in recommender systems. This simple method has been proven to be comparable to some sophisticated recommender algorithms [6].
- **BPR-MF** implements matrix factorization with the Bayesian personal ranking loss. It is one the most popular methods for recommendations [13, 20, 28].
- **Item-KNN** is a popular item-based recommender algorithm that uses similarities between items for recommendations [7]. In particular, the similarity is calculated with $\text{sim}(i, j) = \frac{\text{Freq}(ij)}{\text{Freq}(i) \times \text{Freq}(j)}$, where $\text{Freq}(i)$ is the number of sequences that an item i shows up [7].
- **Word2vec** makes recommendations based on embedding of the last item in the sequence [10] by Word2vec [25]. It has been proved to be effective in recommendation [10].
- **Word2vec-avg** makes recommendations based on the average embedding of all items in the sequence [41].
- **RIB** [41] is a state-of-the-art method that uses RNN and the attention mechanism to model user's micro-behaviors for recommendation.
- **Time-LSTM** [43] integrates the time interval information between user's item-level behaviors into LSTM.
- **S-HRNN** [27] utilizes a hierarchical GRUs to model users' interactions across sessions.

5.4 Evaluation Metrics

We use two widely used metrics $\text{Recall}@K$ and $\text{MRR}@K$ [14, 27, 29, 41] to compare our model with the baselines. For the item recommendation problem, as previous work did [41], we use $\text{Recall}@20$ and $\text{MRR}@20$ for evaluation. For category recommendation, we use $\text{Recall}@5$ and $\text{MRR}@5$ instead because user's interests in categories are relatively stable. We have implemented our framework HUP with Keras 2.2. The embedding size of items behaviors, categories, dwell time and time intervals are set to 30, 5, 8, 5 and 5 respectively, the batch size is 128 and the hidden size of the PRNN layers is 100.

Model	Appliances				Computers			
	Item Rec		Category Rec		Item Rec		Category Rec	
	<i>Recall@20</i>	<i>MRR@20</i>	<i>Recall@5</i>	<i>MRR@5</i>	<i>Recall@20</i>	<i>MRR@20</i>	<i>Recall@5</i>	<i>MRR@5</i>
POP	3.1	0.5	45.0	24.0	3.4	1.0	44.0	28.6
BPR-MF	13.1	3.1	55.4	35.0	11.3	3.0	70.1	42.9
Item-KNN	42.9	9.6	87.0	43.1	29.8	6.8	68.8	32.7
Word2vec	38.5	8.8	91.1	90.6	28.4	6.2	84.1	81.6
Word2vec-avg	38.7	13.1	86.7	80.0	24.4	7.1	81.0	71.5
RIB	47.6	14.3	92.9	91.2	28.6	7.6	88.0	83.0
Time-LSTM	49.4	18.9	93.4	91.3	32.8	10.9	88.7	83.9
S-HRNN	49.8	19.2	92.6	90.4	33.0	11.0	88.2	82.9
HUP	51.5*	20.5*	93.8*	91.6*	35.0*	12.0*	89.2*	84.4*
HUP-NoMicro	49.6	19.3	93.1	91.1	32.6	10.6	88.2	83.6
HUP-LSTM	50.1	19.6	93.2	91.3	32.7	10.7	88.3	83.6
HUP-TLSTM	50.9	20.0	93.5	91.3	33.8	11.3	88.8	84.0
HUP-NoAtt	50.3	19.7	93.6	91.5	33.8	11.4	89.1	84.2
HUP-Single	50.5	19.7	93.4	91.5	33.9	11.4	88.6	83.8

Table 3: Performance of different methods for category recommendation and item recommendation on two datasets. “*” indicates the statistically significant improvements (i.e., two-sided t-test with $p < 0.01$) over both the best baseline and all variants.

6 EXPERIMENTAL RESULTS

6.1 Comparison with Baselines

Table 3 shows the experimental results of different methods for the item and category recommendation tasks on the Appliances and Computers datasets. We conducted significance testing (t-test) on the improvements of our approaches over all baselines. “*” denotes strong significant divergence with p -value < 0.01 . From the table we can find that:

- HUP significantly outperforms state-of-the-art methods for the two tasks on both datasets. Specifically, for item recommendation, HUP outperforms state-of-the-art method by 3.4%, 6.1% in *Recall@20* and 6.7%, 9.1% in *MRR@20* for the “Appliances” and “Computers” datasets respectively. For category recommendation, our performance gains are relatively subtle, as this problem is easier than item recommendation resulting from the denser dataset.
- The POP and BPR-MF methods perform the worst.
- Three RNN-based methods including RIB, Time-LSTM and S-HRNN significantly overcome conventional baselines.
- By modeling the temporal information, Time-LSTM achieves better performance than RIB.

6.2 Effectiveness of Components in HUP

To systematically validate the effectiveness of each component in HUP, we implement the following variants of HUP, each eliminating a specific model component.

- **HUP-NoMicro.** This variant does not use micro behaviors for modeling. It only uses the item-level and category-level RNN layers based on users’ interactions with items and categories.
- **HUP-LSTM.** This variant uses LSTM in the PRNN layers. The time-related mechanism and type of micro-behaviors are absent in the method.

- **HUP-TLSTM.** This variant uses Time-LSTM [43] in the PRNN layers, where only the time gates are used in the RNN layer to model the time interval among behaviors.
- **HUP-NoAtt.** This variant removes the attention layers from HUP.
- **HUP-Single.** This variant solves each recommendation task independently, which includes a single Behavior-LSTM layer and an attention layer in the framework.

The performance of different variants are shown in the bottom part of Table 3. From the table, we can see that the full version of HUP outperforms all of the variants and find that:

- (1) **Pyramid Recurrent Neural Networks.** Comparing with the HUP-Single method, the performance improvement demonstrate the effectiveness of PRNN.
- (2) **Micro-behaviors.** The comparison with HUP-NoMicro demonstrates the importance of micro-behaviors in HUP. We also notice that HUP-NoMicro obtains the worst performance in all metrics.
- (3) **Temporal mechanisms.** Evidenced by the performance loss of HUP-LSTM against HUP-TLSTM, and HUP, temporal information is necessary in modeling user interests. Furthermore, sophisticated temporal mechanisms could receive improved performance. For example, equipped with time gates, HUP-TLSTM can achieve better performance than HUP-LSTM; HUP outperforms HUP-TLSTM by further using Behavior gates and time-mechanisms in the attention layers.
- (4) **Attention layers.** As demonstrated in our experiments, attention layers can significantly improve the performance of HUP against the HUP-NoAtt variant. Meanwhile, attention mechanisms also help interpret and visualize the recommendation results as well. We will show that in the later case study section.

6.3 Trade-off in Loss Function

As formulated in Equation (7), the loss function is composed of three components. According to our experiments, HUP achieves





































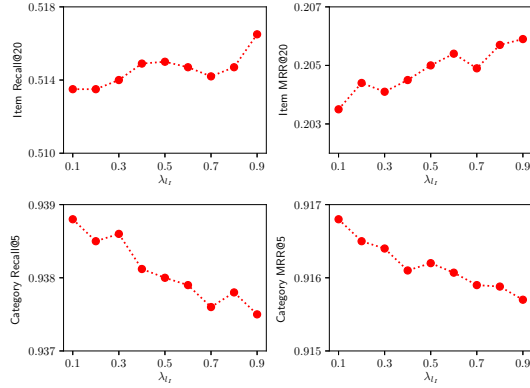
						<div>lowhigh</div> <div>Micro behaviors Attention</div> <div>Item Attention</div> <div>Category Attention</div>					
	Input					Item Recommendation			Category Recommendation		
Category	Item	Micro Behaviors	Dwell Time	Time Interval	RIB	HUP-TLSTM	HUP	RIB	HUP-TLSTM	HUP	
Multi-function Pot		Midea Electric Heating Lunch Box MZ-LYH18-A	ShopList2Product	36s	9s						
		Detail_comments		2s							
		Midea Air fryer MF-TN20B	ShopList2Product	8s	87s						
Electromagnetic Oven		Midea Electromagnetic Oven C21-WK2102	Search2Product	12s	14s						
Electric Kettle		Bear Electric Kettle YSH-B18T1	Search2Product	14s	36s						
Yogurt Maker and Ice Cream Machine Bundle		Bear Yogurt Maker SNJ-5012 and Bear Ice Cream Machine BQL-A12G1 Bundle	ShopList2Product	4s							
		Detail_comments		13s	1s						
		Bear Egg Cooker	Cart	2s							
Egg Cooker			ShopList2Product	3s							
			Cart	2s							
Yogurt Maker and Toaster Bundle		Bear Yogurt Maker SNJ-560 and Bear Toaster DSL-606 discounted Bundle	ShopList2Product	14s							
		Detail_specifications			Not in top 20	Rank 4th	Rank 2nd	Rank 5th	Rank 5th	Rank 3rd	

Figure 5: A case study in an E-commerce Recommender Systems

Figure 6: Performance of HUP with λ_{I_I} which is the weight of the losses of item-level RNN layer.

the best performance when $\lambda_{I_M} = 0$. Because the micro-level RNN layer is useful for predicting the next micro-behaviors based on the historical ones, but does not directly affect the predictions of the items and categories. However, this layer can be used to interpret the real-time effectiveness of the historical micro-behaviors for each user. Without loss of generality, we set $\lambda_{I_C} = 1 - \lambda_{I_I}$ and $0 \leq \lambda_{I_I} \leq 1$. Therefore we only need to tune λ_{I_I} in the loss function for a trade-off between item and category recommendation tasks. Figure 6 shows the performance of HUP with respect to different values of λ_{I_I} on the “Appliances” dataset. The curves on the “Computers” dataset are similar and thus absent from this paper for satisfying the requirement of the page limit. From the figure we can see that the metrics for item recommendation decline when λ_{I_I} is getting lower while the trends are completely opposite for category

recommendation. To balance the performance of the user profiles in different levels, we set λ_{I_I} and λ_{I_C} both to 0.5 in the experiments.

6.4 Case Study

Figure 5 demonstrates a real case from the “Appliances” dataset to explain how HUP works. The last 12 micro-behaviors on 7 items from 6 categories are listed in the figure. The last item is the ground-truth, which spans 2 micro-behaviors. The right side of the figure visualizes the attention weights of these micro-behaviors from our proposed HUP, HUP-TLSTM (a variant of HUP) and RIB (a state-of-the-art baseline). From the figure we can see that:

(1) The attention weights of the micro behaviors “Cart” and “Search2Product” are higher than others for all methods, which means these two micro behaviors are important for modeling user interests.

(2) The time interval between the browsing behaviors on the item “Bear Electric Kettle” and the next one is 36 seconds. As the attention weights shown, HUP-TLSTM and HUP pay much less attentions to the first 4 items than RIB resulting from the time gates. It illustrates their ability of forgetting history behaviors happened long time ago by modeling the time interval information.

(3) Time interval between the browsing behaviors on “Yogurt Maker and Ice Cream Machine” and “Bear Egg Cooker” is merely 2 seconds. This number between “Bear Egg Cooker” and the next item (ground-truth) is also 2 seconds. Both are very short. HUP-TLSTM retains such history information and still pays much attention to these two items resulting from the short time interval. However, HUP can notice the fact that the user has already added these two items to cart. It thus reduces the importance on these two items and their categories, and then chooses an item from a related category (Yogurt Maker and Toaster Bundle) for return.

7 CONCLUSIONS

In this paper, we investigate the hierarchical user profiling problem, aiming to model users' real-time interests in different granularity. It is crucial for multiple-level recommendation tasks, such as item, category, topic, theme recommendations and so on. We hence propose HUP, a hierarchical user profiling framework, which leverages a Pyramid Recurrent Neural Networks to abstract users' interests in different granularity simultaneously from users' micro-behaviors. To better model users' real-time interests, we design Behavior-LSTM cells to integrate the meta information of behaviors (e.g. the type, dwell time and time interval information) into HUP. Extensive experiments on two real-world E-commerce datasets verify the effectiveness of our method for both item and category recommendation tasks.

Resulting from its effectiveness and flexibility, our framework can be widely used to recommend items (e.g. movies, music, news) and corresponding categories (e.g. science fiction films, rock music, breaking news) in various web services (e.g. videos or music sharing sites, social networks).

REFERENCES

- [1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering* 6 (2005), 734–749.
- [2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv* (2014). arXiv:1409.0473
- [3] Ting Bai, Lixin Zou, Wayne Xin Zhao, Pan Du, Weidong Liu, Jian-Yun Nie, and Ji-Rong Wen. 2019. CTRec: A Long-Short Demands Evolution Model for Continuous-Time Recommendation. In *SIGIR*. 675–684.
- [4] Shiyu Chang, Yang Zhang, Jiliang Tang, Dawei Yin, Yi Chang, Mark A Hasegawa-Johnson, and Thomas S Huang. 2017. Streaming recommender systems. In *WWW*. 381–389.
- [5] Weijian Chen, Yulong Gu, Zhaochun Ren, Xiangnan He, Hongtao Xie, Tong Guo, Dawei Yin, and Yongdong Zhang. 2019. Semi-supervised user profiling with heterogeneous graph attention networks. In *IJCAI*. 2116–2122.
- [6] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of Recommender Algorithms on Top-N Recommendation Tasks. In *RecSys*. 39–46.
- [7] Mukund Deshpande and George Karypis. 2004. Item-based Top-N Recommendation Algorithms. *ACM Transactions on Information Systems* 22, 1 (2004), 143–177.
- [8] Li Gao, Jia Wu, Chuan Zhou, and Yue Hu. 2017. Collaborative Dynamic Sparse Topic Regression with User Profile Evolution for Item Recommendation. In *AAAI*. 1316–1322.
- [9] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. In *ICASSP*. 6645–6649.
- [10] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikrit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in Your Inbox: Product Recommendations at Scale. In *KDD*. 1809–1818.
- [11] Yulong Gu, Jiaying Song, Weidong Liu, and Lixin Zou. 2016. HLGPS: A Home Location Global Positioning System in Location-Based Social Networks. In *ICDM*. 901–906.
- [12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence* 37, 9 (2015), 1904–1916.
- [13] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *SIGIR*. 549–558.
- [14] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *arXiv* (2015). arXiv:1511.06939
- [15] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *RecSys*. 241–248.
- [16] Chao Huang, Xian Wu, Xuchao Zhang, Chuxu Zhang, Jiashu Zhao, Dawei Yin, and Nitesh V Chawla. 2019. Online Purchase Prediction via Multi-Scale Modeling of Behavior Dynamics. In *KDD*. 2613–2622.
- [17] Jin Huang, Zhaochun Ren, Wayne Xin Zhao, Gaole He, Ji-Rong Wen, and Daxiang Dong. 2019. Taxonomy-aware multi-hop reasoning networks for sequential recommendation. In *WSDM*. 573–581.
- [18] Gabriella Kazai, Iskander Yusof, and Daoud Clarke. 2016. Personalised News and Blog Recommendations based on User Location, Facebook and Twitter User Profiling. In *SIGIR*. 1129–1132.
- [19] Yehuda Koren. 2009. Collaborative Filtering with Temporal Dynamics. In *KDD*. 447–456.
- [20] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.
- [21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet Classification with Deep Convolutional Neural Networks. In *NIPS*. 1097–1105.
- [22] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM*. 1419–1428.
- [23] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.
- [24] Zhongqi Lu, Sinno Jialin Pan, Yong Li, Jie Jiang, and Qiang Yang. 2016. Collaborative Evolution for User Profiling in Recommender Systems. In *IJCAI*. 3804–3810.
- [25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *NIPS*. 3111–3119.
- [26] Michael J. Pazzani and Daniel Billsus. 2007. The Adaptive Web. Springer-Verlag, Berlin, Heidelberg, Chapter Content-based Recommendation Systems, 325–341.
- [27] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In *RecSys*. 130–137.
- [28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.
- [29] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *RecSys*. 17–22.
- [30] Mengting Wan and Julian McAuley. 2018. Item Recommendation on Monotonic Behavior Chains. In *RecSys*. ACM, 86–94.
- [31] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. 2018. A path-constrained framework for discriminating substitutable and complementary products in e-commerce. In *WSDM*. 619–627.
- [32] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent Recommender Networks. In *WSDM*. 495–503.
- [33] Mohammad Yahya H. and Al-Shamri. 2016. User Profiling Approaches for Demographic Recommender Systems. *Knowledge-Based Systems* 100 (2016), 175–187.
- [34] Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin. 2019. Deep reinforcement learning for search, recommendation, and online advertising: a survey by Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin with Martin Vesely as coordinator. *ACM SIGWEB Newsletter* Spring (2019), 4.
- [35] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for page-wise recommendations. In *RecSys*. 95–103.
- [36] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for page-wise recommendations. In *RecSys*. 95–103.
- [37] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *KDD*. 1040–1048.
- [38] Zhe Zhao, Zhiyuan Cheng, Lichan Hong, and Ed H. Chi. 2015. Improving User Topic Interest Profiles by Behavior Factorization. In *WWW*. 1406–1416.
- [39] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *AAAI*. 5941–5948.
- [40] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *KDD*. 1059–1068.
- [41] Mei-zi Zhou, Zhuoye Ding, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *WSDM*. 727–735.
- [42] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *KDD*. 1079–1088.
- [43] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do Next: Modeling User Behaviors by Time-LSTM. In *IJCAI*. 3602–3608.
- [44] Lixin Zou, Long Xia, Zhuoye Ding, Jiaying Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement Learning to Optimize Long-term User Engagement in Recommender Systems. In *KDD*. 2810–2818.
- [45] Lixin Zou, Long Xia, Zhuoye Ding, Dawei Yin, Jiaying Song, and Weidong Liu. 2019. Reinforcement Learning to Diversify Top-N Recommendation. In *DASFAA*. 104–120.
- [46] Lixin Zou, Long Xia, Pan Du, Zhuo Zhang, Ting Bai, Weidong Liu, Jian-Yun Nie, and Dawei Yin. 2020. Pseudo Dyna-Q: A Reinforcement Learning Framework for Interactive Recommendation. In *WSDM*.