Multi-Behavior Sequential Transformer Recommender

Enming Yuan*
yem19@mails.tsinghua.edu.cn
Institute for Interdisciplinary Information Sciences, Tsinghua University
Beijing, China

Wei Guo
guowei67@huawei.com
Noah's Ark Lab, Huawei
Shenzhen, China

Zhicheng He
hezhicheng9@huawei.com
Noah's Ark Lab, Huawei
Shenzhen, China

Huifeng Guo
huifeng.guo@huawei.com
Noah's Ark Lab, Huawei
Shenzhen, China

Chengkai Liu
liuchengkai@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Ruiming Tang†
tangruiming@huawei.com
Noah's Ark Lab, Huawei
Shenzhen, China

ABSTRACT

In most real-world recommender systems, users interact with items in a sequential and multi-behavioral manner. Exploring the fine-grained relationship of items behind the users' multi-behavior interactions is critical in improving the performance of recommender systems. Despite the great successes, existing methods seem to have limitations on modelling heterogeneous item-level multi-behavior dependencies, capturing diverse multi-behavior sequential dynamics, or alleviating data sparsity problems. In this paper, we show it is possible to derive a framework to address all the above three limitations. The proposed framework MB-STR, a **M**ulti-**B**ehavior **S**equential **T**ransformer **R**ecommender, is equipped with the multi-behavior transformer layer (MB-Trans), the multi-behavior sequential pattern generator (MB-SPG) and the behavior-aware prediction module (BA-Pred). Compared with a typical transformer, we design MB-Trans to capture multi-behavior heterogeneous dependencies as well as behavior-specific semantics, propose MB-SPG to encode the diverse sequential patterns among multiple behaviors, and incorporate BA-Pred to better leverage multi-behavior supervision. Comprehensive experiments on three real-world datasets show the effectiveness of MB-STR by significantly boosting the recommendation performance compared with various competitive baselines. Further ablation studies demonstrate the superiority of different modules of MB-STR.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Multi-Behavior Modeling, Sequential Recommendation, Transformer

*Work done when he was a research intern at Noah's Ark Lab, Huawei.
†Ruiming Tang is the corresponding author.

SIGIR '22, July 11–15, 2022, Madrid, Spain.

ACM ISBN 978-1-4503-8732-3/22/07.
https://doi.org/10.1145/3477495.3532023

ACM Reference Format:
Enming Yuan, Wei Guo, Zhicheng He, Huifeng Guo, Chengkai Liu, and Ruiming Tang. 2022. Multi-Behavior Sequential Transformer Recommender. In *Proceedings of the 45th Int'l ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3477495.3532023

1 INTRODUCTION

Personalized recommender systems have been playing an important role in many online service platforms, from online advertising and online retailing [4, 12, 37] to music and video recommendation [5, 6, 35]. To provide precise and customized services, these systems attempt to recommend products that users are likely to be interested in based on their historical interaction data.

In most real-world recommendation scenarios, there are two important characteristics of user interaction data: **sequential** and **multi-behavioral**. Users interact with items in a **sequential** manner, and their interest patterns are intrinsically diverse and keep evolving [23, 42, 43]. Therefore, it is important to consider the informative sequential dynamics of user behaviors in the recommendation model. Additionally, users interact with items in a **multi-behavioral** way. For example, interactions take many forms of behavior in an E-commerce platform, including *click*, *add-to-favorite*, *add-to-cart*, and *purchase*. Such multi-behavioral property brings us two significant benefits. First, different types of behaviors, such as *click* and *purchase*, reflect different user intentions. Therefore, the multi-behavior interaction data provides us with an opportunity to capture the fine-grained interest dynamic of users. Second, *target behavior* data (e.g., *purchase* on E-commerce platforms, which are conventionally the most concerned.) is usually very sparse. Severe cold-start problems will happen if we model *target behavior* data independently. Fortunately, this problem can be alleviated if we utilize the plentiful *auxiliary behavior* data reasonably.

Recently, a bunch of work has been proposed to model user interaction data from either sequential (e.g., GRU4Rec [17], Caser [34], SASRec [20], BERT4Rec [33]) or multi-behavioral (e.g., NMTR [9], MATN [39], MB-GCN [19]) perspectives. However, despite the potential benefits of taking both **sequential** and **multi-behavioral** properties of user behavior into consideration, the multi-behavior sequential recommendation problem is still underexplored. Specifically, there are three major challenges making it a non-trivial problem:

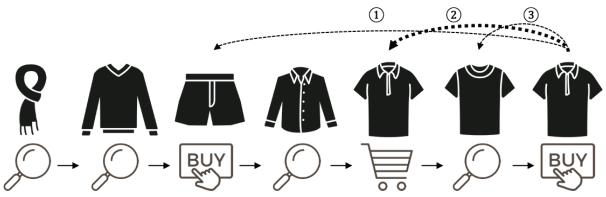


Figure 1: An example of multi-behavior sequential recommendation. A user interacts with items through different behaviors. Arrows indicate plausible explanations for the final purchasing decision.

- ***C1:*** One important characteristic of multi-behavior data is the behavior-specific semantics, since different behaviors reflect different user intentions. Moreover, items that interacted with different intentions will form complex fine-grained multi-behavior dependencies. For example, in Figure 1, the final purchasing decision is affected by other historical actions from different aspects, including the purchase of a matching shorts (arrow ①), the add-to-cart of the same T-shirt (arrow ②), and the recent click of other T-shirts (arrow ③). Previous work (e.g., MATN [39], MB-GCN [19], DIPN [13] and DMT [11]) follows a two-stage aggregation paradigm, which first aggregates items under each behavior to get a unified representation, and then models the dependencies for all behaviors through attention or weighted summation operation. However, it should be noted that these works only explicitly model the **behavior-level** dependencies (arrow ①) and fail to capture the fine-grained item-to-item relationship among multi-behavior (arrow ② & ③), which we defined as **item-level** dependencies. Moreover, the **heterogeneous** properties of item relations should also be considered as there exist multiple behavior transition patterns if we model **item-level** dependencies. Therefore, Challenge 1 is: *how to model **heterogeneous** multi-behavior dependencies at the fine-grained **item-level**?*
- ***C2:*** The dependencies across different types of user-item interactions become even more complex when we consider the sequential information. In Figure 1, take *click* and *purchase* as an example. *Click* represents short-term interest, so the recently clicked items (arrow ③) may impact the current purchasing decision, while the items clicked long ago usually have little influence on the current purchasing decision. However, for *purchase*, items purchased a long time ago may still strongly influence current interest (arrow ①). This leads to **diverse multi-behavior** sequential patterns. However, few existing multi-behavior models consider sequential information. Although DIPN [13] and DMT [11] aggregate features in a sequential manner, the sequential patterns among multiple types of behaviors are not distinguished, with a **fixed single behavior** pattern. Thus Challenge 2 is: *how to model **diverse multi-behavior** sequential patterns effectively?*
- ***C3:*** The problem of target behavior data sparsity can be alleviated by integrating plentiful auxiliary behavior data. However, if we only use auxiliary behavior data as features instead of training signals (e.g., MATN [39], MBGCN [19]), rich supervision signals from auxiliary behavior are ignored. Moreover, simply using auxiliary behavior as supervision signals may lead to performance degradation and negative transfer due to the complex behavior correlation. Therefore, appropriate designs are required. So Challenge 3 is: *how to effectively mine users' multi-behavior sequence with multi-behavior supervision signals?*

In this paper, to address the above challenges, we propose **M**ulti-**B**ehavior **S**equential **T**ransformer **R**ecommender (**MB-STR**) framework. First, to model the fine-grained multi-behavior dependencies in users' interaction sequences (***C1***), we design a novel Multi-Behavior Transformer layer (MB-Trans), which performs heterogeneous behavior aggregation and behavior-specific transformation. Second, to capture diverse multi-behavior sequential patterns (***C2***), a Multi-Behavior Sequential Pattern Generator (MB-SPG) is proposed. This module cooperates with the MB-Trans module to encode the diverse multi-behavior sequential patterns into attention bias matrices. Finally, to better leverage auxiliary data to facilitate target behavior prediction (***C3***), we propose the Behavior-Aware masked item Prediction module (BA-Pred), which can mine the multi-behavior interaction sequence effectively.

In summary, the major contributions of this work are as follows:

- We highlight the sequential and multi-behavioral nature of real-world user interactions and summarize the challenges in the multi-behavior sequential recommendation problem.
- To address the multi-behavior sequential recommendation problem, we propose a new framework named MB-STR, which has three key components. The heterogeneous item-level dependencies are modeled through a novel Multi-Behavior Transformer (MB-Trans) layer. Moreover, a multi-behavior sequential pattern generator module (MB-SPG) is incorporated to encode the diverse multi-behavior sequential pattern. Furthermore, a Behavior-Aware masked item Prediction module (BA-Pred) is used to facilitate the prediction of target behavior by training on both target behavior and auxiliary behavior data.
- We conduct extensive experiments on three real-world datasets. Experimental results demonstrate the effectiveness of MB-STR as compared with several baseline models. Further ablation studies explain the superiority of our designed modules.

2 RELATED WORK

2.1 Sequential Recommendation

Sequential recommenders are designed to model the sequential dynamics in user behaviors to which various sequence models have been applied, including Recurrent Neural Network (RNN) [28], Convolutional Neural Network (CNN) [22], and Transformer [36]. For example, Hidasi et al. use the Gated Recurrent Unit (GRU) network to model user click sequences [16, 17]. By taking the sequence embedding matrix as an image, CNN is applied to extract user interest patterns in the Caser model [34]. Due to the high model capacity and effective training techniques, many Transformer-based sequential recommendation methods are proposed and achieve significant improvements, such as SASRec [20], TiSASRec [24], and BERT4Rec [33]. For the same reason, the proposed model in this paper is also based on Transformer, while many improvements are made for multi-behavior modeling.

Table 1: Comparison of related works.

	Multi-Behavior Modeling	Sequential Information	Behavior-Specific Prediction
MATN [39]	behavior-level	✗	✗
NMTR [9]	behavior-level	✗	✓
MBGCN [19]	behavior-level	✗	✗
MB-GMN [40]	behavior-level	✗	✓
DIPN [13]	behavior-level	fixed single behavior	✓
DMT [11]	behavior-level	fixed single behavior	✓
MB-STR(our)	heterogeneous item-level	diverse multi-behavior	✓

2.2 Multi-Behavior Recommendation

Multi-behavior recommendation utilizes multiple types of user-item interactions to enhance the performance regarding target behavior. Previous works can be divided into three categories. The first category of works considers the behavior dependencies and only uses the target behavior data as supervision signals such as the attentive memory network MATN [39] and the graph convolutional model MBGCN [19]. Another line of works further considers multi-task learning to utilize both target and auxiliary behavior data for supervision, including the cascaded prediction model NMTR [9] and the multi-task graph meta network MB-GMN [40]. The last category of works considers both the sequential information and the supervision signals in multiple user behaviors, such as the multi-view sequential network DIPN [13] and the MMoE-based [18, 26] multi-sequence model DMT [11].

As shown in Table 1, we briefly compare the representative multi-behavior models and our MB-STR from three aspects:

- *Multi-behavior modeling*: Models like MATN [39], MBGCN [19], DIPN [13] and DMT [11] follow a two-stage aggregation paradigm and only model coarse-grained behavior-level dependencies. Differently, our MB-STR can model heterogeneous item-level multi-behavior dependencies via the multi-behavior transformer.
- *Sequential information*: Models like NMTR [9], MBGCN [19], MATN [39] and MB-GMN [40] do not consider sequential information. DIPN [13] and DMT [11] only capture the sequential pattern with a fixed single behavior. In MB-STR, we propose the MB-SPG component to model diverse multi-behavior sequential patterns.
- *Behavior-Specific Prediction*: NMTR [9], MB-GMN [40], DIPN [13] and DMT [11] are trained in a multi-task manner with behavior specific prediction to utilize the rich supervision signals from auxiliary behaviors. However, they ignore the shared and task-specific knowledge exists in multi-behavior recommendation. We further separate shared components and behavior-specific components explicitly to alleviate the conflicting parameter interference between different behaviors.

3 PRELIMINARY

3.1 Problem Definition

Let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ represent the set of users and items, respectively, where $|\mathcal{U}|$ and $|\mathcal{V}|$ denote the number of users and items. Suppose there are $|\mathcal{B}|$ types of behavior. Also, the set of behavior is denoted as $\mathcal{B} = \{B_1, B_2, ..., B_{|\mathcal{B}|}\}$, among which one type of behavior we care about most is regarded as the *target behavior*, and the other types of behaviors are called *auxiliary behaviors*. For each user, her historical interaction forms a multi-behavior interaction sequence X_u, defined as follows:

Definition 1. ***Multi-Behavior Interaction Sequence*** X_u. *X_u consists of a pair of sequences $(\mathbf{x}_u, \mathbf{b}_u)$. Each pair of $(\mathbf{x}_u[i], \mathbf{b}_u[i])$ describes a user interaction, which is composed of an item $\mathbf{x}_u[i] \in \mathcal{V}$ and the corresponding interacting behavior type $\mathbf{b}_u[i] \in \mathcal{B}$. The interaction actions are sorted in chronological order. The X_u is further transformed to a fixed length n: if the sequence length is less than n, special* `[padding]` *tokens are padded to the left as dummy past interactions on both $\mathbf{x}_u$ and $\mathbf{b}_u$. If the sequence length is greater than n, the most recent n actions are kept.*

Then the multi-behavior sequential recommendation problem can be formulated as follows: Given the multi-behavior interaction sequences for all users $\mathbf{X}$, the set of users $\mathcal{U}$, the set of items $\mathcal{V}$, and the set of behavior types $\mathcal{B}$, we aim to train a model that takes multi-behavior interaction sequence $X_{(u)}$ of user u as input, and estimates the probability that u interacts with an item i under the target behavior at time step $n+1$.

3.2 Transformer in Recommendation

Improvements in the field of Natural Language Processing have driven many recent advances in recommender systems, especially sequential recommendation [20, 33]. In particular, equipped with higher model capacities and efficient training techniques, the Transformer architecture [36] significantly improves the effectiveness of sequential recommendation.

As the Transformer architecture is highly relevant to our work, we briefly describe the key components of Transformer here, including Multi-head Self-Attention (MSA) and Multi-Layer Perceptron (MLP). For a sequence input, MSA and MLP perform whole sequence aggregation and position-wise transformation, respectively.

3.2.1 MSA. At each head of the MSA, the inputs $\mathbf{X} \in \mathbb{R}^{n \times d}$ are linearly transformed to three hidden representations, i.e., queries $\mathbf{Q}_i \in \mathbb{R}^{n \times \frac{d}{h}}$, keys $\mathbf{K}_i \in \mathbb{R}^{n \times \frac{d}{h}}$ and values $\mathbf{V}_i \in \mathbb{R}^{n \times \frac{d}{h}}$, where i indicates a specific head, n is the sequence length, d is the dimensionality of inputs and h is the total number of heads. Then, the scaled dot-product attention is calculated as below:

$$\mathbf{Attn}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = softmax\left(\frac{\mathbf{Q}_i \mathbf{K}_i^\top}{\sqrt{d_k}}\right)\mathbf{V}_i. \tag{1}$$

Multi-head self-attention performs the above self-attention mechanism h times in parallel, and then the outputs of each head are concatenated together and linearly projected to get the final output:

$$\mathbf{MSA}(\mathbf{X}) = \text{Concat}(\mathbf{Attn}(\mathbf{Q}_1, \mathbf{K}_1, \mathbf{V}_1), ..., \mathbf{Attn}(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h))W_o. \tag{2}$$

3.2.2 MLP. The MLP is used for introducing non-linearity and feature transformation between MSA layers:

$$\text{MLP}(\mathbf{X}) = \text{FC}(\sigma(\text{FC}(\mathbf{X}))), \text{FC}(\mathbf{X}) = \mathbf{XW} + \mathbf{b}, \tag{3}$$

where $\mathbf{b}$ and $\mathbf{W}$ are the bias and weights of a fully-connected layer, and $\sigma(\cdot)$ is the activation function such as ReLU [10].

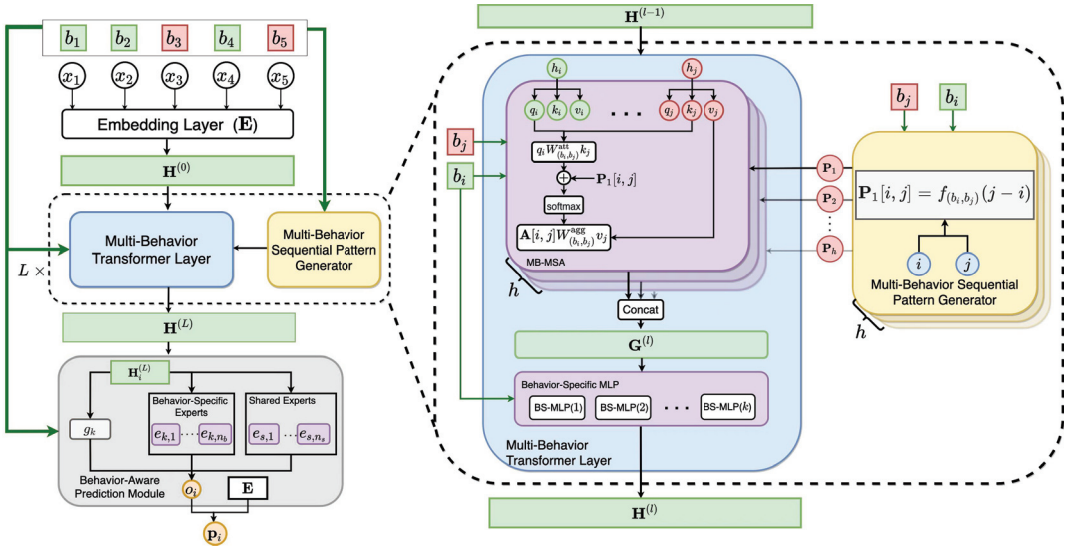


Figure 2: The model architecture of MB-STR. The overall structure is shown in the left panel. The zoom-in view of the MB-Trans module and MB-SPG module are shown in the middle and right panel, respectively.

4 MB-STR

We now introduce the MB-STR framework. As illustrated in Figure 2, there are three key components: (1) the Multi-Behavior Transformer layer (MB-Trans); (2) the Multi-Behavior Sequential Pattern Generator (MB-SPG); and (3) Behavior-Aware masked item Prediction module (BA-Pred).

As shown in Figure 2, we first transform the item sequence $\mathbf{x}$ to an input hidden representation matrix $\mathbf{H}^{(0)}$ through an embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$. Then, we iteratively compute hidden representations $\mathbf{H}^{(l)}$ at each MB-Trans layer l (Section 4.1). Simultaneously, the MB-SPG module is coupled with the MB-Trans layer to inject multi-behavior sequential patterns (Section 4.2). Finally, the BA-Pred module predicts the masked items with respect to their corresponding behavior type (Section 4.3). Next, we will elaborate on these three components.

4.1 Multi-Behavior Transformer Layer

Typically, the inputs to the transformer, whether they are words [7, 30], audio segments [3, 32] or image patches [8, 25], are essentially homogeneous. However, in the multi-behavior sequential recommendation scenario, items are interacted by different types of behaviors. This leads to **behavior-specific semantics** and **fine-grained heterogeneous dependencies**, which vanilla transformer fails to capture. Here, we propose a novel multi-behavior transformer (MB-Trans) layer, which captures fine-grained heterogeneous dependencies through a multi-behavior multi-head self-attention mechanism (MB-MSA) and models behavior semantics by behavior-specific MLPs.

4.1.1 Multi-Behavior Multi-head Self-Attention. To endow MSA with the ability to capture multi-behavior heterogeneous dependencies, we generalize the original MSA to Multi-Behavior Multi-head Self-Attention (MB-MSA). Overall, MB-MSA differs from the original MSA in four folds: (1) behavior-specific linear projection is applied to get queries, keys and values; (2) behavior heterogeneity is considered when calculating attention scores; (3) sequential patterns are injected to the raw attention matrix; and (4) behavior heterogeneity is considered when aggregating values.

Algorithm 1: Multi-Behavior Multi-head Self-Attention

Input: $\mathbf{H}^{(l-1)} \in \mathbb{R}^{n \times d}, \mathbf{b} \in \mathcal{B}^n, \mathbf{P}^{(l)} \in \mathbb{R}^{h \times n \times n}$
Output: $\mathbf{G}^{(l)} \in \mathbb{R}^{n \times d}$

for *head* $m = 1$ *to* h **do**
  /* Step 1. Behavior-specific projection. */
  $\mathbf{Q}_m \leftarrow f_{Q_m}(\mathbf{H}^{(l-1)}, \mathbf{b})$
  $\mathbf{K}_m \leftarrow f_{K_m}(\mathbf{H}^{(l-1)}, \mathbf{b})$
  $\mathbf{V}_m \leftarrow f_{V_m}(\mathbf{H}^{(l-1)}, \mathbf{b})$
  /* Step 2. Cross behavior similarity. */
  for $i = 1$ *to* n, $j = 1$ *to* n **do**
    $\mathbf{A}_m[i,j] = \frac{\mathbf{Q}_m[i]\mathbf{W}^{att}_{(\mathbf{b}[i],\mathbf{b}[j])}\mathbf{K}_m[j]}{\sqrt{d}}$
  end
  /* Step 3. Sequential pattern injection and softmax. */
  $\mathbf{A}_m \leftarrow softmax(\mathbf{A}_m + \mathbf{P}[m])$
  /* Step 4. Cross behavior information aggregation. */
  for $i = 1$ *to* n **do**
    $\mathbf{G}^{(l)}_m[i] \leftarrow \sum_j \mathbf{A}_m[i,j] \cdot \mathbf{W}^{agg}_{(\mathbf{b}[i],\mathbf{b}[j])} \cdot \mathbf{V}_m[j]$
  end
end
$\mathbf{G}^{(l)} \leftarrow \mathbf{Concat}(\mathbf{G}^{(l)}_1, ..., \mathbf{G}^{(l)}_h)$

The details of MB-MSA are described in Algorithm 1 and illustrated in Figure 2. Let $\mathbf{H}^{(l-1)} \in \mathbb{R}^{n \times d}$ be the input of the l-th layer, where n is the behavior sequence length, d is the model dimension.

$\mathbf{b} \in \mathcal{B}^n$ is the corresponding behavior sequence indicating the behavior type at each position of $\mathbf{H}^{(l-1)}$.

First, to distinguish intentions under different types of behaviors, we perform behavior-specific linear projection $f_{(\cdot)}(\mathbf{H}^{(l-1)}, \mathbf{b}) : \mathbb{R}^{n\times d} \rightarrow \mathbb{R}^{n\times \frac{d}{h}}$ to get $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$, respectively (**Step 1.** in Algorithm 1). $f_{(\cdot)}(\mathbf{H}^{(l-1)}, \mathbf{b})$ is a wrapper of $|\mathcal{B}|$ linear layers, each corresponds to a specific type of behavior. At each position i, $\mathbf{H}^{(l-1)}[i]$ is projected according to the behavior type $\mathbf{b}[i]$.

Next, we want to calculate the raw attention score with respect to multi-behavior dependencies. To achieve this, instead of directly calculating the dot product between the Queries and Keys, we construct a distinct parameter matrix $\mathbf{W}^{att}_{(B_i,B_j)} \in \mathbb{R}^{\frac{d}{h}\times\frac{d}{h}}$ for each behavior pair (**Step 2.** in Algorithm 1). Pairwise multi-behavior dependencies can be captured in this way. High-order dependencies can also be modeled by stacking multiple layers.

Furthermore, the self-attention mechanism is not sensitive to sequence order, let alone the complex multi-behavior sequential patterns. To address this problem, we inject sequential information by adding multi-behavior sequential pattern matrix $\mathbf{P}^{(l)}$ generated by the MB-SPG module (Section 4.2) to the raw attention matrix $\mathbf{A}$ (**Step 3.** in Algorithm 1) and apply a softmax operation.

Finally, we would aggregate information from the values $\mathbf{V}$ regarding behavior heterogeneity to alleviate the distribution difference between different behaviors. Thus, similar to the calculation of attention, we construct a distinct parameter matrix $\mathbf{W}^{agg}_{(B_i,B_j)} \in \mathbb{R}^{\frac{d}{h}\times\frac{d}{h}}$ for each behavior pair (**Step 4.** in Algorithm 1).

MB-MSA performs the above attention h times in parallel. The outputs of h heads are concatenated to get the final output $\mathbf{G}^{(l)}$.

4.1.2 Behavior-Specific Multi-Layer Perceptron. To model behavior semantics and perform feature transformation, we apply Behavior-Specific MLPs (BS-MLP). Specifically, we specify a distinct MLP layer for each type of behavior. The BS-MLP is defined as follows:

$$\mathbf{H}^{(l)} = \text{BS-MLP}(\mathbf{G}^{(l)}, \mathbf{b}), \tag{4}$$

where BS-MLP$(\cdot)$ is a wrapper of $|\mathcal{B}|$ standard MLP layers, each corresponds to a specific type of behavior. At each position i, the intermediate representation $\mathbf{G}^{(l)}[i]$ is processed using behavior-specific MLP specified by behavior $\mathbf{b}[i]$.

4.1.3 The Multi-Behavior Transformer Layer. Finally, residual connections [14] and layer normalizations [2] are employed to connect the MB-MSA and BS-MLP modules to get the MB-Trans layer:

$$\begin{aligned}\mathbf{G}^{(l)} &= \text{LayerNorm}\left(\mathbf{H}^{(l-1)} + \text{MB-MSA}(\mathbf{H}^{(l-1)}, \mathbf{b}, \mathbf{P}^{(l)})\right)\\ \mathbf{H}^{(l)} &= \text{LayerNorm}\left(\mathbf{G}^{(l)} + \text{BS-MLP}(\mathbf{G}^{(l)}, \mathbf{b})\right)\end{aligned} \tag{5}$$

4.2 Multi-Behavior Sequential Pattern Modeling

One unique characteristic of user interaction sequences is diverse sequential patterns across multiple behaviors. Take *click* and *purchase* as an example, *click* usually indicates short-term interest, so the recently clicked items may strongly impact the current behavior, while the items clicked long ago usually have little influence on the current behavior. However, items purchased long ago may still strongly influence current interest due to the long-term interest contained *in purchase* behaviors. To encode such diverse multi-behavior sequential patterns, we propose a Multi-Behavior Sequential Pattern Generator (MB-SPG).

4.2.1 The design rationale of MB-SPG. There should be a sequential relationship between any two actions indexed by i, j in a user interaction sequence. Intuitively, this relationship depends on the temporal distance between two actions and their corresponding behavior types $\mathbf{b}[i]$ and $\mathbf{b}[j]$.

Based on this intuition, we design a multi-behavior sequential pattern generator (MB-SPG) to encode such sequential dependency. As shown in Figure 2, the MB-SPG module is coupled with the MB-Trans layer to produce attention bias terms. Specifically, given an interaction sequence, the MB-SPG module generates multi-behavior relative positional embedding matrix $\mathbf{P} \in \mathbb{R}^{h\times n\times n}$ (the same dimension with raw attention scores), where each element $\mathbf{P}[k, i, j]$ measures how much attention position i should pay to position j with respect to the relative distance $(j - i)$, and behavior type $\mathbf{b}[i]$ and $\mathbf{b}[j]$ at a specific head k.

Consequently, the design of MB-SPG boils down to designing $|\mathcal{B}| \times |\mathcal{B}|$ encoding functions $f_{(B_i,B_j)} : \mathcal{R} \rightarrow \mathbb{R}$ from the set $\mathcal{R}$ of relative positions to a real number strength term. Each encoding function $f_{(B_i,B_j)}$ corresponds to a behavior type pair and encodes the temporal relationship of the two types of behaviors. Formally, at head k, for any two actions indexed by i and j,

$$\mathbf{P}[k, i, j] = f_{(\mathbf{b}[i],\mathbf{b}[j])}(j - i), \tag{6}$$

where $(j-i)$ is the distance of j relative to i, and $f_{(\mathbf{b}[i]-\mathbf{b}[j])}(\cdot)$ is the encoding function specified by the behavior type pair $(\mathbf{b}[i], \mathbf{b}[j])$. By doing so, multi-behavior sequential patterns are captured and further used for attention calculation. Next, we will describe the implementation of a single encoding function $f(\cdot)$ in great detail.

4.2.2 Relative Positional Encoding Function. A straightforward implementation of the relative positional encoding function is mapping a relative position to a real number through the trivial embedding lookup table. However, the number of position pairs with the same relative distance is unbalanced. For example, there are 49 pairs {(0,1),(1,2), ... (48,49)} with relative position of 1 for a sequence of length 50, and only 1 pair (0,49) with relative position of 49. Such an unbalanced issue might lead to insufficient learning of long-range sequential patterns.

Similar to T5 [31] in natural language processing, we design a heuristic bucketing mechanism in the relative positional encoding function to alleviate the unbalanced issue. Specifically, f is further expressed as the composition $h \circ b$ of a bucketing function b and an embedding lookup table h. The former maps a relative position to a "bucket" index, and the latter encodes a bucket index as a bias term. $\circ$ indicates the function composition, where $h \circ b(x) = h(b(x))$.

First, we introduce the design of bucketing function b. Suppose the max sequence length is n and the number of buckets is N_B, with indices from 0 to $N_B - 1$, where $n > N_B$. For the input relative position $(j - i)$, the bucketing function $b(\cdot)$ are defined as follows:

$$b(j-i) = \begin{cases} B(j-i) & \text{if } (j-i) \geq 0 \\ B(-(j-i)) + \frac{N_B}{2} & \text{if } (j-i) < 0 \end{cases} \tag{7}$$

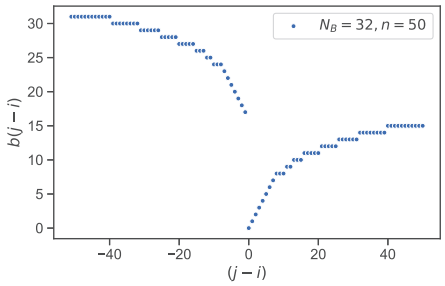


Figure 3: An illustration of the bucketing function with $N_B = 32$ and $n = 50$. The horizontal axis is the relative position $(j - i)$. And the vertical axis is the output value of bucketing function $b(j - i)$.

where half of the buckets are used for positive relative positions, and the other half for negative. $B(x)$ is defined as follows:

$$B(x) = \begin{cases} x & \text{if } 0 \le x < \frac{N_B}{4} \\ \min\left(\frac{N_B}{4} + \left\lfloor \frac{\log 2x - \log \frac{N_B}{2}}{\log 2n - \log \frac{N_B}{2}} \cdot \left(\frac{N_B}{4}\right) \right\rfloor, \frac{N_B}{2} - 1\right) & \text{if } x \ge \frac{N_B}{4} \end{cases}. \tag{8}$$

We plot the bucket function b with $N_B = 32$ and $n = 50$ in Figure 3.

Then, an embedding lookup table h maps the bucket indices $\{0, 1, ..., N_B - 1\}$ to real numbers. Let $\mathbf{E}_b \in \mathbb{R}^{N_B \times h}$ denote the bucket-embedding matrix, where h is the number of heads. $\mathbf{E}_b$ works like a normal embedding layer, which is randomly initialized and continuously updated during training.

4.3 Behavior-Aware Masked Item Prediction

The item-level multi-behavior dependencies and sequential patterns can be modeled with the MB-Trans layer and MB-SPG module acting as the model backbones. Nevertheless, a new challenge arises: how to effectively mine information from users' multi-behavior interaction sequence? We propose a Behavior-Aware Prediction module (BA-Pred) and a behavior-aware masked item prediction objective to tackle this challenge.

4.3.1 Behavior-Aware Prediction Module. Previous works have shown that using multi-behavior supervision signals instead of only target behavior can significantly improve the recommendation performance [9, 11, 40]. Therefore, the multi-behavior recommendation is usually formulated as a multi-task learning problem. However, different types of behavior may be weakly correlated or even conflicted (e.g., *like* and *dislike* in a rating website), which may lead to performance degradation, called *negative transfer*. Therefore, to better capture the behavior-specific information and model the commonalities among different behaviors, we further design a Behavior-Aware Prediction module (BA-Pred).

As shown in Figure 2, the BA-Pred module is a parameter sharing structure like Multi-gated Mixture of Experts (MMoE) [27]. Compared with straightforwardly sharing parameters in MMoE, we separate shared experts and behavior-specific experts explicitly. In this way, BA-Pred can better alleviate conflicting parameter interference between shared and behavior-specific information.

Specifically, when making predictions at position i, the BA-Pred structure can be formulated as:

$$o_i = g_k(\mathbf{H}^{(L)}[i])^\top E_k(\mathbf{H}^{(L)}[i]), \tag{9}$$

where $\mathbf{H}^{(L)}$ is the hidden representation of the last layer of MB-Trans, and $g_k(\cdot)$ is the gating function to calculate gating weights of a specific behavior type $k = \mathbf{b}[i]$ through a linear transformation and a softmax operation:

$$g_k(x) = softmax(\mathbf{W}_g^k x), \tag{10}$$

where $\mathbf{W}_g^k \in \mathbb{R}^{(n_b+n_s)\times d}$ is the gating parameter, n_b and n_s are the number of behavior specific experts and shared experts, respectively. $E_k(\cdot)$ is the combination of behavior-specific experts $e_{k,i}(\cdot)$ and shared experts $e_{s,i}(\cdot)$:

$$E_k(x) = \left[e_{k,1}(x), e_{k,2}(x), ..., e_{k,n_b}(x), e_{s,1}(x), e_{s,2}(x), ..., e_{s,n_s}(x)\right], \tag{11}$$

where each expert is a simple linear layer. Finally, the prediction is made by a dot-product of the final behavior-specific representation o and an item embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}|\times d}$, which is shared with the input item embedding matrix to reduce the model size and alleviate overfitting:

$$\mathbf{p}_i(v) = softmax(o_i \cdot \mathbf{E}^\top), \tag{12}$$

where $\mathbf{p}_i(v) \in \mathbb{R}^{|\mathcal{V}|}$ is the predicted probability distribution over all items $v \in \mathcal{V}$ at position i.

4.3.2 Behavior-Aware Masked Item Prediction Objective. Incorporating the BA-Pred module, we propose a novel training objective to mine the multi-behavior interaction sequences effectively.

Specifically, given the multi-behavior interaction sequence $X = (\mathbf{x}, \mathbf{b})$, we first construct a corrupted item sequence $\hat{\mathbf{x}}$ by randomly setting a portion (ρ) of items in $\mathbf{x}$ to a special mask symbol `[MASK]`, while keeping the behavior sequence $\mathbf{b}$ unmasked. Let the masked items be $\bar{\mathbf{x}}$. Then the behavior-aware masked item prediction objective is to reconstruct $\bar{\mathbf{x}}$ based on $\hat{\mathbf{x}}$ and $\mathbf{b}$:

$$\min_\theta -\log p_\theta(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}, \mathbf{b}) = -\sum_{i=1}^{n} m_i \log p_\theta\left(\mathbf{x}[i] \mid \hat{\mathbf{x}}, \mathbf{b}\right), \tag{13}$$

where $m_i = 1$ indicates $\mathbf{x}[i]$ is masked, θ is the model parameters, and the probability $p_\theta\left(\mathbf{x}[i] \mid \hat{\mathbf{x}}, \mathbf{b}\right)$ is calculated as $\mathbf{p}_i(v = \mathbf{x}[i])$ in Eq. 12. At a particular masked position i, the prediction is made based on the contextual actions around position i and the behavior type performed at position i. By masking only items and keeping behavior types unmasked, the model performs behavior-aware predictions, which better captures diverse user preferences under different types of behaviors.

4.4 Model Learning and Complexity Analysis

4.4.1 Training. MB-STR is trained by optimizing the behavior-aware masked item prediction objective defined in Eq. 13.

4.4.2 Testing. Given a multi-behavior interaction sequence $X = (\mathbf{x}, \mathbf{b})$, we append the special token "`[MASK]`" to the end of item sequence $\mathbf{x}$ and append the target behavior type to the end of the behavior sequence $\mathbf{b}$. Then, we predict the next item under target behavior based on the historical multi-behavior interaction data.

4.4.3 Space Complexity. The learnable parameters in our model are from the item embeddings ($O(|\mathcal{V}|d)$) and parameters in the MB-Trans ($O(|\mathcal{B}|d^2)$), MB-SPG ($O(n)$) and the behavior-aware prediction head ($O(|\mathcal{B}|d^2)$). The total number of parameters is $O(|\mathcal{V}|d + |\mathcal{B}|d^2 + n)$, which is moderate compared to other methods (e.g., $O(|\mathcal{V}|d + d^2 + nd)$ for SASRec and BERT4Rec) since the $|\mathcal{B}|$ and d are relatively small (4 and 16, respectively).

4.4.4 Time Complexity. Since MB-SPG module is composed of hard encoded bucketing functions and embedding lookup tables, and the behavior-aware prediction head consists of only linear experts. Therefore, the computational complexity of our MB-STR is mainly due to the MB-Trans layer. Specifically, MB-MSA and BS-MLP take $O(n^2d + nd^2)$, where the dominant term is typically $O(n^2d)$ from the MB-MSA layer. However, a convenient property in our model is that the computation in each self-attention layer is fully parallelizable, which is amenable to GPU acceleration.

5 EVALUATION

We conduct extensive experiments to answer the following questions:

- **RQ1**: How does MB-STR perform against various recommendation baselines?
- **RQ2**: Do all the designed components of MB-STR bring benefits in line with their design purposes?
- **RQ3**: How do different types of behaviors contribute to the prediction of target behavior?
- **RQ4**: How do different hyper-parameter settings impact the performances of MB-STR?
- **RQ5**: How is the model interpretability of MB-STR? Can the multi-behavior sequential patterns captured by MB-STR be represented in a human-understandable way?

5.1 Experiment Settings

5.1.1 Datasets. We evaluate the proposed MB-STR model on the following three public datasets:

Yelp Data. This is a widely used recommendation dataset collected from Yelp challenge. According to the explicit user rating scores (i.e., ranging from 1 (worst) to 5 (best)), the user interactions are split into three types of user behaviors: *dislike* (rating ≤ 2), *neutral* (rating >2 and <4), and *like* (rating $\geq$4). In addition to the user ratings, there is an additional *tip* behavior representing that a user writes tips on visited venues. Here, we regard *like* as target behavior.

Taobao Data. This is a real-world e-commerce dataset collected from Taobao, one of the largest e-commerce platforms. There are four types of behaviors, i.e., *click, add-to-favorite, add-to-cart and buy*, in which *buy* is regarded as the target behavior.

IJCAI Data. This dataset is released by IJCAI competition for user activity modeling from an online e-commerce platform. Four types of behavior are included, i.e., *click, add-to-favorite, add-to-cart and buy*, in which *buy* is regarded as the target behavior.

For a fair comparison, we closely follow the settings of MB-GMN [40]. We use the same pre-processed datasets with MB-GMN. As the time information of the original used Beibei dataset in [40] is not available, we use the same pre-processed Yelp dataset from MATN [39] as an alternative. Table 2 shows the statistics of the three datasets.

Table 2: Dataset statistics.

Dataset	#users	#items	#interactions	Behavior types
Yelp	19,800	22,734	1.4×10^6	{Tip, Dislike, Neutral, Like}
Taobao	147,894	99,037	7.6×10^6	{Click, Favorite, Cart, Buy}
IJICAI	423,423	874,328	3.6×10^7	{Click, Favorite, Cart, Buy}

5.1.2 Evaluation Settings and Metrics. We evaluate MB-STR and baseline models with two metrics, *i.e.*, the Hit Ratio (*HR@k*) and the Normalized Discounted Cumulative Gain (*NDCG@k*), which have been widely used in top-N recommendation tasks [38–40].

5.1.3 Baseline Models. To comprehensively demonstrate the effectiveness of the proposed MB-STR model, we compare it with various baselines from different lines of research topics: (1) **Single-Behavior Non-Sequential Models**: MF [21], DMF [41], NGCF [38] and LightGCN [15], (2) **Single-Behavior Sequential Models**: SASRec [20] and BERT4Rec [33], (3) **Multi-Behavior Non-Sequential Models**: NGCF$_M$, LightGCN$_M$ (we replace the original single-behavior graph with multi-behavioral graph to enhance NGCF and LightGCN), NMTR [9], MATN [39], MBGCN [19] and MB-GMN [40], (4) **Multi-Behavior Sequential Models**: DIPN [13], SASRec$_M$, BERT4Rec$_M$ (we treat the multi-behavior sequences as single-behavior sequences to enhance SASRec and BERT4Rec.) and DMT [11].

5.1.4 Implementation Details. The MB-STR model is implemented in PyTorch [29]. The data and source code are available [1]. All parameters are initialized with the Gaussian distribution $\mathcal{N}(0, 0.02)$ and optimized using the Adam optimizer with a learning rate of 0.001. We set the training batch size to 128, dropout rate to 0.2. Also, we set max sequence length to $n = 50$ and bucket number in MB-SPG to $N_B = 32$ for all three datasets. For a fair comparison, we set the model dimensionality to 16, which is consistent with [39, 40]. There are 2 MB-Trans layers and 2 heads per layer. The multi-behavior interaction sequence is randomly masked with a portion of $\rho = 0.2$. We run the codes released by the authors for SASRec [2], BERT4Rec [3], LightGCN [4], and DMT [5], and obtain the best results from [39, 40] directly for other models (MF, DMF, NGCF, NMTR, MATN, MBGCN, MB-GMN and DIPN).

5.2 Overall Performance Evaluation (Q1)

We evaluate the performance of predicting the target behavior among all baseline models and our MB-STR model. The results on three datasets are reported in Table 3. From the results, we summarize the following observations.

- **The effectiveness of MB-STR model**. Table 3 shows that MB-STR overperforms all baseline models on three real-world datasets. The average improvement of MB-STR to the best baseline model is 1.15% and 6.67% for Recall and NDCG on the Yelp dataset, 13.78% and 27.73% on Taobao dataset and 7.72% and 12.82% on IJCAI dataset, which demonstrates the effectiveness of MB-STR. Notice that the improvement in Yelp dataset is less

[1] https://github.com/huawei-noah/benchmark/tree/main/FuxiCTR/model_zoo
[2] https://github.com/kang205/SASRec
[3] https://github.com/FeiSun/BERT4Rec
[4] https://github.com/kuandeng/LightGCN
[5] https://github.com/guyulongcs/CIKM2020_DMT

Table 3: Overall model performance comparison with the metrics of HR@10 and NDCG@10. "*" indicates the statistically significant improvements (i.e., p-value < 0.05) over the best baseline (underlined values). The categories of baseline models are shown, where "O" and "M" represent using one single behavior or multi-behavior, respectively, and "S" and "NS" indicate sequential or non-sequential, respectively.

Dataset			Yelp		Taobao		IJCAI	
Metrics			HR	NDCG	HR	NDCG	HR	NDCG
O	NS	MF	0.755	0.481	0.262	0.153	0.285	0.185
		DMF	0.756	0.485	0.305	0.189	0.392	0.250
		NGCF	0.789	0.500	0.302	0.185	0.461	0.292
		LightGCN	0.810	0.513	0.373	0.235	0.443	0.283
	S	SASRec	0.796	0.504	0.372	0.221	0.597	0.406
		BERT4Rec	0.816	0.531	0.385	0.234	0.605	0.431
M	NS	NGCF$_M$	0.793	0.492	0.374	0.221	0.481	0.307
		LightGCN$_M$	0.872	0.585	0.391	0.243	0.486	0.317
		NMTR	0.790	0.478	0.332	0.179	0.481	0.304
		MATN	0.826	0.530	0.354	0.209	0.489	0.309
		MBGCN	0.796	0.502	0.369	0.222	0.463	0.277
		MB-GMN	0.87	0.582	0.491	0.300	0.532	0.345
	S	DIPN	0.791	0.500	0.317	0.178	0.475	0.296
		SASRec$_M$	0.819	0.531	0.637	0.442	0.795	0.611
		BERT4Rec$_M$	0.838	0.558	0.675	0.476	0.816	0.632
		DMT	0.652	0.515	0.666	0.415	0.682	0.513
Our MB-STR			**0.882***	**0.624***	**0.768***	**0.608***	**0.879***	**0.713***
Rela, Improv.			1.15%	6.67%	13.78%	27.73%	7.72%	12.82%

significant than in the Taobao and IJCAI dataset. A possible reason is that the multiple behaviors (dislike, neutral, and like) in Yelp are generated by splitting users' ratings, which are mutually exclusive. Thus the modeling of multi-behavior dependencies is less beneficial for performance improvements.

- **Both sequential and multi-behavioral information bring benefits to model performance**. Despite the big difference in model design between different baselines, there is a trend that considering either multi-behavioral or sequential information leads to performance improvement. For example, BERT4Rec and MB-GMN achieve much better performance than NGCF and LightGCN on all three datasets. However, special designs are required to better utilize these two characteristics, and naive adoption might lead to performance degradation. For example, LightGCN, which only uses single behavior, performs much better than NMTR with multi-behavior and DIPN with sequential information in Yelp and Taobao datasets.
- **MB-STR consistently outperforms multi-behavior sequential baseline models.** There is a significant performance gain when comparing MB-STR with other multi-behavior sequential counterparts. This performance gain can be attributed to the effective modeling of heterogeneous item-level multi-behavior dependencies and diverse multi-behavior sequential patterns. For example, the evaluation results reveal the limitations of the DIPN and DMT, which follow a two-stage aggregation paradigm to aggregate different types of behavioral patterns via weighted summation. These approaches fail to capture the fine-grained dependencies across different types of behaviors. Besides, SASRec$_M$ and BERT4Rec$_M$ utilize vanilla transformer layers and positional encodings, thus cannot differentiate heterogeneous multi-behavior dependencies and diverse multi-behavior sequential patterns.

5.3 Model Ablation Study (Q2)

There are three key components in the MB-STR model, each corresponding to a modeling consideration. To study the validity of each component, we introduce the following MB-STR variants according to their design purpose:

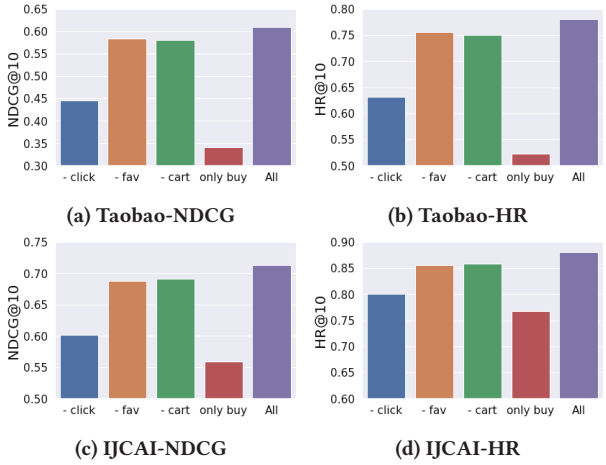
- *MB-STR w/o MB-Trans*: The MB-Trans layer is replaced with a vanilla transformer layer [36]. It treats different behavior transition patterns the same at the transformer layer, thus ignoring multi-behavior heterogeneous dependencies. (Noted as "homogeneous item-level" in multi-behavior modeling in Table 4.)
- *MB-STR w/o MB-SPG*: The MB-SPG module is removed. This model variant is insensitive to sequential information.
- *MB-STR w/o BA-Pred*: The BA-Pred is replaced with a simple linear prediction head as in BERT4Rec [33], which can not make behavior specific predictions.

To further demonstrate the efficacy of each modeling consideration, we compare different variants of MB-STR with representative baselines categorized by their design purpose. The results are shown in Table 4, and we draw the following conclusions:

- **Each of the three key components of the MB-STR brings benefits to performance.** Comparing the prediction performance of MB-STR and its three variants, there is a significant performance degradation when any key components are removed or replaced by other counterparts. Particularly, the performance gap between MB-STR and the *w/o MB-Trans* variant indicates the advantage of MB-Trans in modeling fine-grained heterogeneous dependencies. The MB-SPG module effectively captures diverse multi-behavior sequential patterns, as the *w/o MB-SPG* variant has a performance degradation up to 26.8% in terms of NDCG@10 on Taobao dataset. Besides, the BA-Pred module shows positive effects on all datasets, which demonstrates the rationality of our behavior-aware prediction design.
- **MB-STR and its variants show superior performances compared to baselines with similar design purposes.** Compared to SASRec$_M$ and BERT4Rec$_M$, the better performance of the *w/o MB-Trans & w/o BA-Pred* variant verifies that modeling sequential patterns with incorporating diverse behavior transition patterns is of great importance. We also observe that when sequential information and behavior-specific prediction are both removed, the *w/o MB-SPG & w/o BA-Pred* variant achieves large improvements compared to MATN and MB-GCN. This verifies the significance of modeling multi-behavior dependencies at heterogeneous item-level. And further, with the advantages of multi-behavior modeling and behavior-specific prediction, the *w/o MB-SPG* outperforms NMTR and MB-GMN by a big margin. These comparisons justify the effectiveness and superiority of our proposed MB-Trans layer, MB-SPG module and BA-Pred module for multi-behavior recommendation.

Table 4: Model ablation study. Different variants of MB-STR and representative baselines are categorized according to three design purposes consistent with Table 1. The relative improvement between different variants and the full MB-STR model are highlighted in red. The relative improvement between different baselines and the MB-STR variants with similar design purposes are colored in blue. Best viewed in color.

Model	Model Properties			Yelp		Taobao		IJCAI	
	Multi-Behavior Modeling	Sequential Information	Behavior-Specific Prediction	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
MB-STR	heterogeneous item-level	multi-behavior	✓	**0.882**	**0.624**	**0.768**	**0.608**	**0.879**	**0.713**
w/o MB-Trans	homogeneous item-level	multi-behavior	✓	0.861(-2.4%)	0.596(-4.5%)	0.749(-2.5%)	0.585(-3.8%)	0.869(-1.1%)	0.690(-3.2%)
w/o BA-Pred	heterogeneous item-level	multi-behavior	✗	0.863(-2.2%)	0.599(-4.0%)	0.729(-5.1%)	0.558(-8.2%)	0.859(-2.3%)	0.689(-3.4%)
w/o MB-SPG	heterogeneous item-level	✗	✓	0.868(-1.6%)	0.604(-3.2%)	0.658(-14.3%)	0.445(-26.8%)	0.850(-3.3%)	0.660(-7.4%)
NMTR	behavior-level	✗	✓	0.790(-9.0%)	0.478(-20.9%)	0.332(-49.5%)	0.179(-59.8%)	0.481(-43.4%)	0.304(-53.9%)
MB-GMN	behavior-level	✗	✓	0.870(+0.2%)	0.582(-3.6%)	0.491(-25.4%)	0.300(-32.6%)	0.532(-37.4%)	0.345(-47.7%)
w/o MB-Trans & w/o BA-Pred	homogeneous item-level	multi-behavior	✗	0.851(-3.5%)	0.573(-8.2%)	0.710(-7.6%)	0.542(-10.9%)	0.831(-5.5%)	0.666(-6.6%)
SASRec_M	homogeneous item-level	single-behavior	✗	0.819(-3.8%)	0.531(-7.3%)	0.637(-10.3%)	0.442(-18.5%)	0.795(-4.3%)	0.611(-8.3%)
BERT4Rec_M	homogeneous item-level	single-behavior	✗	0.838(-1.5%)	0.558(-2.6%)	0.675(-4.9%)	0.476(-12.2%)	0.816(-1.8%)	0.632(-5.1%)
w/o MB-SPG & w/o BA-Pred	heterogeneous item-level	✗	✗	0.862(-2.3%)	0.593(-5.0%)	0.640(-16.7%)	0.412(-32.2%)	0.840(-4.4%)	0.654(-8.3%)
MATN	behavior-level	✗	✗	0.826(-4.2%)	0.530(-10.6%)	0.354(-44.7%)	0.209(-44.7%)	0.489(-41.8%)	0.309(-52.8%)
MB-GCN	behavior-level	✗	✗	0.796(-7.7%)	0.502(-15.3%)	0.369(-42.3%)	0.222(-46.1%)	0.463(-44.9%)	0.277(-57.6%)



(a) Taobao-NDCG (b) Taobao-HR

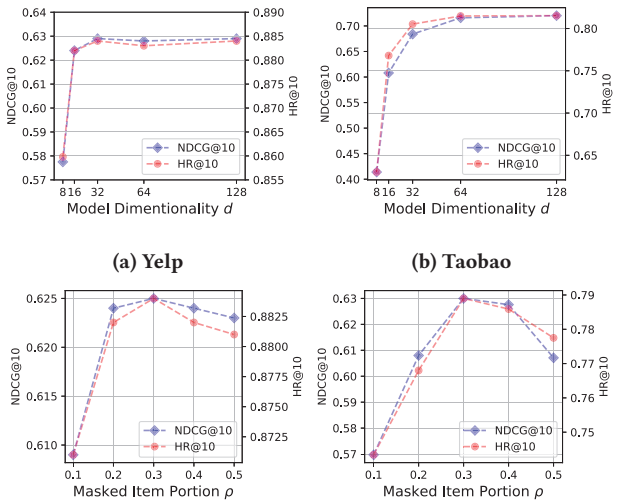
(c) IJCAI-NDCG (d) IJCAI-HR

Figure 4: Effect of auxiliary behaviors.

5.4 Effect of Auxiliary Behaviors (Q3)

We further perform a data ablation study to investigate the effectiveness of incorporating different auxiliary behaviors. We run four variants of the MB-STR on two real-world e-commerce dataset, where "-click", "-fav", and "-cart" indicates the MB-STR without incorporating the *click*, *add-to-favorite* and *add-to-cart* behavior. Also, "only buy" represents the variant that only uses the target behavior. The evaluation results are shown in Figure 4.

The results show that MB-STR with all four behaviors achieves the best performance compared with the ablation variants, emphasizing the necessity of integrating more auxiliary behaviors to help the recommendation under target behavior. Moreover, it demonstrates that MB-STR can effectively learn user intentions from multiple types of user behaviors.



(a) Yelp (b) Taobao

(c) Yelp (d) Taobao

Figure 5: Hyper-parameter study of the MB-STR.

5.5 Hyper-parameter Analysis (Q4)

To investigate the effect of different hyper-parameter settings on MB-STR, we perform experiments on Yelp and Taobao datasets with different configurations of key hyper-parameters, namely, the model dimensionality d, and the masked item portion ρ. As shown in Figure 5, we conclude as follows:

- **Model dimensionality d.** Figure 5a and Figure 5b show the performance when model hidden dimension d grows from 8 to 128. When d increases from 8 to 32, the performance grows on both Yelp and Taobao datasets, as higher hidden dimensionality leads to larger representation capacity. However, with the further increase of the model dimensionality from 32 to 64, the performance saturates on Yelp yet keeps improving on Taobao. When

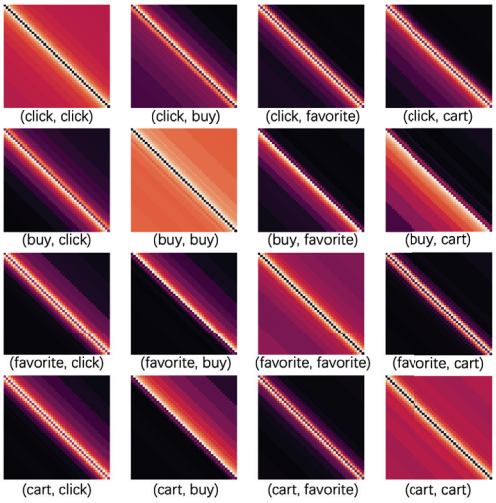


Figure 6: Heat-maps of multi-behavior sequential pattern learned on the Taobao dataset, where x-axis and y-axis denote different positions at the behavior sequence. The colors represent the attention weights, where lighter color implies higher level of importance.

the model dimensionality reaches a threshold, d will no longer be the performance bottleneck. Increasing the model dimension will not improve performance but increase computation overhead. Furthermore, the threshold depends on the characteristics of datasets. For example, the item set $\mathcal{V}$ of Taobao is 4 times larger than that of Yelp, so a higher hidden dimension is required for Taobao to achieve better performance.

- **Masked item portion ρ.** The masked item portion ρ controls the proportion of items used as prediction targets. Figure 5c and Figure 5d show how ρ affects the performances of MB-STR on Yelp and Taobao, respectively. We can observe that the performance increases first and then decreases. When ρ is too small, the training signals will be insufficient as only a small portion of items are used as training targets; however, when ρ is too large, the behavior sequence is corrupted by introducing too many artificial symbols `[MASK]`. Both circumstances will degrade the performance of MB-STR.

5.6 Interpretability Analysis (Q5)

To further investigate the interpretability of MB-STR, we conduct analysis via visualizing the multi-behavior sequential patterns learned in MB-SPG (Section 4.2). Heat-maps of multi-behavior sequential patterns learned on the Taobao dataset are drawn in Figure 6. Specifically, each heat-map matrix in Figure 6 visualizes an inter-behavior relative positional encoding function $f_{(B_i,B_j)}(j-i)$ as formulated in Equation 6, with i, j in x-axis and y-axis representing different positions at the behavior sequence. Taking the heat-map M indexed by *(buy, cart)* as an example, each element $M[i,j]$ represents the attention bias score from the *buy* behavior of the location i to the *cart* behavior of the location j.

From Figure 6, meaningful patterns emerged, which are in line with our intuition. For example, the *buy* behavior pays long-range attention to other items interacted by *buy* (the heat-map at row 2, column 2), indicating that *buy* represents long-term user interest. Besides, *buy* actions pay more attention to the previous *cart* actions (the heat-map at row 2, column 4), which reflects the high propensity that *cart* can convert to *buy*. Correspondingly, *cart* actions pay more attention to the latter *buy* actions (the heat-map at row 2, column 4) and we can refer to this mechanism as "looking to the future". Note that such a mechanism can not be employed during the evaluation stage since "future" data are unavailable. Nevertheless, during the training stage, "looking to the future" plays an essential role in mining the sequential patterns and learning more comprehensive representations.

6 CONCLUSION AND FUTURE WORK

In this paper, we proposed the Multi-Behavior Sequential Transformer Recommender (MB-STR) framework for addressing real-world recommendation problems. MB-STR gains performance improvement from these advantages: (1) it captures heterogeneous fine-grained item-level multi-behavior dependencies through Multi-Behavior Transformer Layer (MB-Trans); (2) it models diverse multi-behavior sequential patterns via Multi-Behavior Sequential Pattern Generator (MB-SPG); (3) it is able to mine the multi-behavior interaction sequence effectively by the Behavior-Aware masked item Prediction module (BA-Pred). Comprehensive experiments on three real-world recommendation datasets demonstrate the effectiveness of the MB-STR model. In future works, we will further explore the integration of side information into MB-STR and model pre-training for cold-start problems.

ACKNOWLEDGMENTS

We thank MindSpore [1] for the partial support of this work, which is a new deep learning computing framework.

REFERENCES

[1] 2020. *MindSpore.* https://www.mindspore.cn
[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).
[3] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477* (2020).
[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proc. Workshop Deep Learning for Recommender Systems.*
[5] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems.* 293–296.
[6] Yashar Deldjoo, Mehdi Elahi, Paolo Cremonesi, Franca Garzotto, Pietro Piazzolla, and Massimo Quadrana. 2016. Content-based video recommendation system based on stylistic visual features. *Journal on Data Semantics* 5, 2 (2016), 99–113.
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).
[9] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2019. Neural multi-task recommendation from multi-behavior data. In *2019 IEEE 35th international conference on data engineering (ICDE).* IEEE, 1554–1557.
[10] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on*

artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 315–323.

[11] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Lixin Zou, Yiding Liu, and Dawei Yin. 2020. Deep Multifaceted Transformers for Multi-objective Ranking in Large-Scale E-commerce Recommender Systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2493–2500.

[12] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *IJCAI*.

[13] Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, and Bin Cui. 2019. Buying or browsing?: Predicting real-time purchasing intent using attention-based deep network with multiple behavior. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1984–1992.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[15] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[16] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 843–852.

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

[18] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.

[19] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-behavior recommendation with graph convolutional networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 659–668.

[20] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012), 1097–1105.

[23] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-interest network with dynamic routing for recommendation at Tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2615–2623.

[24] Jiacheng Li, Yujie Wang, and Julian J. McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of WSDM*, James Caverlee, Xia (Ben) Hu, Mounia Lalmas, and Wei Wang (Eds.). ACM, Houston, TX, USA, 322–330.

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030* (2021).

[26] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1930–1939.

[27] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1930–1939.

[28] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model.. In *Interspeech*, Vol. 2. Makuhari, 1045–1048.

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).

[30] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. (2018).

[31] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[32] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862* (2019).

[33] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[34] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 565–573.

[35] Aäron Van Den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Neural Information Processing Systems Conference (NIPS 2013)*, Vol. 26. Neural Information Processing Systems Foundation (NIPS).

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in NeurIPS 2017*. 5998–6008.

[37] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 839–848.

[38] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[39] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Bo Zhang, and Liefeng Bo. 2020. Multiplex behavioral relation learning for recommendation via memory augmented transformer network. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2397–2406.

[40] Lianghao Xia, Yong Xu, Chao Huang, Peng Dai, and Liefeng Bo. 2021. Graph meta network for multi-behavior recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 757–766.

[41] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems.. In *IJCAI*, Vol. 17. Melbourne, Australia, 3203–3209.

[42] 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 *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[43] 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.