Micro Behaviors: A New Perspective in E-commerce Recommender Systems

Meizi Zhou*
University of Minnesota
zhou0793@umn.edu

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

Jiliang Tang
Michigan State University
tangjili@msu.edu

Dawei Yin†
Data Science Lab, JD.com
yindawei@acm.org

ABSTRACT

The explosive popularity of e-commerce sites has reshaped users' shopping habits and an increasing number of users prefer to spend more time shopping online. This evolution allows e-commerce sites to observe rich data about users. The majority of traditional recommender systems have focused on the macro interactions between users and items, i.e., the purchase history of a customer. However, within each macro interaction between a user and an item, the user actually performs a sequence of micro behaviors, which indicate how the user locates the item, what activities the user conducts on the item (e.g., reading the comments, carting, and ordering) and how long the user stays with the item. Such micro behaviors offer fine-grained and deep understandings about users and provide tremendous opportunities to advance recommender systems in e-commerce. However, exploiting micro behaviors for recommendations is rather limited, which motivates us to investigate e-commerce recommendations from a micro-behavior perspective in this paper. Particularly, we uncover the effects of micro behaviors on recommendations and propose an interpretable **R**ecommendation framework RIB, which models inherently the sequence of m**I**cro **B**ehaviors and their effects. Experimental results on datasets from a real e-commence site demonstrate the effectiveness of the proposed framework and the importance of micro behaviors for recommendations.

CCS CONCEPTS

• **Information systems** → **Summarization**;

KEYWORDS

Micro behaviors, RNN, attention mechanism, e-commerce, recommendation

*This work was done, when the author was an internship at Data Science Lab of JD.com.
†Corresponding author

WSDM 2018, February 5–9, 2018, Marina Del Rey, CA, USA

ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159671

ACM Reference Format:
Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *Proceedings of WSDM 2018: The Eleventh ACM International Conference on Web Search and Data Mining (WSDM 2018).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3159652.3159671

1 INTRODUCTION

The modern e-commerce sites such as Amazon [1] and eBay [2] offer hundreds of millions of products for sale. For example, as on June 20th, 2017, Amazon has more than 372 million products [3]. It has become increasingly challenging for consumers to find their interested items. Recommender systems play a crucial role in mitigating this information overload problem by suggesting products that have potentials to fit consumers' needs. They have been proven to not only help increase customer satisfaction and create customer loyalty [34] but also boost many aspects of e-commerce services such as revenue and growth [24].

Meanwhile, the popularity of e-commerce sites has reshaped users' shopping habits and users prefer to spend more time shopping online. For example, on average, American parents spend 7 hours per week on e-commerce sites [4]. This evolution enables e-commerce sites to observe rich data about their users. Figure 1 illustrates a real example of observed data on a user from an e-commerce site in a short period. The user first enters a page of iPhone 7 from searching result page. She reads the detailed description, as well as others' comments and adds it to the cart. Then she shifts to a page of iPhone 6 from the searching result page and reads the comments. After that, she browses a page of iPhone 7 cases from the sale page and orders the case. Finally, she jumps to a page of Samsung Galaxy from the home page of the e-commerce site. From a macro perspective as shown in the top subfigure in Figure 1, the user interacted with iPhone 7, iPhone 6, iPhone 7 cases and Samsung Galaxy. While from a micro perspective as shown in the bottom subfigure, each macro interaction includes a sequence of behaviors that can indicate how the user located the product page (e.g., the search engine or the sale promotion), whether the user clicks detailed information about a product (e.g., comments, or specifications), whether a user carts or orders a product, and how long the user dwells on a product. In this work, we refer these behaviors

[1] https://www.amazon.com/
[2] https://www.ebay.com/
[3] https://www.scrapehero.com/number-of-products-sold-on-amazon-com-june-2017/
[4] https://www.bigcommerce.com/blog/ecommerce-trends/#cmtoc_anchor_id_0

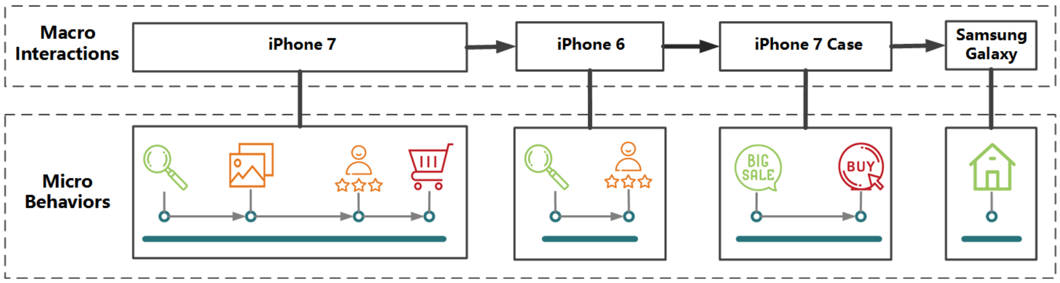

Figure 1: An illustrative example of observed data on a user from a real e-commerce site.

in macro interactions as micro-behaviors. These micro behaviors can provide fine-grained understandings about users. For example, locating a product from searching could indicate stronger intents than from the e-commerce homepage; and longer dwell time on a product suggests more interests in the product than shorter dwell time. Hence, exploiting micro behaviors has immense potential to advance recommender systems. However, such research is rather limited in the literature.

In this paper, we investigate e-commerce recommendations from the micro-behavior perspective where (a) data of a user is inherently viewed as a sequence of macro interactions between users and items; and (b) each macro interaction between a user and an item includes a sequence of micro behaviors as shown in Figure 1. One major advantage of the new perspective is – it provides a unified setting that makes various settings of existing recommender systems become its special cases:

- if we completely ignore the sequential information of macro interactions and their micro-behaviors, macro interactions can be denoted as an user-item matrix, which is the typically setting for traditional collaborative filtering [29];
- when we completely ignore the sequential information of macro interactions but consider certain micro behaviors in macro interactions, our setting is boiled down to traditional collaborative filtering with implicit feedback from certain micro behaviors such clicks [12] and dwell time [35]; and
- when we only consider the sequential information of macro interactions in a session and ignore micro-behaviors within macro interactions, the studied problem become session-based recommendations [14, 15, 31].

On the other hand, it also poses tremendous challenges including (1) how to model sequential information and (2) how to capture effects from a variety of micro behaviors. Solutions to these two challenges lead to a novel recommendation framework. Our major contributions are summarizas follows:

- We uncover the effects of micro behaviors on e-commerce recommendations;
- We provide a principled approach to capture the sequence of various micro behaviors mathematically;
- We propose an interpretable **R**ecommendation framework from the m**I**cro **B**ehavior perspective **RIB**, which incorporates the sequence of micro behaviors and their corresponding effects into a coherent model; and
- We demonstrate the effectiveness of the proposed framework and the importance of micro behaviors on data from a real e-commerce site.

The rest of this paper is organized as follows. In Section 2, we formally define the problem. We perform preliminary data analysis on micro behaviors in Section 3. In Section 4, we detail the proposed framework RIB. Experimental results with discussions are presented in Section 5. In Section 6, we briefly review related work. Finally we conclude our work and discuss the future work in Section 7.

2 PROBLEM STATEMENT

Let $P = \{p_1, p_2, ..., p_N\}$ be the set of products where N is the number of products. $A = \{a_1, a_2, ..., a_M\}$ be the set of activities a user can perform where M is the number of activities. In addition, we also consider dwell time, which indicates how long a user performs activities in A. We further assume that there are K different dwell time choices as $D = \{d_1, d_2, ..., d_K\}$. Note that dwell time is typically continuous and in this work, we discretize it into K segments. Recall that in our studied problem, (1) data of a user is a sequence of interactions between the user and items and (2) each interaction includes a sequence of micro-behaviors. With the definitions of P, A, D, data in our study can be represented as a sequence of tuples (p_i, a_j, d_k) (or micro behaviors) where $p_i \in P$, $a_j \in A$, and $d_k \in D$. The tuple (p_i, a_j, d_k) denotes that the user performs the activity a_j on the product p_i with d_k dwell time.

With the notations and definitions, the problem we want to study in this work is formally stated as: *Given their historical data of a set of users, i.e., sequences of tuples (or micro behaviors) (p_i, a_j, d_k), we aim to build a recommender system that can suggest the next product for each user which she is interested in.*

3 UNCOVERING EFFECTS OF MICRO-BEHAVIORS

In this section, we begin by introducing the dataset we use for this analysis and then investigate the effects of micro-behaviors on e-commerce.

3.1 Data

We collect a dataset from a real e-commerce site in a given period. It contains four types of information as:

- **Click Source.** The entrence to a product page (or click source) can indicate why the user is interested in this product. For example, if a customer enters a product page from

Table 1: Definitions specific to the dataset

Var	Attribute	Description
p_i	Product ID	SKU (Stock Keeping Unit)
c_i	Category ID	Product category
a_1	Home2Product	Enter p_i from homepage
a_2	ShopList2Product	Enter p_i from category page
a_3	Sale2Product	Enter p_i from sale page
a_4	Cart2Product	Enter p_i from carted page
a_5	SearchList2Product	Enter p_i from searched results
a_6	Detail_comments	In p_i's comment module
a_7	Detail_specification	In p_i's specification module
a_8	Detail_bottom	At the bottom
a_9	Cart	Add p_i to cart
a_{10}	Order	Order p_i
t_1	Dwell time	0∼9 seconds
t_2	Dwell time	10∼24 seconds
t_3	Dwell time	25∼60 seconds
t_4	Dwell time	61∼120 seconds
t_5	Dwell time	>120 seconds

the home page, she may just want to look around and has no target to buy; while if a customer searches for certain products and enters one from the searching result page, he may have a strong intent to buy it. Click source includes the home page, shopping cart page, sale page, searching result page and so on.

- **Browsing Modules.** In a product page, there are several common modules, such as the brief information (name, price, thumbnail picture), comments, and specification. Browsing Modules can help us understand users' interests. For instance, if a user browses the comments and specifications instead of only browsing the brief information, he would have a higher probability of buying this product.
- **Cart and Order.** Adding to cart and ordering actions offer strong signals for recommendations. Adding to cart is usually a strong sign of buying a product. However, depending on the characteristics of products, ordering may mean interest shifting or high potential for re-purchase. For example, if a customer purchases some snacks, she may repurchase it in the near future; however, if a customer buys a TV, she is less likely to buy a TV very soon.
- **Dwell time.** Dwell time is commonly adopted to measure the importance of a webpage on a search engine [1, 18, 35]. It denotes the length of time that a visitor spends on a page before jumping to another page. Typically, the longer the dwell time, the more interesting the page. The dwell time also exits in e-commerce sites where it indicates the time a customer spends on one product.

In the dataset, each product is identified by its SKU (Stock Keeping Unit) as what is done in industry. We discretize continuous dwell time into 5 segments and each segment has a similar number of micro behaviors. The definitions of P, A and D specific to the dataset are demonstrated in Table 1.

Eventually this dataset consists of more than 2, 000, 000 users and 150, 000 items. The total number of micro behaviors is more than 90, 000, 000.

3.2 Effects of Micro Behaviors

In this subsection, we investigate the correlations among micro-behaviors in the sequence. In particular, each time, we choose one micro behavior and then check how others relate to it.

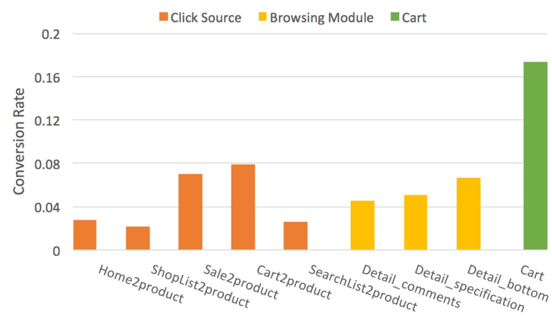

Figure 2: Ordering vs. Other micro behaviors

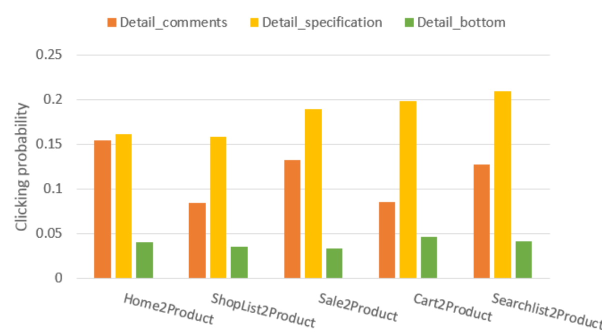

Figure 3: Click Source vs. Browsing Modules

Ordering: to study how ordering relates to other micro behaviors, we investigate the relation between certain micro behaviors and the conversion rate. The conversion rate means the percent of a given behavior that ends with ordering in a time window (or a session). The conversion rate of a micro behavior a_i can be formulated as:

$$Conversion\ rate = \frac{\#\text{ behaviors of } a_i \text{ ended with ordering}}{\#\text{ behaviors of } a_i} \quad (1)$$

The relations between ordering and other micro behaviors are demonstrated in Figure 2 and Figure 4. In Figure 2, the micro behavior "Cart" has the highest conversion rate, which means if a

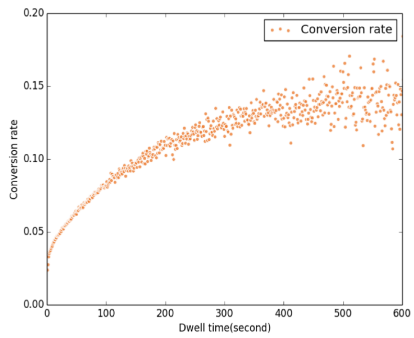

Figure 4: Ordering vs. Dwell time.

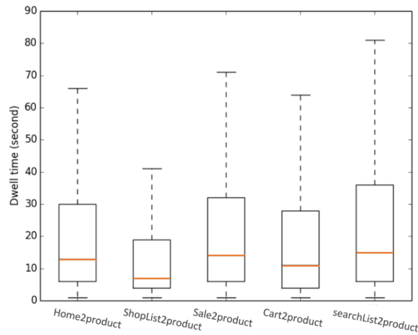

Figure 5: Dwell Time vs. Click Source

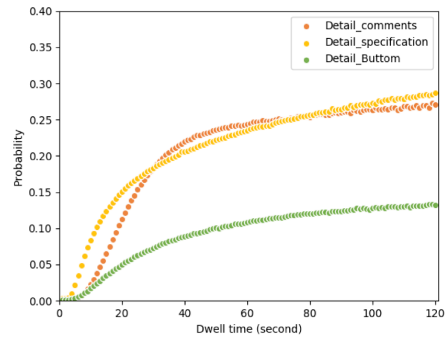

Figure 6: Dwell Time vs. Browsing Modules

user adds a product to cart, he is more likely to order it in the end. Similarly, if a user enters a product page from the cart list, he is also very likely to order it. Besides, if the user reads comments, specifications or finishes all the contents of a page, he is also very likely to order it. But if the user just enters a product page without other behaviors, he is less likely to buy the product in the end. In Figure 4, we note that within a certain range, the longer the dwell time is, the higher the conversion rate is. When the dwell time is out of a certain range, the conversion rate would drop down. If the user stays much longer than he needs to finish the page, he might have transferred his attention offline.

Dwell Time: Figure 4 has demonstrated the impact of dwell time on ordering. We further show how dwell time is related to micro-behaviors of Click Source and Browsing Modules in Figure 5 and Figure 6, respectively. In Figure 5, dwell time on a product is related to how a user locates the product. For example, from searching pages, she may spend longer time on this product; while if from the shop list, she is more likely to spend less time on this product. We also observe from Figure 6 that the longer the dwell time, the more likely a user would visit the detailed modules including reading comments and specifications.

Click Source: We have demonstrated that micro behaviors of Click Source are related to ordering and dwell time. We further show the relations between Click Source and Browsing Modules in Figure 3. We notice that the likelihood of clicking detailed modules is related to the source how a user locates the project. For example, if a customer locates a product from searching, she has a high probability to read the comments and specifications of the product.

To sum up, we make two key observations about micro behaviors via the above analysis – (1) micro behaviors are correlated; and (2) the effects of one micro-behavior on others are varied. These two observations suggest the complexity and challenges of modeling micro behaviors, however, they also offer important insights to build a meaningful model for micro behaviors in the following section.

4 E-COMMERCE RECOMMENDER SYSTEMS FROM A MICRO BEHAVIORS PERSPECTIVE

To model micro-behaviors, there are three major challenges. First, a user is a sequence of tuples (p_i, a_j, d_k) and there are in total $N \times M \times K$ tuples; thus the user representation (or input data) is very sparse and high-dimensional. Second, micro behaviors in the sequence are correlated, then how to model sequential information of micro behaviors. Third, different micro behaviors have distinct importance; hence, how to capture varied effects of micro behaviors. In this work, we propose a framework, which can tackle these three challenges simultaneously. The architecture of our framework is shown in Figure 7. It consists of five layers – an input layer, an embedding layer to solve the sparse and high-dimensional challenge, a RNN layer to model sequential information, an attention layer to capture varied effects of micro behaviors and an output layer. In the following subsections, we will detail each layer.

4.1 The Input and Embedding Layers

The input of the model is the data of a user u with a sequence of n micro behaviors. We formally define it as a sequence $S_u = \{x_1, x_2, ..., x_n\}$, where each x_i is a tuple.

$$x_t = (p_v, a_m, d_k) \tag{2}$$

$p_v \in \mathbb{R}^V$ is an one-hot indicator vector where $p_v(i) = 1$ if x_i is about the i-th product and other entities are zero. Similarly $a_m \in \mathbb{R}^M$ and $d_k \in \mathbb{R}^K$ are indicator vectors for activities and dwell time, respectively. Each of them indicates a unique element in the product set P, activity set A, dwell time set D respectively.

The vocabulary sizes of P, A, D are V, M, K respectively, and there are $V \times M \times K$ tuples in total. Therefore, the input data is extremely sparse and high-dimensional. We design an embedding layer to transform the input x_t into a low-dimensional dense vector e_t, which is formally defined as:

$$e_t = concatenate(W_P p_v, W_A a_m, W_D d_k) \tag{3}$$

where $W_P \in R^{d_P \times V}$, $W_A \in R^{d_A \times M}$, $W_D \in R^{d_D \times K}$ where $d_P \ll N$, $d_A \ll M$ and $d_D \ll K$ are the numbers of latent dimensions for products, activities and dwell time, separately. The initial weights of W_P, W_A, W_D are trained via Word2vec [25]. And the final embedding of x_t is the concatenation of three embeddings. The new representation of x_t, e_t is dense with dimension of $d_P + d_A + d_D$, which is much smaller than $V \times M \times K$.

4.2 The RNN Layer

Recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This

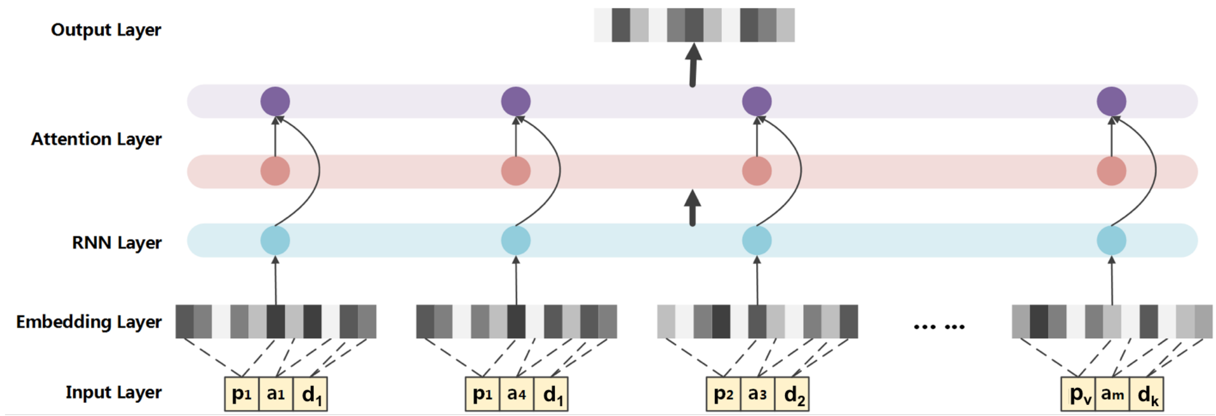

Figure 7: The architecture of the proposed framework.

creates an internal state of the network which allows it to exhibit dynamic temporal behavior. Therefore, we build a RNN layer to capture the sequential information of micro behaviors. The output of the embedding layer e_t is the input of the RNN layer. The t^{th} hidden state unit output is calculated by

$$h_t = \sigma(W_{eh}e_t + W_{hh}h_{t-1} + b_t) \tag{4}$$

where $\sigma(\cdot)$ is a nonlinear activation function, e.g. ReLU, sigmoid or tanh; $W_{eh} \in R^{d_h \times d_e}$, $W_{hh} \in R^{d_h \times d_h}$, and $b_i \in R^{d_h}$. Thus, the output on time t is not only decided by the t^{th} input but also by the output of previous time h_{t-1}. Recurrently, the historical inputs in a sequence are taken into consideration when making new decisions. Micro behaviors in the sequence may exist long dependencies. For example, a customer purchased water one week ago and now she may be interested in repurchasing the same product again. However, when the input sequence is long, RNN may have the problem of gradient vanishing. In other words, when the current micro behavior is influenced both by the recent and previous ones, RNN may not work well. There are two possible solutions. One is the long short-term memory model (LSTM) [16]. The LSTM updates its hidden unit by introducing the input gate i_t, forget gate f_t and output gate o_t as:

$$\begin{aligned}
i_t &= \sigma(W_{ei}e_t + W_{hi}h_{t-1} + b_i) \\
f_t &= \sigma(W_{ef}e_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{eo}e_t + W_{ho}h_{t-1} + b_o) \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot tanh(W_{ec}e_t + W_{hc}h_{t-1} + b_c) \\
h_t &= o_t \cdot tanh(c_t)
\end{aligned} \tag{5}$$

The other solution is GRU, which has the gated recurrent unit[6]. The GRU is similar to the LSTM, using gates to control the hidden states. However, instead of using input gate i_t and forget gate f_t to generate a new state, GRU utilizes an update gate z_t. GRU has a reset gate r_t to control the input of the former state h_{t-1}. But unlike LSTM, GRU doesn't use the output gate.

$$\begin{aligned}
r_t &= \sigma(W_{er}e_t + W_{hr}h_{t-1}) \\
z_t &= \sigma(W_{ez}e_t + W_{hz}h_{t-1}) \\
c_t &= \tanh(W_{ec}e_t + W_{hc}(r_t \cdot h_{t-1})) \\
h_t &= (1 - z_t)h_{t-1} + z_t c_t
\end{aligned} \tag{6}$$

We empirically find that LSTM and GRU achieve very similar performance in the evaluation. Given the simplified structure and faster training speed of GRU, we choose GRU for the proposed framework. In our task, each h_t means the representation of the t^{th} product and the micro-behaviors on it. When the context changes, the representation of a product is adjusted according to the whole sequence before it.

4.3 The Attention Layer

As we shown in the last section, the micro-behaviors in the sequence have varied effects on the following micro behaviors. Therefore, it is important to capture such impacts. To achieve this goal, we introduce an attention layer [5, 36, 37] that assigns proper weight on each hidden unit. It helps get a more balanced output. The attention is formed as

$$\begin{aligned}
M_t &= \tanh(W_m h_t + b_m), M_t \in \mathbb{R}^{k \times L} \\
att_t &= \text{softmax}(W_a M_t + b_a), att_t \in \mathbb{R}^L \\
output &= \sum_{t=1}^{T} att_t h_t, output \in \mathbb{R}^k
\end{aligned} \tag{7}$$

where a_t is the attention weight on time t. The attention weight is mapped from the hidden layer vector into a real valued score by a function $\sigma(\cdot)$. In order to achieve enough expressive ability, the $\sigma(\cdot)$ is usually implemented by a neural network layer. One of the implementation is achieved by the above transformation, including tanh and softmax activation functions. Then the final output is an attention weighted pooling of the RNN layer. The significance of the attention layer has two fold. First, it models varied effects of micro-behaviors in the sequence on recommendations. Second, it increases the interpretation ability of the proposed framework since we can rely on the attention layer to understand how the proposed framework works.

4.4 Loss Function

After the output layer, we need to apply a powerful loss function to help learn the weight matrices in the whole model. We choose a pointwise loss function to compute the cross-entropy

$$L = \sum_i -t_i \log(o_i) \tag{8}$$

where t is the normalized target embedding and o is the normalized output embedding.

5 EXPERIMENT

In this section, we empirically evaluate our model on real world e-commerce datasets and demonstrate the effectiveness of our model. We first introduce the datasets and experimental settings. Then we compare the proposed framework with representative baselines and finally we illustrate the working of the proposed framework via a case study.

5.1 Datasets

We collect two datasets with different product categories and sizes from a real e-commerce site. They are named following their product categories, "Appliances" and "Computers". The detailed statistics of the two datasets are shown in Table 2. Both of them include 10 activities and 5 dwell time buckets as described in Table 1. Note that "TNMI" in the table denotes the total number of macro interactions and "TNMB" denotes the total number of micro behaviors.

Table 2: Statistics of Datasets.

Datasets	Appliances		Computers	
	Training	Test	Training	Test
Users	1,058,099	117,566	2,842,871	315,874
SKU	115,498	12,833	134,996	14,999
TNMI	5,290,497	587,833	11,371,485	1,263,498
TNMB	38,091,578	4,232,397	44,222,445	4,913,604

We chooses two widely used metrics to assess the recommendation performance [14, 15, 17, 22, 23, 31, 32]:

- **Recall@k**: The proportion of cases with the target among the top k predictions.
- **MRR@k**: Mean Reciprocal Rank is the average of reciprocal ranks of the target. When the rank is larger than k, the reciprocal rank is set to zero.

Note that we empirically set $k = 20$ in our experiments. To understand the effects of micro behaviors on recommendation performance, we systematically construct the following four datasets for each raw dataset:

- **SKU**: Each entity is a tuple with only a SKU, i.e. $x_t = (p_v)$.
- **SKU+Activity**: Each entity is a tuple with a SKU and an activity, i.e. $x_t = (p_v, a_m)$.
- **SKU+Dwell**: Each entity is a tuple with a SKU and a dwell time bucket, i.e. $x_t = (p_v, d_k)$.
- **SKU+Micro-behaviors**: Each entity is a micro behavior tuple with a SKU, an activity and a dwell time bucket, i.e. $x_t = (p_v, a_m, d_k)$.

5.2 Performance Comparison

We compare our model with the following representative methods.

- **POP** is a naive baseline that always recommends the most popular items of the training set. It's a non-personalized recommendation method, however, it has been proved to be comparable to some sophisticated personalized algorithms [8].
- **BPR-MF** is one of the most commonly used matrix factorization methods defined in [28].
- **Item-KNN** is defined as conditional probability-based similarity in [9]. The similarity between items can be defined as $sim(i,j) = \frac{Freq(ij)}{Freq(i) \times Freq(j)}$, where $Freq(t)$ means the number of sequences where an item t shows up.
- **Word2vec** [26] is a shallow, two-layer neural network that is trained to reconstruct linguistic contexts of words. In our work, we could regard each item as a word and train it via Word2vec. And we only use the word vector of the last item in a sequence for prediction.
- **Word2vec Avg.** It applies the same embeddings trained from Word2vec method. The difference is that it uses the average of all the embeddings in a sequence for prediction.
- **RIB-Attention**: It is a variant of the proposed framework RIB by removing the attention layer.

Note that in this work, we choose GRU as the RNN layer for both RIB and RIB-Attention. We implement our model using Keras [5]. The experiments are run on a Tesla K20 GPU. The comparison results are shown in Table 3. Note that numbers in the parentheses denote the performance improvement compared to the best baseline. Since the performance of Word2vec and Word2vec Avg. are different on the two datasets. For the "Appliances" dataset, we choose Word2vec as the best baseline. For the "Computers", we choose Word2vec Avg. as the best baseline. From Table 3, we made the following observations:

- Methods capturing sequential information including Word2vec, Word2vec Avg, RIB and RIB-Attention outperforms those ignoring sequential information such as POP, BPR-MF and Item-KNN.
- RIB obtains better performance than RIB-Attention. These results support that capturing varied effects of micro behaviors can boost the performance of recommendation.
- Among SKU, SKU+Activity, SKU+Dwell and SKU + Micro-behaviors, RIB consistently obtains the best performance on SKU+Micro-behaviors. These observations suggest the importance of micro-behaviors on recommendations.

To sum up, the proposed framework achieves the best performance because (1) it models sequential information; (2) it captures micro behaviors and (3) it models varied effects of micro behaviors. In the following subsection, we use a case study to understand why the proposed framework works.

[5] https://keras.io

Table 3: Performance Comparison

Data type	Model	Appliances		Computers	
		Recall@20	MRR@20	Recall@20	MRR@20
SKU	POP	0.0088	0.0028	0.0149	0.0085
	BPR-MF	0.1255	0.0578	0.0747	0.0271
	Item-KNN	0.1806	0.0738	0.1101	0.0464
	Word2vec	0.3645	0.1295	0.3012	0.1044
	Word2vec Avg.	0.3668	0.1268	0.3152	0.1088
	RIB-Attention	0.4587(+25.84%)	0.1676(+29.42%)	0.3816(+21.07%)	0.1362(+25.18%)
	RIB	0.4732(+29.82%)	0.1718(+32.66%)	0.4043(+28.27%)	0.1456(+33.82%)
SKU+Activity	RIB-Attention	0.4615(+26.61%)	0.1724(+33.13%)	0.4092(+29.82%)	0.1443(+32.63%)
	RIB	0.4842(+32.82%)	0.1776(+37.14%)	0.4204(+33.38%)	0.1481(+36.12%)
SKU+Dwell	RIB-Attention	0.4673(+28.2%)	0.1745(+34.75%)	0.4108(+30.33%)	0.1482(+36.21%)
	RIB	0.4822(+32.29%)	0.1766(+36.37%)	0.4269(+35.44%)	0.149(+36.95%)
SKU+Micro-behaviors	RIB-Attention	0.474(+30.04%)	0.1784(+37.76%)	0.4227(+34.11%)	0.1516(+39.34%)
	RIB	0.4889(+34.13%)	0.1793(+38.46%)	0.4332(+37.44%)	0.1533(+40.9%)

5.3 A Case Study

In the last subsection, we demonstrate that the proposed framework outperforms various representative baselines. In this subsection, we aim to understand why the proposed framework RIB works. As discussed before, the attention layer equipped the proposed framework with the ability of interpretation. Therefore, we can understand the effects of micro behavior on recommendations via visualizing the attention layer.

Figure 8 illustrates an example from the "Appliances" dataset and we visualize the attention layers of the proposed framework on SKU, SKU+Activity, SKU+Dwell and SKU+Micro-behaviors. The left part shows a sequence of macro interactions and micro behaviors. The left bottom is what the customer actually browses in this sequence for the next time. There are 4 types of input tuples to show the influence of micro behaviors. The predicted rank of the ground truth is shown at the right bottom. From this case study, it can be observed:

- When the input tuple only consists of SKU, the attention is mainly decided by position and repetition of the product. Since Joyoung Juicer is the last of the input and shows twice in the sequence, the two Joyoung products have higher attention.
- When the activity is added to SKU, the attention becomes more fine grained. The last Joyoung still has the highest attention. But Aux Blender is getting more attention. This may be due to the activity number and the "Cart" and "Detail_specification" activities.
- When dwell time is added to SKU, the difference is that Aux Blender gets higher attention than the first Joyoung Juicer. This corresponds to the dwell time on these two products. Aux Blender has the longest dwell time which may mean the customer's deeper interests.
- When both the activity and dwell time are added to SKU, Aux Blender and Joyoung Juicer get higher attention. This may due to their "Cart" activity and long dwell time.
- From the final prediction result, we could tell the attention change corresponds to the rank precision. Micro-behaviors help understand the macro sequence and thus improve the prediction.

Via the case study, the proposed framework not only can achieve more accurate recommendations via modeling micro behaviors but also can naturally provide the interpretation for recommendations.

6 RELATED WORK

The most common and traditional methods used in item recommendation are matrix factorization models and neighborhood methods. The matrix factorization models learn a latent representation of users and items that, when decomposed, produce an approximation of the rating that a user would give to an item. There exist a lot of improved factorization-based models, such as non-negative matrix factorization, Bayesian personalized ranking[28], general factorization framework[2], hierarchical Poisson factorization[11], dynamic Poisson factorization[4], etc. Another widely used method for item recommendation is KNN[21]. The similarity between two items is decided by the co-occurrences of them in sessions. There are also some

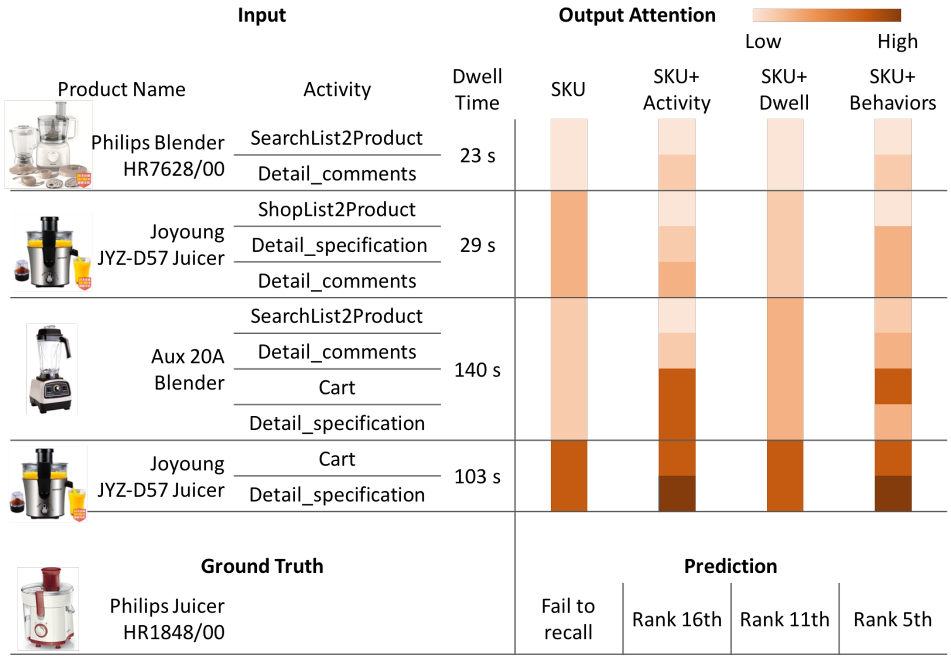

Figure 8: A case to understand the working of the proposed framework.

work combining contextual information for recommendation[20, 27]. But these methods rarely apply users' behaviors directly and capture the change of behaviors slowly. And deep learning methods perform better on these problems.

6.1 Deep Learning for Recommendation

Deep learning methods have been applied in kinds of recommendation problems. [32] achieves improved recommendation accuracy by jointly performing deep representation learning (a Bayesian model) for the item content information and collaborative filtering for the ratings matrix. [7] uses deep neural network (DNN) for recommending YouTube videos, focusing on solving candidate generation and ranking problems. [10] uses Multi-view DNN to match rich user features to item features. The proposed model is scalable to large datasets for dimensionality reduction, and it combines different domains into a single model for learning. [39] makes use of stacked convolutional auto-encoder to extract the semantic representation of visual content. And a joint model is designed by integrating collaborative filtering and visual, textual, structural knowledge to boost recommendation quality.

6.2 RNN on Sequence Data

Recurrent neural network (RNN) [33] is a deep learning model helping process sequential data. Long Short-Term Memory (LSTM) model [16] is a RNN architecture which is suitable when there are very long time lags between important events in a session. Gated Recurrent Units (GRU) [6] is a simple version of LSTM.

There exists many works using RNN to classify, process or predict sequence data. [38] proposes an attention and RNN based architecture for modeling queries and ads in online advertising. It learns to assign attention scores to words within a sequence. The scores can be successfully applied in query rewriting tasks and can also help modify BM25 metric performance. [40] introduces an RNN framework for click prediction. It models the dependency on user's sequential behaviors into click prediction through the recurrent structure in RNN. This significantly improves the prediction accuracy. And by enriching user features as input, this work fairly compares the prediction capability of different user behaviors. [13] demonstrates an RNN-based item categorization method called DeepCN. DeepCN uses multiple RNNs for generating features from text metadata and uses fully connected layers for classifying item categories from the generated features. It improves the categorization accuracy compared to the models using single RNN or using unigram-based bag-of-words. In [3], RNNs are leveraged to provide vector representations for the text content associated with items in collaborative filtering. It helps perform cold-start prediction on new items.

In recommender systems, RNN is just started to be used for session-based recommendations. In [14], Hidasi novelly applies GRU on session-based recommending problems. Comparing with item-to-item recommendations, this method proves that considering the whole session instead of one item provides more accurate recommendations. But [14] only uses item IDs as features. For a richer representation of the clicked items, in [15], Hidasi continues his work by applying parallel RNNs for processing both text and image features. [30] and [19] also solve the session-based recommendation using recurrent networks. [31] provides two new session-aware recommendation methods, matrix factorization and RNN. It concludes that the usage of RNN is considered more attractive and flexible when no session modeling assumptions is made.

[17] uses LSTM to learn users' returning time prediction and item recommendation. It shows the problem of survival analysis can be solved by RNN.

7 CONCLUSIONS

In this paper, we study recommender systems from the micro behaviors perspective. Micro behaviors provide deeper understandings about users, which can be utilized to advance recommender sytems. The new perspective provides a unified setting for several existing ones in traditional recommendations. However, modeling micro behaviors is challenging, which motivates us to develop a novel framework RIB. Experimental results on datasets from a real e-commerce site demonstrate the effectiveness of the proposed framework and uncover the importance of micro behaviors in recommendations.

In this work, we only consider four types of micro behaviors. We would like to study more types of micro behaviors and their effects on recommendations. Meanwhile, our current focus is the domain of e-commerce, however, micro behaviors also exist in other domains such as movies and news. Therefore, we would like to recommender systems in more domains from the micro behavior perspective.

ACKNOWLEDGEMENTS

Jiliang Tang is supported by the National Science Foundation (NSF) under grant number IIS-1714741 and IIS-1715940.

REFERENCES

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving Web Search Ranking by Incorporating User Behavior Information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06)*. ACM, New York, NY, USA, 19–26. https://doi.org/10.1145/1148170.1148177

[2] Hidasi Balazs and Tikk Domonkos. 2015. General factorization framework for context-aware recommendations. *Data Mining and Knowledge Discovery* (2015), 1–30.

[3] T. Bansal, D. Belanger, and A. Mccallum. 2016. Ask the GRU : Multi-task Learning for Deep Text Recommendations. *ACM Conference on Recommender Systems* (2016).

[4] L. Charlin, R. Ranganath, J. McInerney, and D. M. Blei. 2015. Dynamic poisson factorization. *CoRR* (2015).

[5] Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. 2015. ABC-CNN: An Attention Based Convolutional Neural Network for Visual Question Answering. *CoRR* abs/1511.05960 (2015). http://arxiv.org/abs/1511.05960

[6] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *CoRR* abs/1409.1259 (2014). http://arxiv.org/abs/1409.1259

[7] P. Covington, J. Adams, and E. Sargin. 2016. Deep Neural Networks for YouTube Recommendations. *Proceedings of the 10th ACM Conference on Recommender Systems* (2016), 191–198.

[8] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2011. Performance of recommender algorithms on top-n recommendation tasks. *RecSys* (2011).

[9] Mukund Deshpande and George Karypis. 2004. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* 22, 1 (Jan. 2004), 143–177. https://doi.org/10.1145/963770.963776

[10] A. Elkahky, Y. Song, and X. He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. *WWW* (2015).

[11] P. Gopalan, J. M. Hofman, and D. M. Blei. 2015. Scalable recommendation with hierarchical poisson factorization. *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence,* (2015).

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

[13] J. W. Ha, H. Pyo, and J. Kim. 2016. Large-Scale Item Categorization in e-Commerce Using Multiple Recurrent Neural Networks. *KDD* (2016).

[14] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. *ICLR* (2016).

[15] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. *Proceedings of the 10th ACM Conference on Recommender Systems* (2016), 241–248.

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

[17] How Jing and Alexander J Smola. 2017. Neural Survival Recommender. *WSDM* (2017).

[18] Youngho Kim, Ahmed Hassan, Ryen W. White, and Imed Zitouni. 2014. Modeling Dwell Time to Predict Click-level Satisfaction. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. ACM, New York, NY, USA, 193–202. https://doi.org/10.1145/2556195.2556220

[19] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. *CIKM* (2017).

[20] P Li, Z Wang, Z Ren, L Bing, and W Lam. 2017. Neural Rating Regression with Abstractive Tips Generation for Recommendation. *SIGIR* (2017).

[21] G Linden, B Smith, and J York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE* 7, 1 (2003), 76–80.

[22] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware Sequential Recommendation. *CoRR* abs/1609.05787 (2016). http://arxiv.org/abs/1609.05787

[23] Qiang Liu, Shu Wu, and Liang Wang. 2017. Multi-Behavioral Sequential Prediction with Recurrent Log-Bilinear Model. *IEEE Trans. on Knowl. and Data Eng.* 29, 6 (June 2017), 1254–1267. https://doi.org/10.1109/TKDE.2017.2661760

[24] Wei Lu, Shanshan Chen, Keqian Li, and Laks V. S. Lakshmanan. 2014. Show Me the Money: Dynamic Recommendations for Revenue Maximization. *Proc. VLDB Endow.* 7, 14 (Oct. 2014), 1785–1796. https://doi.org/10.14778/2733085.2733086

[25] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).

[26] 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 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[27] Z Ren, S Liang, P Li, S Wang, and M de Rijke. 2017. Social Collaborative Viewpoint Regression with Explainable Recommendations. *The 10th ACM International Conference on Web Search and Data Mining* (2017).

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. (2009), 452–461. http://dl.acm.org/citation.cfm?id=1795114.1795167

[29] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. *Collaborative Filtering Recommender Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 291–324. https://doi.org/10.1007/978-3-540-72079-9_9

[30] Y. K. Tan, X. Xu, and Y. Liu. 2016. Improved recurrent neural networks for session-based recommendations. *CoRR* (2016).

[31] B. Twardowski. 2016. Modelling Contextual Information in Session-Aware Recommender Systems with Neural Networks. *ACM Conference on Recommender Systems* (2016), 273–276.

[32] H. Wang, N. Wang, and D. Yeung. 2015. Collaborative Deep Learning for Recommender Systems. *KDD* (2015).

[33] P. J. Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proc. IEEE* 78, 10 (1990), 1550–1560.

[34] Chenyan Xu, Daniel Peak, and Victor Prybutok. 2015. A customer value, satisfaction, and loyalty perspective of mobile application recommendations. *Decision Support Systems* 79 (2015), 171–183.

[35] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond Clicks: Dwell Time for Personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. ACM, New York, NY, USA, 113–120. https://doi.org/10.1145/2645710.2645724

[36] Wenpeng Yin, Sebastian Ebert, and Hinrich Schütze. 2016. Attention-Based Convolutional Neural Network for Machine Comprehension. *CoRR* abs/1602.04341 (2016). http://arxiv.org/abs/1602.04341

[37] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *CoRR* abs/1512.05193 (2015). http://arxiv.org/abs/1512.05193

[38] S. Zhai, H. Chang, R. Zhang, and Z. Zhang. 2016. DeepIntent: learning attentions for online advertising with recurrent neural networks. *KDD* (2016).

[39] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. *KDD* (2016).

[40] Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang, and T. Liu. 2014. Sequential Click Prediction for Sponsored Search with Recurrent Neural Networks. *AAAI* (2014).