Data Science and Engineering
https://doi.org/10.1007/s41019-023-00238-3

RESEARCH PAPER

Explicit Behavior Interaction with Heterogeneous Graph for Multi-behavior Recommendation

Zhongping Zhang[1,2,3] · Yin Jia[1] · Yuehan Hou[1] · Xinlu Yu[4]

Received: 28 July 2023 / Revised: 26 October 2023 / Accepted: 11 December 2023

Abstract
Multi-behavior recommendation systems exploit multi-type user–item interactions (e.g., clicking, adding to cart and collecting) as auxiliary behaviors for user modeling, which can alleviate the problem of data sparsity faced by traditional recommendation systems. The key point of multi-behavior recommendation systems is to make full use of the auxiliary behavior information for the learning of user preferences. However, there are two challenges in existing methods that need to be explored: (1) capturing personalized user preferences based on multiple auxiliary behaviors, especially for negative feedback signals; and (2) explicitly modeling the semantics between auxiliary and target behaviors, and learning the explicit interactions between multiple behaviors. To tackle the two problems described above, we propose a novel model, called explicit behavior interaction with heterogeneous graph for multi-behavior recommendation (MB-EBIH). In particular, we first construct a heterogeneous behavior graph, including both positive and negative behaviors. A pre-trained model based on graph neural network (GNN) is then used to generate explicit behavior interaction values as the edge weights for the heterogeneous behavior graph. These weights reflect the importance of each of the auxiliary behaviors in an explicit manner. Finally, the extracted explicit behavior interaction information is incorporated into the multi-behavior user–item bipartite graphs to learn better representations. Experimental results on four real-world datasets demonstrate the effectiveness of our model in terms of exploring multi-behavioral data; and ablation and analysis experiments further demonstrate the effectiveness of explicit behavior interaction information.

Keywords Multi-behavior recommendation · Collaborative filtering · Heterogeneous graph · Explicit behavior interaction

1 Introduction

Recommender systems (RSs) are designed to capture personalized user preferences and to suggest high-quality items, and this has become an emerging research topic with many online applications [1–13]. The main idea of an RS is to learn the user's personalized interests from historical user–item interactions, such as feedback from the user on a movie and a purchase of shoes. However, traditional RS methods only consider a single type of user behavior in regard to an item, meaning that they fail to model the comprehensive preferences of the user and suffer from the problem of data sparsity. In the real world, given an item, users often show multiple types of behaviors, and the relationships between these different behaviors can reflect the user's preferences from multiple perspectives. For example, in regard to short video platforms, the user may show various behaviors based on the watch time, likes, follows, comments and forwards. When considered together, these show

✉ Yin Jia
yjia_ysu@163.com

Zhongping Zhang
zpzhang@ysu.edu.cn

Yuehan Hou
202222040120@stumail.ysu.edu.cn

Xinlu Yu
3223705322@qq.com

[1] School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China

[2] The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao 066004, China

[3] The Key Laboratory of Software Engineering of Hebei Province, Qinhuangdao 066004, China

[4] School of International Business, Henan University, Kaifeng 475001, China

Published online: 19 January 2024

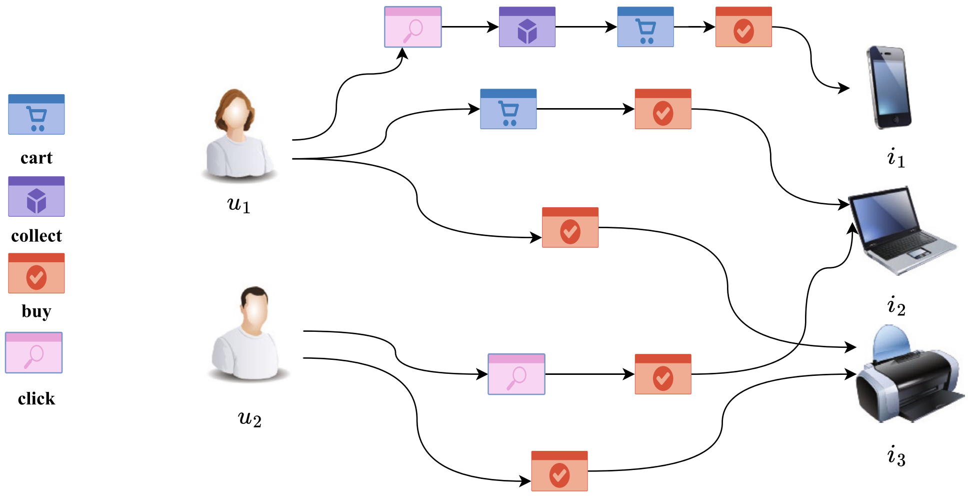


Fig. 1 Illustration of negative feedback signals for auxiliary behaviors: u_1 and u_2 can purchase directly without auxiliary behaviors (i.e., auxiliary behaviors convey negative feedback signals)

complex relationships, and comprehensive user modeling is difficult if any parts of these behaviors are missing.

Some works have therefore attempted to make full use of the auxiliary behaviors and to capture the dependence relationships between them. For example, models such as MBGCN [14], CRGCN [15] and MB-CGCN [16] explore the dependencies between behaviors by constructing user–item bipartite graphs with positive feedback on auxiliary behaviors. NMTR [17] is a cascading deep model based on NCF [4], which was developed to investigate the dependencies between behaviors. MATN [18] utilizes an attention mechanism and a transformer network to encode the relationships between multiple behaviors. CIGF [19] leverages matrix multiplication to model the relationships between multiple behaviors, and a multitask learning network is used to optimize the model. GNMR [20] exploits the dependencies among multiple behaviors via recursive embedding propagation.

Despite their effectiveness, we believe that existing works suffer from two limitations:

- *They fail to capture the personalized preferences of the user in terms of behaviors, especially negative feedback signals.* In a real-world e-commerce platform, before a user buys an item (target behavior), they may show multiple auxiliary behaviors, such as *clicking*, *adding to cart* and *collecting*. These auxiliary behaviors have complex relationships with the target behaviors. In addition, the interaction relationships between behaviors are highly customized for different users and reflect the personalized tastes of the user. For example, when shopping on an e-commerce platform, some people may add preferred items to the cart before buying them, while others may buy them directly. We therefore need to carefully model these personalized preferences based on the interaction between behaviors. The negative signals associated with auxiliary behaviors (i.e., the user does not perform the auxiliary behavior) are also useful and have often been ignored in previous methods. An example of personalized behavior preferences and negative feedback is shown in Fig. 1.
- *Explicit interactions between auxiliary and target behaviors are not fully explored.* Since different users have various behavior preferences, we need to fully explore the explicit relationships between multiple behaviors. To learn the user's preferences, the explicit contributions of different auxiliary behaviors to the target behavior should be explored. However, existing methods [14, 18, 21] typically consider user–item bipartite graphs for each behavior separately, an approach that fails to jointly model the explicit contributions of these auxiliary behaviors. In [16] and [15], behavior chains are adopted to explore the dependencies of behaviors, but this approach cannot handle complex sequences of user behavior. Nevertheless, the explicit behavior interactions (i.e., statistical explicit semantic information) directly reflect the probability that a user would show the target behavior after showing the auxiliary behavior. Thus, explicit statistical behavior information is of vital importance for user modeling and has potential research value for multi-behavior RSs.

In order to address the two issues described above, we study explicit behavior interactions in multi-behavior data. Inspired by the cross-feature approach used in traditional recommendation tasks [19, 22, 23], we propose a model called explicit behavior interaction with heterogeneous graph for multi-behavior recommendation (MB-EBIH). This model mainly consists of two modules: an explicit behavior interaction information extraction module and a fusion module. In first of these modules, we construct a heterogeneous behavior informative graph that includes both positive and negative behaviors based on multi-behavior historical data.

Here, the nodes represent the user, the item, and the negative and positive signals of auxiliary behaviors, and each edge represents an interaction between a pair of nodes. We investigate information on the behavior interactions based on this graph structure. We then design a self-supervised task to obtain a GNN-based pre-trained knowledge model, which is used to generate explicit behavior interaction values as the edge weights for the heterogeneous behavior graph, where the weights explicitly represent the importance of the auxiliary behaviors. In the second module, the extracted explicit behavior interaction information is incorporated into multi-behavior user–item bipartite graphs to learn better representations.

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

- We propose a new model called MB-EBIH for multi-behavior recommendation, which can capture personalized user preferences with both positive and negative feedback from auxiliary behaviors. Moreover, it explicitly models the relations between the auxiliary and target behaviors and learns the explicit interactions between multiple behaviors.
- To the best of our knowledge, we are the first to model the explicit behavior interactions in multi-behavior RSs and to explore the relationships between different behaviors to perform better personalized user modeling.
- We conduct comprehensive experiments on four real-world datasets to evaluate the effectiveness of MB-EBIH and the generalization of the explicit behavior interactions. The results show that our model has significantly improved recommendation performance compared to other baseline models, and demonstrate the effectiveness of capturing the explicit behavior interactions in multi-behavior RSs.

2 Related Work

Existing multi-behavior recommendation models can be classified into three categories: the first category that based on matrix factorization (MF) [24–26] directly extended the MF technique from a single-behavior to multi-behavior RSs. For example, [25] extended the MF model to simultaneously factorize multiple matrices, while sharing embeddings on the item side. References [24, 26] extended this model to perform matrix factorization of multiple behaviors by sharing user or item embeddings.

The second that based on deep neural network (DNN) involves initially learning user and item embeddings separately from each behavior using the designed network. Subsequently, these embeddings, acquired from various behaviors, are aggregated to predict the target behavior. For example, NMTR [17] uses a cascading deep model based on NCF [4] to investigate the dependencies between behaviors and uses a multitask framework for optimization. MATN [18] and DIPN [27] both utilize an attention mechanism to capture the relationships between multiple behaviors, while MATN also adopts a transformer network to encode the relationships between multiple behaviors.

The third that based on GNN construct user–item bipartite graphs with positive feedback on auxiliary behaviors and then use graph convolution network (GCN) for embedding learning. For example, MBGCN [14] uses positive feedback on auxiliary behaviors to construct user–item bipartite graphs and item–item graphs to explore the behavioral dependencies and behavioral semantics, respectively. CRGCN [15] and MB-CGCN [16] adopt a cascading GCN structure to investigate the dependencies between multiple behaviors. CRGCN delivers the embeddings of users and items learned from the positive signals of previous behavior to the next behavior in a chain, based on the residual structure, and applies multitask learning to optimize the model. MB-CGCN inherits the cascading structure of CRGCN, but replaces the residual structure with feature transformation operations. GNMR [20] exploits the dependencies among multiple behaviors via recursive embedding propagation. MGNN [28] adopts a multiple-layer network to simultaneously learn shared user and item embeddings, as well as distinct embeddings for each behavior. MB-GMN [29] employs a meta-graph neural network to effectively model diverse multi-behavior patterns, capturing the heterogeneity and diversity of behaviors within a unified graph. MMCLR [30] employs three contrastive learning tasks for training the model parameters, considering similar preferences among behaviors for specific users or items. Both KMCLR [31] and KHGT [21] leverage knowledge graphs to augment information, mitigating the issue of sparse target behavioral data. Furthermore, KMCLR integrates contrastive learning techniques. HMGGR [32] employs graph contrastive learning among the constructed hyper-meta-graphs to adaptively learn complex dependencies among different behaviors for embedding learning. MBA [33] utilizes multiple types of implicit user behavior data and data denoising techniques to enhance the prediction of target behaviors. EHCF [34], GHCF [35] and FPD [36] all adopt the non-sampling training strategy, with FPD additionally utilizing a multilayer perception to extract distinctions among various behaviors.

However, all of the aforementioned approaches fail to model the explicit interactions between behaviors, leading to an inability to accurately model explicit semantic relationships between target and auxiliary behaviors. Moreover, these approaches neglect the negative feedback signals from auxiliary behaviors, thereby inadequately capturing the user's personalized preferences comprehensively.

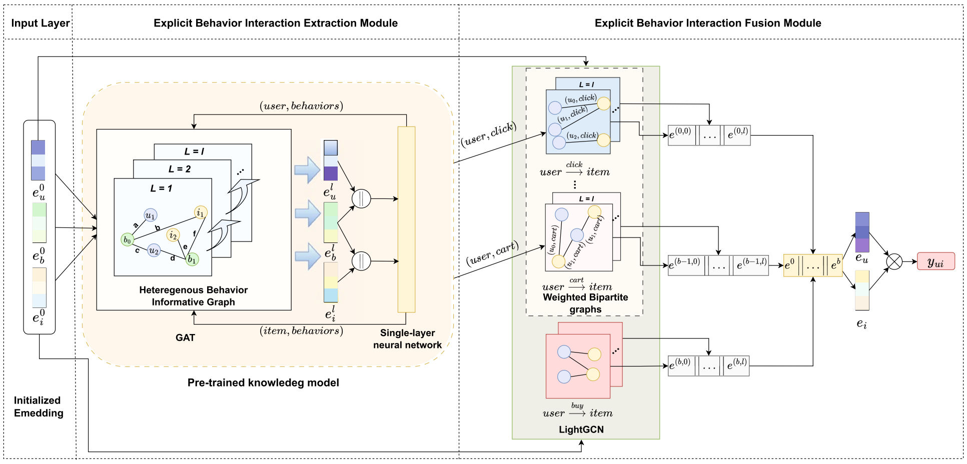

Input Layer
Explicit Behavior Interaction Extraction Module
Explicit Behavior Interaction Fusion Module
e_u^0
e_b^0
e_i^0
Initialized Emedding
$(user, behaviors)$
L = l
L = 2
L = 1
u_1
i_1
i_2
b_0
u_2
b_1
a
b
c
d
e
f
Heteregenous Behavior Informative Graph
GAT
e_u^l
e_b^l
e_i^l
$(item, behaviors)$
Single-layer neural network
Pre-trained knowledeg model
$(user, click)$
$(user, cart)$
$(u_0, click)$
$(u_1, click)$
$(u_2, click)$
$user \xrightarrow{click} item$
$(u_0, cart)$
$(u_1, cart)$
$user \xrightarrow{cart} item$
Weighted Bipartite graphs
$user \xrightarrow{buy} item$
LightGCN
$e^{(0,0)} \| \dots \| e^{(0,l)}$
$e^{(b-1,0)} \| \dots \| e^{(b-1,l)}$
$e^{(b,0)} \| \dots \| e^{(b,l)}$
$e^0 \| \dots \| e^b$
e_u
e_i
y_{ui}

Fig. 2 Overview of MB-EBIH model. *(user, behavior) and (item, behavior)* denote the explicit behavior interactions. $\|$ is the concatenate operation. $\otimes$ is the inner product operation

Unlike the works mentioned above, our model can more accurately model relationships between behaviors by constructing a heterogeneous behavioral graph to effectively explore the explicit interaction semantics between behaviors in a more flexible manner. The heterogeneous graph we construct also includes negative feedback signals for the auxiliary behaviors, enabling a more comprehensive modeling of the user's personalized preferences across different behaviors.

3 Preliminaries

3.1 Problem Formulation

Traditional RSs typically model users' preferences based on single behavior (called the target behavior); however, the users in the real world usually show various behaviors. For example, in the e-commerce platforms, users often interact with the provided items via multiple behaviors before showing the target behavior (such as *clicking*, *adding to cart* or *collecting*), and these behaviors can reflect their preferences in different respects. In this work, we address the issues mentioned in the Introduction with the goal of designing a recommendation model that can better utilize auxiliary behaviors, in order to improve the performance of the model for target behavior prediction.

3.2 Task Formulation

In this section, we give a formal definition of our model for the multi-behavior recommendation task.

We assume a set of multi-behavior data $\{\mathcal{U}, \mathcal{I}, \mathcal{B}\}$, where $\mathcal{U} = \left\{u_1, u_2, \dots, u_M\right\}$ and $\mathcal{I} = \left\{i_1, i_2, \dots, i_N\right\}$ denote the sets of users and items, respectively, and $\mathcal{B} = \left\{b_1, b_2, \dots, b_K\right\}$ denote the possible types of behavior (where M, N, K are the number of users, items and behavior types, respectively). We denote b_K as the target behavior and the rest as the auxiliary behaviors.

Multi-Behavior User–Item Bipartite Graph. We use the user–item interaction matrices for each behavior to represent user–item bipartite graphs. Given behavior types $\mathcal{B} = \left\{b_1, b_2, \dots, b_K\right\}$, we specify that $\left\{Y_{b_1}, Y_{b_2}, \dots, Y_{b_k}\right\}$ represent the matrices of behaviors, where Y_{b_k} is the matrix for the target behavior. Each entry in the matrices is an integer, which is defined by:

$$y_{u,i}^b = \begin{cases} 1, & \text{If } u \text{ has interacted with } i \text{ via behavior } b \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Negative Feedback Signals. In real-world scenarios, when a user performs a certain behavior, we define it as a positive feedback signal $behavior_1$ for the corresponding behavior and conversely as a negative feedback signal $behavior_0$. Combining the negative feedback signals of user behaviors can reflect the user's preferences more comprehensively and thus model the users more accurately [37, 38].

Formally, we can express the task of multi-behavior recommendation as follows:

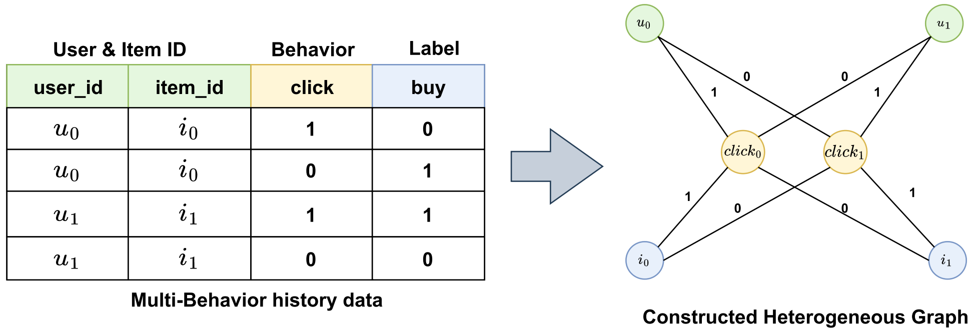
Input: The user–item interaction data of $\mathcal{B}$ types of behaviors, $\{\mathcal{U}, \mathcal{I}, \mathcal{B}\}$.

Output: A recommendation model that estimates the probability that a user u will interact with an item i via the *kth* behavior (i.e., the target behavior)

4 Method

4.1 Overview

In this section, we provide a detailed description of the MB-EBIH model. However, before delving into the specifics, we offer an overview of the model from a global perspective. Figure 2 shows the overall structure of the model, which has



User & Item ID		Behavior	Label
user_id	item_id	click	buy
u_0	i_0	1	0
u_0	i_0	0	1
u_1	i_1	1	1
u_1	i_1	0	0

Multi-Behavior history data

Fig. 3 Process of constructing heterogeneous graph, *click*$_0$ and *click*$_1$ denotes the negative and the positive feedback signals of *click*

several important components: (1) *the shared embedding module*, which initializes the embeddings of user, item and behavior nodes in both the heterogeneous information graph and the multi-behavior bipartite graphs; (2) *the explicit behavior interaction extraction module*, in which a heterogeneous behavior informative graph is first constructed, then a GAT-based self-supervised task is designed to extract explicit behavior interaction information from the graph and train a pre-trained knowledge model capable of inferring explicit behavior interaction information; and (3) *the explicit behavior interaction fusion module*, which incorporates explicit behavior interaction information into the embedding learning of users and items. Toward the conclusion of this section, we analyze the complexity of the MB-EBIH model and discuss the advantages of explicitly modeling behavioral interactions.

4.2 Shared Embedding Module

Given user u and i, we transform them into learnable embeddings $\boldsymbol{e_u}^{(\mathbf{0})} \in \mathbb{R}^{d_0}$ and $\boldsymbol{e_i}^{(\mathbf{0})} \in \mathbb{R}^{d_0}$, where d_0 denotes the embedding size, to initialize a user and an item.

For each of auxiliary behavior nodes in the heterogeneous behavior graph, we initialize the negative and positive feedback *behavior*$_0$ and *behavior*$_1$, respectively. In a similar way to the definition of *user (item) embedding*, we define the initial embedding of each auxiliary behavior b as $\left\{\boldsymbol{e}_{\boldsymbol{b_0}}^{(\mathbf{0})}, \boldsymbol{e}_{\boldsymbol{b_1}}^{(\mathbf{0})}\right\} \in \mathbb{R}^{d_1}$, where d_1, b_0 and b_1 denote the embedding size of the auxiliary behaviors, and the negative and positive signals of auxiliary behavior b, respectively.

4.3 Explicit Behavior Interaction Extraction Module

In this section, we introduce the core component of our model, the explicit behavior interaction extraction module, which carries out the following main steps:

4.3.1 Heterogeneous Behavior Informative Graph

In this stage, we aim to transform the multi-behavior historical data into a heterogeneous behavior informative graph, which contains abundant explicit behavior interaction information. More specifically, we construct an undirected heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ consists of the user nodes $u \in \mathcal{U}$, item nodes $i \in \mathcal{I}$ and auxiliary behavior nodes $b \in \mathcal{B}$, denoting the set of all types of nodes and $\mathcal{E}$ denotes the set of edges referring to the behavior interactions.

Unlike previous works [14], the edges of the unified heterogeneous behavior graph constructed here are assigned attributes, the values of which the attributes represent the semantics of the explicit behavior interaction.

Heterogeneous Graph Construction. As shown in Fig. 3, we first choose one behavior from the set of user behaviors $\mathcal{B}$ as the target behavior (i.e., the **Label** in Fig. 3); here, we assume that the target behavior is *buying*. Inspired by prior works on explicit cross-features [6, 22, 23], we calculate the weights of the edges as follows:

$$w_{(f_0,f_1)} = \frac{sum\left(f_0,f_1\right) \mid (buy = 1)}{sum\left(f_0,f_1\right) \mid (buy = 1 \text{ and } buy = 0)} \tag{2}$$

where f_0 and f_1 denote two nodes in the constructed heterogeneous graph, and the value of $sum\left(f_0,f_1\right)$ represents the total number of co-occurrences of f_0 and f_1 in the multi-behavior history data under certain conditions.

For brevity, Fig. 3 shows only one behavior *click* (i.e., auxiliary behavior) in our example here, and two or more auxiliary behaviors can be calculated in the same way.

4.3.2 Details of the Explicit Behavior Interaction Extraction Module

In Sect. 4.3.1, we described the transformation of multi-behavior data into a unified weighted heterogeneous informative graph. However, we argue that the *explicit behavior interaction* calculated by Eq. 2 can represent only the explicit dependencies between the user, the positive or negative feedback from a single auxiliary behavior and the target behavior. It may not capture the full extent of complex relationships that can exist between multiple auxiliary behaviors and the user's preferences. To capture information about higher-order neighbor nodes and effectively model explicit interactions between multiple behaviors, we choose GAT [39] to capture the feature information of neighboring nodes, based on the attention mechanism, to better distinguish between the contributions of different behaviors. For the weighted heterogeneous informative graph, we capture information about the of neighboring nodes in two steps, as described below:

Calculation of the attention Coefficient. Since it is necessary to determine the importance of the feature information for each neighbor node in the heterogeneous graph, we define a scalar to measure it, which is known as the attention coefficient. In order to obtain the attention coefficient for a certain node i and its neighbor node j, it is first necessary to calculate the similarity coefficient *sim* between the two nodes i and j:

$$sim_{i,j} = MLP([W\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{l})} \parallel W\boldsymbol{e}_{\boldsymbol{j}}^{(\boldsymbol{l})}]), j \in N_i \tag{3}$$

where $\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{l})}$ and $\boldsymbol{e}_{\boldsymbol{j}}^{(\boldsymbol{l})}$ denote the embeddings for node i and its neighbor node j in layer L, respectively; $\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{0})}$ and $\boldsymbol{e}_{\boldsymbol{j}}^{(\boldsymbol{0})}$ are embeddings initialized in the share embedding module; N_i and W are the set of neighbors of node i and the feature transformation matrix shared by nodes i and j, respectively; and ($\parallel$) represents the vector concatenation operation. *MLP* is a single-layer feed-forward neural network, which serves to map the embedding of the concatenated high-dimensional nodes to a scalar (i.e., the similarity coefficient *sim* between the nodes).

After obtaining $sim_{i,j}$ between the target node i and the neighbor node j, we use the softmax function to normalize $sim_{i,j}$ and then calculate the attention coefficient $a_{i,j}$. Formally, this process can be generally expressed as:

$$\alpha_{i,j} = \frac{\exp\left(LeakyReLU\left(sim_{i,j}\right)\right)}{\sum_{k\in N_i}\exp\left(LeakyReLU\left(sim_{i,k}\right)\right)} \tag{4}$$

where **LeakyReLU(*)** is nonlinear activation function formulated as follows (where the default value of α is 0.2):

$$LeakReLU(z) = \begin{cases} z & z > 0 \\ \alpha z & z <= 0, \alpha = 0.2 \end{cases} \tag{5}$$

Heterogeneous Graph Convolution. After calculating the similarity coefficients between the target node and each neighbor node, we use these coefficients to carry out a weighted summation operation on the feature information of the neighbors. This is adopted as the embedding of the target node in layer L, as follows:

$$\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{l})} = \sigma\left(\sum_{j\in N_i}\alpha_{i,j}W\boldsymbol{e}_{\boldsymbol{j}}^{(\boldsymbol{l-1})}\right) \tag{6}$$

where $\sigma(\cdot)$ denotes the nonlinear activation function. In practice, GAT is usually used with a multi-head attention mechanism to enhance the effectiveness of the attention mechanism, as follows:

$$\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{L})}(\boldsymbol{K}) = \big\Vert_{k=1}^{K}\,\sigma\left(\sum_{j\in N_i}\alpha_{i,j}^{k}W^{k}\boldsymbol{e}_{\boldsymbol{j}}^{(\boldsymbol{L-1})}\right) \tag{7}$$

where $\Vert$ is the vector concatenation operation, K is the number of heads and $\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{L})}(\boldsymbol{K})$ denotes the embedding of node i under head K in the last layer. Since this is not the focus of our work, we set $K = 1$ in this study.

4.3.3 Extracting Explicit Behavior Interaction Information

After extracting the information on the neighbors of each node using GAT, we need to reconstitute the explicit interactions between nodes. Here, we use the embedding of the nodes in the last layer of the *explicit behavior interaction extraction module* to get the updated explicit behavior interaction information between nodes (i.e., the value of the edge between nodes f_0 and f_1), using an explicit behavior interaction inferring function. Formally, the function is expressed as:

$$p_{u,b_k} = \sigma\left(f\left(\boldsymbol{e}_{\boldsymbol{u}}^{\boldsymbol{L}}, \boldsymbol{e}_{\boldsymbol{b_k}}^{\boldsymbol{L}}\right)\right) \tag{8}$$

$$p_{i,b_k} = \sigma\left(f\left(\boldsymbol{e}_{\boldsymbol{i}}^{\boldsymbol{L}}, \boldsymbol{e}_{\boldsymbol{b_k}}^{\boldsymbol{L}}\right)\right) \tag{9}$$

where p_{u,b_k} and p_{i,b_k} are the inferred value of the edge between the user node u and the node of *kth* behavior b_k, and the item node i and the node of *kth* behavior b_k, respectively. We set $f(\cdot)$ to a MLP layer in our model. Here, $\sigma(\cdot)$ is the *sigmoid* function which maps the value to the range of $(0, 1)$.

In order to enable the model to infer the explicit relationship between any two nodes, we design a self-supervised learning task to train our pre-trained knowledge model using $w_{(f_0,f_1)}$ (obtained in Sect. 4.3.1) as the label and $p_{(f_0,f_1)}$ (i.e., p_{u,b_k} and p_{i,b_k}) as the predicted value. Since $w_{(f_0,f_1)}$ contains explicit behavior interaction information between two nodes, when it is used as the label of training task, the final predicted value $p_{(f_0,f_1)}$ can better represent the explicit interaction semantics between the nodes through the guidance of $w_{(f_0,f_1)}$.

In a similar way to traditional self-supervised learning, we can train the model directly using a square loss $\left\Vert p_{(f_0,f_1)} - w_{(f_0,f_1)}\right\Vert^2$. However, in the real world, the amount of positive and negative feedback for different behaviors varies; for example, *clicking* may have more positive feedback than *adding to carting*, and the positive feedback from *adding to cart* will usually be less than its own negative feedback. In order to enable the model to better distinguish the importance of each behavior, we allocate a weight to the square loss of each explicit behavior interaction, which is defined as the total number of occurrences of each pair of explicit behavior interactions $\left(f_0,f_1\right)$, i.e., $sum(f_0,f_1)$. We then obtain the following loss function:

$$Loss_1 = \sum_{n=1}^{N_{edge}} \ln\left(sum\left(f_0,f_1\right) + \beta\right)\left\| p_{\left(f_0,f_1\right)} - w_{\left(f_0,f_1\right)}\right\|^2 \tag{10}$$

where N_{edge} is the total number of the edges in the constructed heterogeneous graph, $sum(f_0,f_1)$ denotes the total number of times the explicit interaction $\left(f_0,f_1\right)$ appears in the multi-behavior historical data. The frequencies of occurrence of different behaviors in the historical data will be different; for example, compared to *adding to cart*, the frequency of *clicking* will be higher. We therefore use a logarithmic function to apply a smoothing operation to the $sum(f_0,f_1)$ of each explicit behavior interaction in the historical data, which gives a relatively balanced weight to each explicit behavior interaction in the loss function. The value of β is a very small positive number that is used to prevent the logarithmic function from smoothing the loss of some explicit behavior interactions to zero. We set β to one in this work.

4.3.4 Inferring Explicit Behavior Interaction

In the previous sections, we described the steps used to extract explicit behavior interaction information from the constructed heterogeneous informative graph. In this way, we can obtain a pre-trained knowledge model of explicit behavior interactions that is capable of inferring explicit behavior interaction information from multi-behavior data. More specifically, given a set of multi-behavior data $\mathcal{D}$,[1] for each user and auxiliary behavior in the data, we use the pre-trained knowledge model to obtain the values for the explicit behavior interactions between each user and each auxiliary behavior in $\mathcal{D}$. The values of the explicit behavior interactions obtained in this step lie in the range (0, 1), and in order to more intuitively reflect the distinctions between different explicit behavior interactions, we map the values obtained from the inferring function to integers. Specifically, we map values in the range $(0, 0.2)$ to zero and then divide the interval $[0.2, 1)$ into eight subintervals with a step size of 0.1. Each subinterval corresponds to an integer from one to eight in sequence, which reflects the strength of the explicit behavior interaction.

4.4 Explicit Behavior Interaction Fusion Module

In Sect. 4.3, we described the extraction of explicit behavior interactions using the constructed heterogeneous informative graph. In this section, we incorporate the learned explicit behavior interaction information into the embedding learning of users and items.

Constructing the Weighted Bipartite Graph. We first construct k user–item bipartite graphs based on the historical interactions of users and items under k behaviors, respectively, according to Eq. 1. Then, for each user–item edge in each bipartite graph constructed based on auxiliary behaviors, we assign the value of the user's explicit behavior interaction with that behavior. This process yields k-1 weighted bipartite graphs of auxiliary behaviors $\left\{\mathcal{G}_{b_0}, \mathcal{G}_{b_1}, \dots, \mathcal{G}_{b_{k-1}}\right\}$.

Note that the multi-behavior bipartite graphs constructed here are different from those described in previous works [14, 16, 19, 20, 36, 40], as our weighted graphs actually contain explicit information on behavior interactions that was learned via the *explicit behavior interaction extraction module* and can more accurately reflect the user's preferences in regard to different behaviors.

Fusion of multi-behavior explicit behavior interactions. Similarly to previous works on multi-behavior recommendations [14, 16], we apply a refined LightGCN [8] to the constructed bipartite graph for message propagation of neighbor nodes as follows:

$$\boldsymbol{e}_{\boldsymbol{u}}^{(\boldsymbol{b},\boldsymbol{l})} = \begin{cases} \sum_{i\in\mathcal{N}_u} \frac{p_{u,b}}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{b},\boldsymbol{l}-\boldsymbol{1})}, & \text{If } b \text{ is auxiliary behavior} \\ \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{b},\boldsymbol{l}-\boldsymbol{1})}, & \text{otherwise} \end{cases} \tag{11}$$

where $\boldsymbol{e}_{\boldsymbol{u}}^{(\boldsymbol{b},\boldsymbol{l})}$ and $\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{b},\boldsymbol{l}-\boldsymbol{1})}$ represent the embeddings of user u and item i, respectively, for auxiliary behavior b after propagation over layers l and l-1. $\mathcal{N}_u$ denotes the set of items that are interacted with user u, and $\mathcal{N}_i$ denotes the set of users that interact with item i. $p_{u,b}$ is the explicit behavior interaction $(user, behavior)$ between user u and behavior b. The definition of $\boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{b},\boldsymbol{l})}$ is similar to that of $\boldsymbol{e}_{\boldsymbol{u}}^{(\boldsymbol{b},\boldsymbol{l})}$.

To avoid over-smoothing, we concatenate the embeddings of each layer using a structure consistent with that described in previous work [7, 8], thus obtaining the user and item embeddings as follows:

$$\boldsymbol{e}_{\boldsymbol{u}}^{\boldsymbol{b}} = \Big\Vert_{l=0}^{L} \boldsymbol{e}_{\boldsymbol{u}}^{(\boldsymbol{b},\boldsymbol{l})}; \quad \boldsymbol{e}_{\boldsymbol{i}}^{\boldsymbol{b}} = \Big\Vert_{l=0}^{L} \boldsymbol{e}_{\boldsymbol{i}}^{(\boldsymbol{b},\boldsymbol{l})} \tag{12}$$

We then concatenate the embedding obtained from each behavior bipartite graph to get the final user and item embedding as follows:

[1] We divided datasets in a 1:1 ratio, with one part used to pre-train the knowledge model in the previous section and the other part is $\mathcal{D}$ that will be used in the explicit behavior interaction fusion module.

$$\boldsymbol{e}_{\boldsymbol{u}} = \Big\Vert_{b=0}^{B} \boldsymbol{e}_{\boldsymbol{u}}^{\boldsymbol{b}}; \quad \boldsymbol{e}_{\boldsymbol{i}} = \Big\Vert_{b=0}^{B} \boldsymbol{e}_{\boldsymbol{i}}^{\boldsymbol{b}} \tag{13}$$

where b denotes the specific auxiliary behavior.

Finally, the predicted value of the target behavior is obtained as the inner product of the embedding of the user and the item:

$$y(u,i) = \boldsymbol{e}_{\boldsymbol{u}}^{\boldsymbol{T}}\boldsymbol{e}_{\boldsymbol{i}} \tag{14}$$

We choose pairwise learning strategies for the *explicit behavior interaction fusion module*. Specifically, we select the Bayesian personalized ranking (BPR) loss, in which the core idea is that items that users have interacted with (i.e., positive samples) should have higher prediction scores than items without interaction (i.e., negative samples). Formally, the optimization function is expressed as:

$$Loss_2 = \sum_{(u,i,j\in O)} -\ln\sigma(y(u,i) - y(u,j)) + \lambda \cdot \|\Theta\|^2 \tag{15}$$

where $O = \{(u,i,j) \mid (u,i) \in R^+, (u,j) \in R^-\}$, (u,i,j) is the set of pairwise target behavior training data, and R^+ and R^- denote the sets of positive and negative samples of items, i.e., items that have been interacted with and those that have not been interacted with via the target behavior, respectively. $\sigma(\cdot)$ denotes the sigmoid function, while Θ denotes all of the trainable parameters in explicit behavior fusion module. We apply L_2 regularization to prevent over-fitting, where λ is the coefficient used to control the regularization.

4.5 Complexity Analysis

4.5.1 Time Complexity

We analyze the time complexity of different modules of MB-EBIH from the following aspects: (1) In the explicit behavior interaction extraction module, constructing the heterogeneous information graph takes $O\big((|\mathcal{U}| + |\mathcal{I}|) \times |\mathcal{B}| \times N_b\big)$, where $|\mathcal{U}|$ and $|\mathcal{I}|$ the total number of users and items, respectively, $|\mathcal{B}|$ is the number of types of behaviors and N_b is the number of user interaction entries under each auxiliary behavior; the time consumed by the process of L-layer heterogeneous graph convolution is $O\big(L \times \big((|\mathcal{U}| + |\mathcal{I}|) \times d_0 + |\mathcal{B}| \times d_1 + |\mathcal{E}| \times \big(d_0 + d_1\big)\big)\big)$, where d_0 and d_1 represent the dimensions of users (items) and behaviors, respectively. $|\mathcal{E}|$ represents the number of edges in the heterogeneous graph, and L represents the number of layers in the GAT; Inference of explicit behavior interactions takes $O\big(|\mathcal{B}| \times (|\mathcal{U}| + |\mathcal{I}|) \times \big(d_0 + d_1\big)\big)$. (2) The explicit behavior interaction fusion module consumes $O\big(|\mathcal{B}| \times \big(|\mathcal{U}| + |\mathcal{I}| + \big|\mathcal{E}_b\big|\big)\big)$, where $\big|\mathcal{E}_b\big|$ represents the number of edges in each behavioral bipartite graph.

4.5.2 Space Complexity

The primary memory consumption of MB-EBIH mainly arises from the following components: the constructed heterogeneous information graph, the embeddings of users, items and behavioral nodes, the weighted multi-behavior bipartite graphs and the GAT. To be more specific, the memory usage for the constructed heterogeneous graphs is $O\big((|\mathcal{U}| + |\mathcal{I}|) \times d_0 + |\mathcal{B}| \times d_1 + |\mathcal{E}|\big)$. For the weighted multi-behavior bipartite graphs, the memory allocation is $O\big(|\mathcal{B}| \times \big((|\mathcal{U}| + |\mathcal{I}|) \times d_0 + \big|\mathcal{E}_b\big|\big)\big)$. The memory consumed by the trainable parameters of the GAT for the explicit behavior interaction extraction layer is $O\big(L \times \big((|\mathcal{U}| + |\mathcal{I}|) \times d_0^2 + |\mathcal{B}| \times d_1^2\big)\big)$.

4.6 Discussion

As previously mentioned, our proposed MB-EBIH distinguishes itself significantly from existing multi-behavior recommendation models by explicitly modeling interactions between multiple behaviors. Existing models, relying on implicit modeling, typically create separate user–item bipartite graphs for each behavior, utilizing the user's interaction data across various behaviors. They subsequently apply deep learning techniques to learn and capture the interactions between these behaviors, thereby assisting in the prediction of the user's target behavior.

However, in real-world scenarios where user behavior is inherently intricate, this black-box implicit modeling approach makes it difficult to accurately distinguish which behaviors are more beneficial for user modeling. In contrast, MB-EBIH is grounded in the user's historical interaction data, using the target behavior as a label to statistically quantify the explicit quantitative relationship between auxiliary behaviors and the target behavior. This approach allows for a more precise modeling of the interaction relationship between multiple behaviors, leveraging this quantitative relationship as a posteriori information. The effectiveness of this explicit modeling of behavioral interactions is further demonstrated through experiments in Sect. 5.5.2. Additionally, such a posteriori statistical features obtained through explicit modeling are frequently integrated into industrial scenarios to enhance the model's recommendation performance [22, 23].

5 Experiments

In this section, we describe extensive experiments conducted on four real-world datasets from different scenarios to evaluate the effectiveness of our proposed MB-EBIH approach and compare it with various existing recommendation methods.

Table 1 Statistics of the datasets used in our experiments

Dataset	User#	Item#	Interaction#	Behavior type#
Beibei	21,716	7,977	3.3×10^6	{Click,Cart,Buy}
Tmall	41,738	11,953	2.3×10^6	{Click,Cart,Collect,Buy}
IJCAI15	55,038	28,728	7.5×10^6	{Click,Cart,Collect,Buy}
QK-article	40,343	19,218	2.4×10^6	{Click,Share,Follow,Like}

5.1 Experimental Settings

5.1.1 Dataset

To evaluate the performance of MB-EBIH and the generalization of the explicit behavior interactions, we choose four datasets from real-world platforms in different domains, as described in detail below:

- *Beibei*: This dataset was collected from Beibei,[2] which is the largest e-commerce platform for baby products in China. It contains 21,716 users and 7,977 items with three types of user–item behaviors, including *clicking*, *adding to cart* or *carting* for short and *buying*.
- *Tmall*: This dataset was collected from Tmall,[3] one of the largest e-commerce platforms in China. There are 41,738 users and 11,953 items with four types of behavior, including *clicking*, *carting*, *buying* and *collecting*.
- *IJCAI15*: This dataset was released from the IJCAI Contest 2015,[4] which is focused on the task of predicting repeat buyers. To ensure that the training data were not too sparse, we filtered out users who bought fewer than 15 times and items that were bought fewer than 20 times. We were left with 55,038 users and 28,728 items, with the same four behaviors as in the Tmall dataset.
- *QK-article* [41]: This dataset was collected from Tencent's news article recommendation platform. Similar to IJCAI15, we filtered out users who bought fewer than 5 times and items that were bought fewer than 5 times. We were left with 40,343 users and 19,218 items with four types of behaviors, including *clicking*, *following*, *sharing*, *liking*.

In this study, we divided each dataset into two parts, in a 1:1 ratio. The first subdataset, was used in the *explicit behavior interaction extraction module* to construct weighted heterogeneous informative graphs, to extract explicit behavior interaction information, while the second, was used in the *explicit behavior interaction fusion module* to construct multi-behavior bipartite graphs to incorporate the explicit behavior interaction information into the embedding learning. In the *explicit behavior interaction fusion module*, we further divided the subdataset into a training set, a validation set and a test set in an 8:1:1 ratio. For the e-commerce platform dataset, we considered *buying*(i.e., the final optimization goal of our model) as the target behavior, while for the news article platform dataset, we chose *liking* as the target behavior, and the other types of behaviors were treated as auxiliary behaviors. Statistical information on the four datasets used in our experiments is summarized in Table 1.

To quantify the performance of each model, we selected two widely used metrics, called recall and normalized discounted cumulative gain (NDCG), which were defined as explained below:

- *Recall@K* quantifies the proportion of relevant items from a test set that are correctly included in the top-K recommendation list. It measures the ability of the system to recall and capture relevant items from the recommended options. The higher the value of Recall@K, the better the ability of the system in terms of recalling relevant items.
- *NDCG@K* evaluates the quality of the ranking of recommended items by assigning higher scores to relevant items that are ranked higher in the top-K list. It emphasizes the importance of both the relevance and the position of each item, with the aim of prioritizing and promoting higher-ranked relevant items in the recommendation list. A higher NDCG@K value indicates a better-ranked list of relevant items.

5.1.2 Baselines

To showcase the efficiency of our MB-EBIH model, we conducted a comparative analysis with various other methods. We categorized the baselines into two groups: single-behavior models, which rely solely on target behavior records, and multi-behavior models, which consider all types of behaviors.

Single-behavior Models:

- *MF-BPR* [3]: This method has demonstrated strong performance in the top-n recommendation task and is frequently employed as a benchmark for assessing

[2] http://www.beibei.com/

[3] http://tianchi.aliyun.com/dataset/dataDetail?dataId=649.

[4] http://http://tianchi.aliyun.com/dataset/42.

the effectiveness of new models. The BPR approach has been extensively utilized as an optimization strategy and is based on the assumption that positive items should receive higher scores than negative items.

- *NGCF* [7]: This is a state-of-the-art graph neural network model that was specially designed to combine graph neural network with an RS.
- *LightGCN* [8]:This state-of-the-art GCN-based recommendation model represents a breakthrough in leveraging high-order neighbors within the user–item bipartite graph to deliver accurate recommendations.

Multi-behavior Models:

- *NMTR* [17]: This is a state-of-the-art method that uses multitask learning to update NCF for multi-behavior tasks. For each type of behavior, it constructs a data-dependent interaction function and links the model predictions for each type of behavior in a cascading fashion.
- *MBGCN* [14]: This is a state-of-the-art multi-behavior recommendation model based on GCN. It effectively considers the varying contributions of multiple behaviors to the target behavior based on a unified graph. It learns the behavior contributions and leverages an item–item graph to capture the behavior semantics.
- *MATN* [18]: This model incorporates attention networks and memory units to distinguish and capture the relationships between users and items.
- *MB-GMN* [29]: This model utilizes a graph meta network to capture personalized signals from multiple behaviors and to effectively model the diverse dependencies between them.
- *GNMR* [40]: This GNN-based approach explores multi-behavior dependencies through recursive embedding propagation on a unified graph. It employs a relation aggregation network to effectively model the heterogeneity of interactions within the graph.
- *CRGCN* [15]: This model utilizes a cascading GCN structure to effectively model multi-behavior data. It employs a residual design to deliver the learned behavioral features from one behavior to the next.
- *MB-CGCN* [16]: This is a recently proposed model that adopts cascading CGN blocks to explicitly leverage multiple behaviors for embedding learning. In this model, a LightGCN learns the features of previous behavior and transfers them to the subsequent behavior through a feature transformation operation. The embeddings obtained from all behaviors are then aggregated to create the final prediction.

5.1.3 Hyper-parameter Settings

We implemented our MB-EBIH model using PyTorch, and the model was optimized using the Adam optimizer with learning rate $3e^{-4}$.

For *explicit behavior interaction extraction module*, we set the dimensions of the user(item) and behavior nodes to eight and four, respectively, and conducted detailed experiments to explore the effect of node dimension in Sect. 5.6.1. For GAT, we set the number of layers L=2 and K=1 in the multi-head mechanism, and set the size of the hidden layer to 64 and the size of the last layer to four. For $f(\cdot)$ in Eqs. 8 and 9, in order to correspond to the dimensions of the user, item and behavior node, we set the input size to 12 and the output size to 1. We set β in the $Loss_1$ to one.

In the *explicit behavior interaction fusion module*, since the purpose of the embedding of users and items obtained via the *explicit behavior interaction extraction module* is to infer user behavior explicit behavior interactions, it is not directly applicable to *explicit behavior interaction fusion module*, so we reset the embedding size to 64 for users and items, and set the batch size to 4096. After several validation experiments, we found that setting the number of layers L of LightGCN to one yielded the best results. To prevent overfitting, we also applied message dropout and node dropout, with values set to 0.2 [14]. We set λ in the L_2 regularization to $1e^{-4}$. For the baseline models, we used the hyper-parameter settings givens in the original papers.

In addition, in order to enhance the stability of the model training process, for both *the explicit behavior interaction extraction module* and *the explicit behavior interaction fusion module*, we selected *ReLU* as the activation function. And in order to initialize the training parameters of the model better, we use the *Kaiming* initializer [42] to initialize the parameters of the two modules mentioned above.

5.2 Overall Performance

In this section, we compare our MB-EBIH model with the other baselines. The results from the four datasets are reported in Table 2.

The best values are highlighted in bold and the second best are underlined. From the results, we can draw the following conclusions:

- *Comparison of model performance.* Table 2 shows that our MB-EBIH outperforms all the baseline models in terms of both the Recall@K and NDCG@K metrics ($K = \{10, 20, 40\}$). Compared to the single-behavior model, we introduce multiple behaviors to reflect user preferences more comprehensively. Compared with the NN-based model, we employ GCN to obtain higher-order neighbor information. Different from the

Table 2 Overall performance comparison between MB-EBIH and baseline models on four datasets

Dataset	Metric	Single-behavior methods			Multi-behavior methods							MB-EBIH
		MF-BPR	NGCF	LightGCN	NMTR	MBGCN	MATN	MB-GMN	GNMR	CRGCN	MB-CGCN	
Beibei	Recall@10	0.0868	0.0901	0.0925	0.0937	0.1249	0.0942	0.1040	0.1105	0.1215	0.1539	**0.1911**
	NDCG@10	0.0414	0.0521	0.0538	0.0571	0.0745	0.0579	0.0586	0.0635	0.0986	0.1095	**0.1235**
	Recall@20	0.1244	0.1463	0.1517	0.1559	0.1925	0.1604	0.1608	0.1635	0.2173	0.2362	**0.2807**
	NDCG@20	0.0641	0.0652	0.0698	0.0796	0.0937	0.0790	0.0740	0.0794	0.1330	0.1547	**0.1674**
	Recall@40	0.1965	0.2311	0.2328	0.2644	0.2938	0.2610	0.2863	0.2872	0.2778	0.3388	**0.3694**
	NDCG@40	0.0817	0.0853	0.0887	0.0995	0.1157	0.1071	0.1078	0.1096	0.1585	0.1910	**0.1991**
Tmall	Recall@10	0.0263	0.0391	0.0417	0.0470	0.0692	0.0562	0.0893	0.0601	0.0901	0.0925	**0.0985**
	NDCG@10	0.0151	0.0225	0.0291	0.0293	0.0424	0.0367	0.0462	0.0388	0.0568	0.0594	**0.0644**
	Recall@20	0.0355	0.0491	0.0553	0.0672	0.0947	0.0790	0.1072	0.0838	0.1136	0.1231	**0.1280**
	NDCG@20	0.0176	0.0356	0.0333	0.0360	0.0493	0.0408	0.0592	0.0452	0.0631	0.0696	**0.0732**
	Recall@40	0.0481	0.0716	0.0719	0.0982	0.1275	0.1042	0.1345	0.1190	0.1543	0.1594	**0.1664**
	NDCG@40	0.0200	0.0350	0.0361	0.0412	0.0558	0.0517	0.0674	0.0529	0.0749	0.0778	**0.0803**
IJCAI15	Recall@10	0.0217	0.0281	0.0286	0.0314	0.0468	0.0350	0.0558	0.0417	0.0562	0.0701	**0.0718**
	NDCG@10	0.0116	0.0153	0.0180	0.0211	0.0304	0.0217	0.0376	0.0272	0.0351	0.0428	**0.0500**
	Recall@20	0.0311	0.0351	0.0387	0.0461	0.0661	0.0412	0.0750	0.0600	0.0884	0.0891	**0.0939**
	NDCG@20	0.0191	0.0198	0.0211	0.0263	0.0358	0.0242	0.0432	0.0393	0.0412	0.0504	**0.0563**
	Recall@40	0.0489	0.0501	0.0519	0.0707	0.0906	0.0714	0.0960	0.0779	0.0924	0.1135	**0.1221**
	NDCG@40	0.0225	0.0231	0.0240	0.0309	0.0415	0.0316	0.0518	0.0327	0.0510	0.0598	**0.0629**
QK-article	Recall@10	0.0615	0.0641	0.0692	0.0704	0.0711	0.0708	0.0946	0.0860	0.1173	0.1359	**0.1509**
	NDCG@10	0.0333	0.0349	0.0357	0.0366	0.0384	0.0380	0.0495	0.0488	0.0691	0.0796	**0.0892**
	Recall@20	0.1021	0.1063	0.1064	0.1107	0.1178	0.1122	0.1402	0.1366	0.1767	0.2005	**0.2205**
	NDCG@20	0.0445	0.0465	0.0493	0.0502	0.0510	0.0506	0.0622	0.0581	0.0748	0.0934	**0.1084**
	Recall@40	0.1673	0.1745	0.1780	0.1821	0.1844	0.1838	0.2202	0.2084	0.2534	0.2846	**0.3096**
	NDCG@40	0.0589	0.0613	0.0625	0.0650	0.0662	0.0658	0.0834	0.0790	0.0907	0.1121	**0.1282**

structure of GCN-based models, our model explicitly exploited interactions between behaviors through a heterogeneous behavior informative graph. By introducing explicit behavior interaction information, our model can more accurately capture users' personalized preferences under different behaviors.

- *Importance of GNN in the RS.* From an investigation of the performance of the single-behavior models in Table 2, we can see that the two GCN-based models, NGCF and LightGCN, perform better than traditional MF, thus proving that the ability of GCN to explore higher-order neighbor information can enable the model to learn more efficient embeddings of users and items.
- *Importance of multiple behaviors in the RS.* From the results in Table 2, we can see that the single-behavior-based models MF-BPR, LightGCN and NGCF gave inferior performance to the multi-behavior-based models, which demonstrates the necessity of considering multi-behavior data in the RS. By exploring the dependencies between multiple behaviors, multi-behavior recommendation models can model user preferences from multiple perspectives

5.3 Modeling User Personalized Behavioral Preferences

In the real world, the probability that a user will perform a target behavior after performing an auxiliary behavior is individualized; for example, some users will definitely buy an item after adding it to the cart, while others will not (i.e., the proportion of *purchases* in the *cart* records for these users are different).

As shown in Fig. 4, for the Beibei dataset, we counted the records of cart behavior for different users and calculated the proportions of purchase behavior, and then grouped the users based on this proportion. From the statistics in Fig. 4, it can be intuitively seen that in real-world scenarios, there are significant personalized differences between different users in terms of whether or not to buy after the cart behavior. This indicates that in the process of user modeling, the user's personalized behavioral preference is a very important feature information, and a more comprehensive portrait of the user can be portrayed by modeling it.

To demonstrate that our proposed MB-EBIH model could accomplish the above mentioned personalized preference modeling of auxiliary behaviors, we compared it with

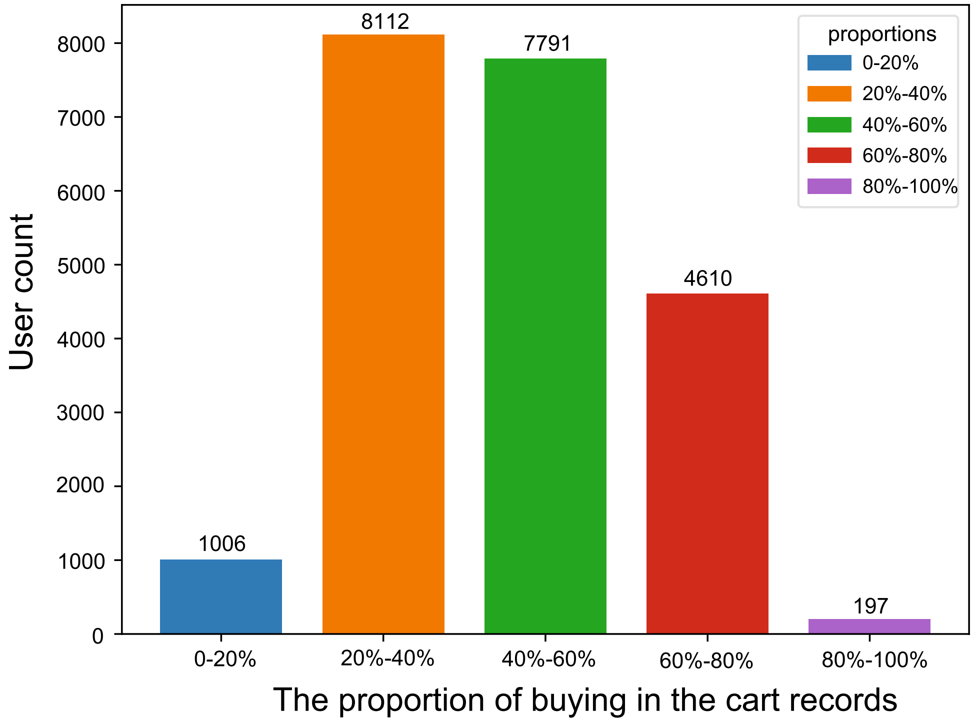


Fig. 4 Proportion of buying behavior from the cart records for different users in the Beibei dataset. Users are classified into hierarchical levels based on this proportion

MBGCN and MB-CGCN on the Beibei dataset. Specifically, we applied the criteria for grouping users in Fig. 4 and considered each group of users as a test set for two different auxiliary behaviors, *adding to cart* and *clicking*, to investigate the performance of the three models on the different groups.[5] The model's proficiency in capturing personalized behavioral preferences can be confirmed by assessing its performance on subgroups of users with diverse individualized behavioral preferences. The results for $K = 10$ are summarized in Fig. 5.

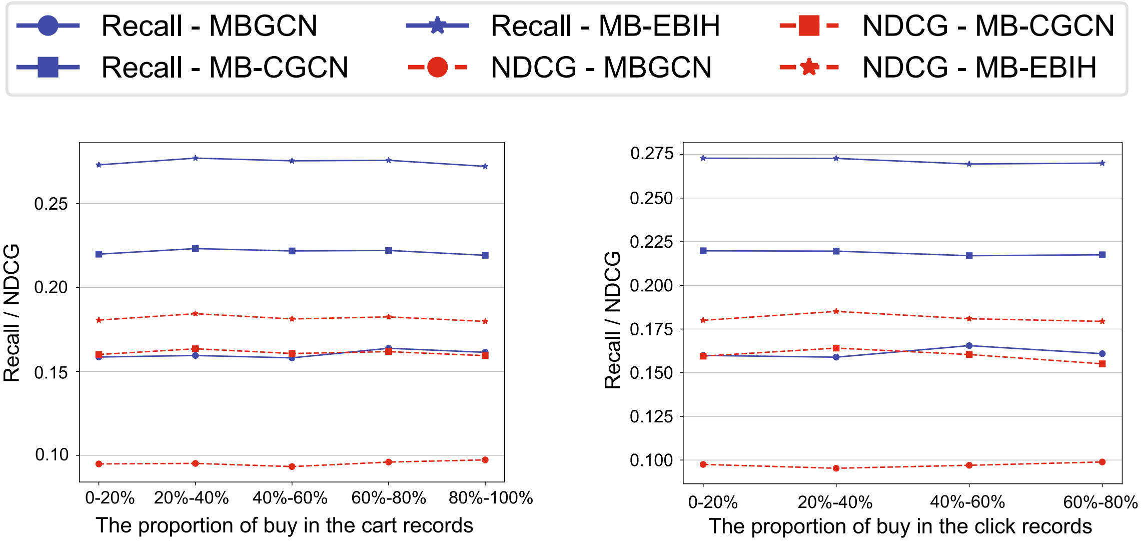
Figure 5 shows that MB-EBIH consistently outperforms both MB-CGCN and MBGCN, regardless of Recall@10 or NDCG@10, for all users grouped under different auxiliary behaviors. When compared to MB-CGCN, a model that explicitly explores dependencies between behaviors based on a chain of behaviors, MB-EBIH demonstrates its capability in modeling user personalized behavioral preferences by consistently delivering excellent performance across subgroups of users with different personalized behavioral preferences. This also confirms the effectiveness of explicit behavior interactions proposed as a posteriori information in this paper in modeling user personalized behavioral preferences. Moreover, MB-EBIH consistently outperforms MBGCN, a model that aggregates multi-behavior information in a weighted manner to distinguish the contribution of behaviors, thereby validating the effectiveness of explicit behavior interaction. Furthermore, in addition to explicitly modeling behavioral interactions, another significant reason for the consistent outperformance of MB-EBIH over MBGCN and MB-CGCN across user subgroups is the incorporation of negative feedback signals related to auxiliary behaviors. Both of the mentioned models overlook the modeling of these negative feedback signals in auxiliary behaviors, which is an essential aspect of personalization. Consequently, they fail to accurately capture the user's personalized behavioral preferences, leading to their underperformance across different user subgroups.

5.4 Effect of the Graph Attention Mechanism

To verify the effectiveness of GAT in our model,[6] we considered four variants:(1) **GraphSage:** We utilized GraphSage [43] as a heterogeneous graph aggregator method for the *explicit behavior interaction extraction module*; (2) ***k-GNNs:*** The heterogeneous graph convolution aggregator was replaced with k-GNNs [44]. (3) **LEGonv:** The heterogeneous graph convolution aggregator was replaced with LEGonv [45]. (4) **GAT:** This was the model proposed in this

[5] For reasons of space, we only report results for the Beibei dataset here, but we note that MB-EBIH also outperformed the two baseline models on the other three datasets.

[6] Since the convolution method in the explicit behavior interaction fusion module was not the focus of this work, the effect of the parameters in this module was not investigated in this study.



(a)

(b)

Fig. 5 Comparison of MBGCN, MB-CGCN and MB-EBIH for different subgroups of users with different proportions of buying behavior from the auxiliary behaviors (*cart* and *click* records) from the Beibei dataset

Table 3 Effect of GAT in MB-EBIH (results based on $K = 10$)

Dataset	Beibei		Tmall		IJCAI15		QK-article	
Method	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
GraphSage	0.1458	0.0867	0.0722	0.0419	0.0619	0.0415	0.1329	0.0765
k-GNNs	0.1838	0.1218	0.0782	0.0469	0.0662	0.0440	0.1292	0.0743
LEConv	0.1866	0.1221	0.0796	0.0482	0.0643	0.0419	0.1387	0.0805
GAT	**0.1911**	**0.1235**	**0.0985**	**0.0644**	**0.0718**	**0.0500**	**0.1509**	**0.0892**

Best values are highlighted in bold

work. For each aggregator, we set the number of layers L = 2 and tuned the parameters to achieve the best performance, to enable a fair comparison.

A comparison of the four heterogeneous graph convolution aggregators in Table 3 shows that MB-EBIH achieves the best performance with the GAT method; this indicates that the introduction of the attention mechanism is necessary, and that the attention mechanism allows the model to better distinguish the contribution of different behaviors to the user. In this way, the model can capture more accurate explicit behavior interactions between the user and the behavior.

5.5 Ablation Study

5.5.1 Effects of Negative Feedback Signals

To demonstrate the effectiveness of negative feedback signals (NFSs) in regard to auxiliary behaviors in multi-behavior RS, we consider two variants: (1) ***w/o. NFS***: in which we remove the auxiliary behavior negative feedback signal nodes (such as $click_0$ and $cart_0$) from the heterogeneous behavioral informative graph in the *explicit behavior interaction extraction module*; and (2) ***w. NFS***, i.e., the original MB-EBIH model.

Figure 6 shows that on the four real-world datasets, ***w. NFS*** consistently outperforms ***w/o. NFS***, which demonstrates the effectiveness of considering negative feedback signals from auxiliary behaviors in multi-behavior RS.

5.5.2 Effects of Different Explicit Behavior Interactions

To demonstrate the effectiveness of the proposed explicit behavior interactions and to explore their impact on the performance of the model, three sets of ablation experiments based on different research questions were conducted for datasets in different real-world scenarios as summarized below:

Single-Behavior Ablation:

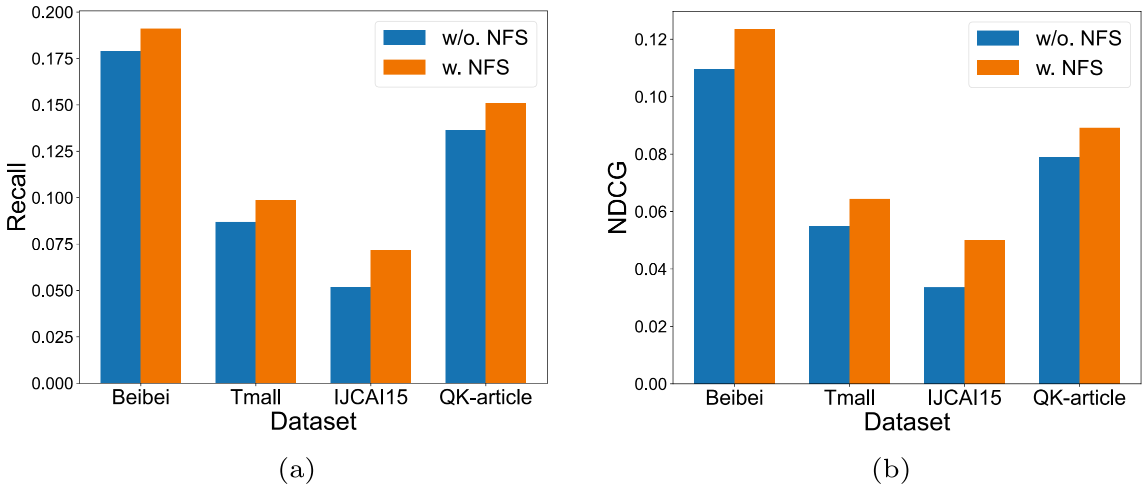


Fig. 6 Effects of negative feedback signals in four datasets (results based on $K = 10$)

Table 4 Effects of different explicit behavior interactions (results based on $K = 10$)

Dataset	Beibei		Tmall		IJCAI15		QK-article	
Variants	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
Base model	**0.1911**	**0.1235**	**0.0985**	**0.0644**	**0.0718**	**0.0500**	**0.1509**	**0.0892**
w/o. click	0.1899	0.1218	0.0703	0.0432	0.0479	0.0310	0.0757	0.0412
w/o. cart(follow)	0.1229	0.0706	0.0951	0.0593	0.0706	0.0485	0.1447	0.0843
w/o. collect(share)	/	/	0.0899	0.0550	0.0712	0.0492	0.1488	0.0862
w/o. cart,click	0.1170	0.0672	0.0674	0.0419	0.0503	0.0329	/	
w/o. cart,collect	/		0.0880	0.0537	0.0596	0.0386		
w/o. collect,click			0.0677	0.0411	0.0467	0.0295		
w/o. follow,click			/				0.0747	0.0406
w/o. follow,share							0.1436	0.0826
w/o. share,click							0.0707	0.0378
w/o. collect,click,cart	/		0.0631	0.0396	0.0413	0.0257	/	
w/o. share,click,follow			/				0.0699	0.0376

The explicit behavior interactions for a single behavior were removed, and for the e-commerce datasets, we consider three variant: *w/o click*, *w/o cart* and *w/o collect*; for the news article dataset we similarly consider three variants: *w/o click*, *w/o follow* and *w/o share*.

Multiple Behaviors Ablation:

The explicit behavior interactions for any two behaviors were removed, for the e-commerce datasets we consider three variant: *w/o cart, click*, *w/o cart, collect* and *w/o collect, click*; for the news article dataset we set the variants to *w/o click, share*, *w/o click, follow* and *w/o follow, share*.

All Behaviors Ablation:

The explicit behavior interactions for all auxiliary behaviors were removed, i.e., *w/o cart, collect, click* and *w/o click, follow, share*. It is worth noting that the set of experiments can also serve as an ablation study for the *explicit behavior interaction extraction module*. This is because removing all explicit behavior interactions for auxiliary behaviors is equivalent to removing the *explicit behavior interaction extraction module*, and MB-EBIH degenerates into a traditional multi-behavior recommendation model.

We note that since the Beibei dataset does not include the *collect*, the results for *w/o cart, click* in the second set of ablation experiments were considered as the results for the third set on the Beibei dataset. The results of the ablation experiments are presented in Table 4, where the best values are highlighted in bold and the second best results are underlined.

The results of these single-behavior ablation experiments reveal that regardless of which type of user explicit behavior interactions are removed, the performance of the model deteriorates on all four real-world datasets. This demonstrates the effectiveness of the explicit behavior interactions. Furthermore, for the Beibei dataset, removing the (*user*, *cart*) explicit behavior interaction has a greater impact on the performance of model compared to removing the (*user*, *click*).

Table 5 Statistics on the number of different sequences of user behavior in the four datasets when buying an item (a value of zero means that the user did not perform the corresponding auxiliary behavior when buying the item)

Dataset	Beibei		QK-article	
Behavior orders	Count	Percentage (%)	Count	Percentage (%)
click, cart, buy	**289,080**	**98.71**	/	
0, 0, buy	3,411	1.16		
click, 0, buy	369	0.13		
click, 0, 0, like	/		**533,667**	**90.65**
click, 0, share, like			46678	7.93
click, follow, 0, like			6219	1.06
click, follow, share, like			2117	0.36
Dataset	Tmall		IJCAI15	
Behavior orders	Count	Percentage (%)	Count	Percentage (%)
click, 0, 0, buy	**211,445**	**73.63**	**481,876**	**73.96**
0, 0, 0, buy	46,438	16.17	102,334	15.71
click, 0, collect, buy	24,092	8.39	55,799	8.56
0, 0, collect, buy	4,838	1.68	10,631	1.63
click, cart, 0, buy	291	0.101	787	0.121
click, cart, collect, buy	29	0.01	75	0.012
0, cart, 0, buy	23	0.008	53	0.0081
0, cart, collect, buy	2	0.0007	6	0.0009

The sequences of behaviors with the highest percentage are bolded

The opposite was observed for the Tmall and IJCAI15 datasets. The underlying reason for this lies in the inconsistency in the distribution of multi-behavior data between the Beibei dataset and the other two datasets.

As shown in Table 5, for the Beibei dataset, there are three sequences of behaviors that were performed by users when buying items: (1) *click* → *cart* → *buy*, i.e., a user buys an item after clicking and adding to the cart; (2) *buy*, i.e., a user buys an item directly, without performing any other auxiliary behaviors; (3) *click* → *buy*, i.e., a user buys an item directly after clicking, without adding it to the cart.

Table 5 shows that the sequence *click* → *cart* → *buy* represents the highest proportion of behaviors for the Beibei dataset, with 98.71%. This results in a strong association between *adding an item to the cart* and a *purchase* by the user. Consequently, removing (*user*, *cart*) greatly diminishes the model's capability to predict the probability of the target behavior.

Similarly, for the Tmall and IJCAI15 datasets, Table 5 shows that the *click* → *buy* sequence makes up the highest proportion of behaviors, which explains why removing (*user*, *click*) results in a significant degradation in the performance of the model compared to the explicit behavior interactions for the other auxiliary behaviors. A similar approach can be used to analyze the results of the three ablation experiments performed on the QK-article dataset using MB-EBIH. As shown in Table 5, the highest percentage of *click* → *like* sequence is found in the QK-article dataset at 90.65%. This indicates a stronger correlation between *clicking* and *liking*, which leads to the worst performance of the *w/o. click* variant in Table 4.

From these multiple behaviors ablation experiments, it can be observed that compared to the single-behavior ablation experiments, removing explicit behavior interactions for two auxiliary behaviors simultaneously further reduces the performance of model. This not only demonstrates that incorporating multiple behaviors into the RS can effectively improve performance but also further validates the effectiveness of the proposed explicit behavior interactions.

In the ablation experiments with all auxiliary behaviors, when the explicit behavior interactions for all auxiliary behaviors are removed (which is equivalent to removing the *explicit behavior interaction extraction module* of MB-EBIH), the model becomes a traditional multi-behavior recommendation model. Compared to the other two experiments described above, the performance metrics for the model are at the lowest level. This not only further illustrates the importance of the explicit behavior interactions proposed in this paper for multi-behavior recommendation models, but also demonstrates the necessity of the *explicit behavior interaction extraction module* in MB-EBIH.

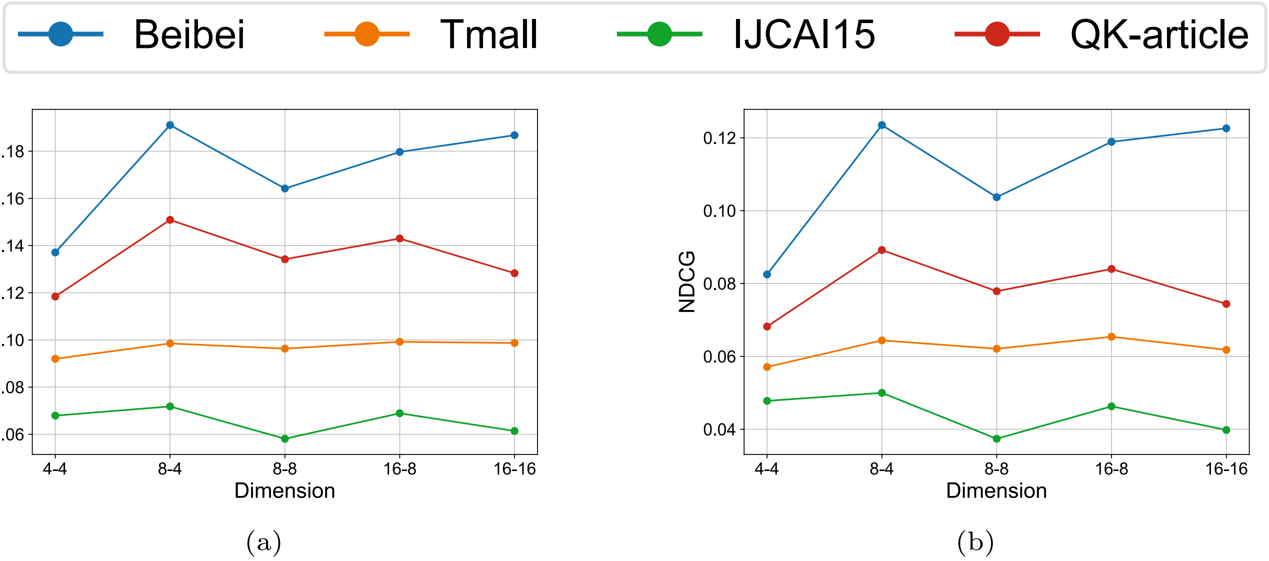


Fig. 7 Impact of node dimension (results based on $K = 10$)

5.6 Parameter Sensitivity Analysis

5.6.1 Impact of Node Dimensions

An appropriate setting for the node dimensions not only enables the learning of better node features, but also reduces the complexity of the model and prevents overfitting. To investigate the impact on the performance of the model of different user(item) and behavior node dimensions in the constructed heterogeneous graph, we explored different combinations of the user(item) dimension d_0 and the behavior node dimension d_1, where d_0 and d_1 were set to $\{4, 8, 16\}$. Specifically, we considered five combinations: (4, 4), (8, 4), (8, 8), (16, 8), (16, 16), where (8, 4) means that the dimension d_0 of the user(item) node is eight, and the dimension d_1 of the behavior node is four. The definitions of the other combinations are similar. The results for $K = 10$ are summarized in Fig. 7.

Figure 7 presents a comparison of the results for the Recall@10 and NDCG@10 metrics of the model, for five combinations of node dimensions. We can intuitively see that for the Beibei, QK-article and IJCAI15 datasets, the model obtains the best performance with the combination **(8,4)**, while for the Tmall dataset, the performance of the model for the four different combinations of dimensions does not show significantly fluctuations except for **(4, 4)**. To reduce the complexity of the model, we adopted the **(8,4)** combination for all four datasets in the *explicit behavior interaction extraction module*.

5.6.2 Impact of GAT Layer Numbers

In order to explore how the depth of the GAT module affected the performance of our model, we conducted experiments with varying number of layers ($L = 1, 2, 3$). To ensure a fair comparison, the node dimensions of the heterogeneous graphs constructed based on the four datasets were consistent. The results are presented in Fig. 8.

Figure 8 shows that for all four datasets, the model performance for $L = 2$ was significantly higher than that for $L = 1$, which demonstrates the effectiveness of the multilayer GAT in terms of capturing more accurate explicit behavior interaction messages through learning higher-order neighborhood messages from the constructed heterogeneous graph. When $L = 3$, the performance of the model decreased in different scales on all four datasets; this was because as the number of layers of GAT increases, the node features in the heterogeneous graph are affected by over-smoothing, which leads to distortion of the learned explicit interaction information and consequently affects the performance of the model. We therefore set the number of GAT layers to two during training of the model.

6 Conclusion

In this work, for the inadequacy of existing multi-behavior recommendation methods in modeling explicit interactions of behaviors and capturing user preferences for multiple

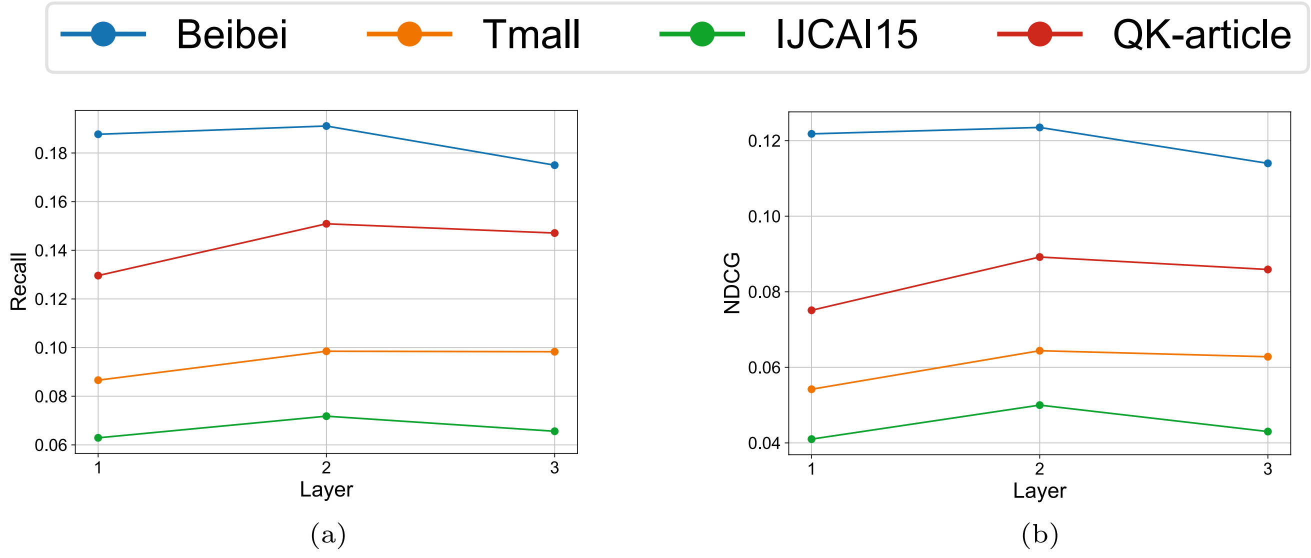


Fig. 8 Impact of GAT layer number (results based on $K = 10$)

behaviors, we have proposed a novel model called explicit behavior interaction with heterogeneous graph for multi-behavior recommendation (MB-EBIH). Our model consists of two modules. In the explicit behavior interaction extraction module, we construct a weighted heterogeneous behavior graph with nodes representing users, items and auxiliary behaviors. GAT is employed as the aggregator for learning node embeddings in a self-supervised learning task. The explicit behavior interaction values are obtained through a MLP. In the explicit behavior interaction fusion module, we construct multiple weighted bipartite graphs, using explicit behavior interaction values as the weights. These graphs are designed to integrate explicit behavior interactions between the user and multiple auxiliary behaviors into the embedded learning process.

Experiments on four real-world datasets from different domains showed that MB-EBIH outperformed all baselines. Additionally, we conducted analysis experiments to demonstrate that MB-EBIH can effectively capture personalized user preferences under different behaviors. Ablation experiments confirmed the effectiveness of the negative feedback signals from auxiliary behaviors and the necessity of explicit behavior interactions. Parameter sensitivity experiments were also conducted to investigate the impact of node dimension and the number of GAT layers on the performance of the model.

In this study, our focus was primarily on modeling the explicit behavior interactions and personalized user preferences. The model presented in this paper still has certain limitations. For instance, when applied in large-scale scenarios, the time required to construct a heterogeneous informative graph increases. One potential solution could involve dividing the interaction data into smaller segments, constructing several subgraphs, extracting the user's explicit behavior interaction information from each subgraph and subsequently combining these subgraphs to derive the ultimate explicit behavior interaction. As another example, our model may be influenced by noise in the user behavior data, potentially leading to bias. This issue will be addressed in our future research. Additionally, we plan to investigate the application of explicit behavior interactions in a broader range of recommendation scenarios in the future.

Acknowledgements This work was supported by a grant from the National Natural Science Foundation of China (No. 61972334), the Innovation Capability Improvement Plan Project of Hebei Province (22567626 H), Local Science and Technology Development Fund Project guided by the Central Government (No. 2Z1707G), the Intelligent image workpiece recognition of Sida Railway (No. x2021134)) and the Performance Appraisal System Qinghuangdao Urban and Health Industry Development CO., LTD (No. x2022247).

Author Contributions Z-ZP presents the core idea of the model and the experimental method. JY implements the model, verifies its validity and writes the paper. H-YH and Y-XL provides guidance and revised the paper.

Availability of Data and Materials Available.

Declarations

Conflict of interest The authors declare that they have no competing interests.

References

1. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on world wide web, pp 285–295
2. Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37
3. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2012) Bpr: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618
4. He X, Liao L, Zhang H, Nie L, Hu X, Chua T-S (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web, pp 173–182
5. Zhang S, Yao L, Sun A, Tay Y (2019) Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv 52(1):1–38
6. Feng F, He X, Zhang H, Chua T-S (2021) Cross-gcn: enhancing graph convolutional network with *k*-order feature interactions. IEEE Trans Knowl Data Eng 35(1):225–236
7. Wang X, He X, Wang M, Feng F, Chua T-S (2019) Neural graph collaborative filtering. In: Proceedings of the 42nd International ACM SIGIR conference on research and development in information retrieval, pp 165–174
8. He X, Deng K, Wang X, Li Y, Zhang Y, Wang M (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, pp 639–648
9. Chen L, Wu L, Hong R, Zhang K, Wang M (2020) Revisiting graph based collaborative filtering: a linear residual graph convolutional network approach. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 27–34
10. Wang X, Jin H, Zhang A, He X, Xu T, Chua T-S (2020) Disentangled graph collaborative filtering. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in information retrieval, pp 1001–1010
11. Su X, Khoshgoftaar TM (2009) A survey of collaborative filtering techniques. Adv Artif Intell 2009
12. Wang R, Fu B, Fu G, Wang M (2017) Deep & cross network for ad click predictions. In: Proceedings of the ADKDD'17, pp 1–7
13. Wu L, He X, Wang X, Zhang K, Wang M (2022) A survey on accuracy-oriented neural recommendation: from collaborative filtering to information-rich recommendation. IEEE Trans Knowl Data Eng 35:4425–4445
14. Jin B, Gao C, He X, Jin D, Li Y (2020) Multi-behavior recommendation with graph convolutional networks. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 659–668
15. Yan M, Cheng Z, Gao C, Sun J, Liu F, Sun F, Li H (2022) Cascading residual graph convolutional network for multi-behavior recommendation. arXiv preprint arXiv:2205.13128
16. Cheng Z, Han S, Liu F, Zhu L, Gao Z, Peng Y (2023) Multi-behavior recommendation with cascading graph convolution networks. In: Proceedings of the ACM web conference 2023, pp 1181–1189
17. Gao C, He X, Gan D, Chen X, Feng F, Li Y, Chua T-S, Yao L, Song Y, Jin D (2019) Learning to recommend with multiple cascading behaviors. IEEE Trans Knowl Data Eng 33(6):2588–2601
18. Xia L, Huang C, Xu Y, Dai P, Zhang B, Bo L (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, pp 2397–2406
19. Guo W, Meng C, Yuan E, He Z, Guo H, Zhang Y, Chen B, Hu Y, Tang R, Li X, et al (2023) Compressed interaction graph based framework for multi-behavior recommendation. arXiv preprint arXiv:2303.02418
20. Xia L, Huang C, Xu Y, Dai P, Lu M, Bo L (2021) Multi-behavior enhanced recommendation with cross-interaction collaborative relation modeling. In: 2021 IEEE 37th international conference on data engineering (ICDE). IEEE, pp 1931–1936
21. Xia L, Huang C, Xu Y, Dai P, Zhang X, Yang H, Pei J, Bo L (2021) Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, pp 4486–4493
22. Li F, Yan B, Long Q, Wang P, Lin W, Xu J, Zheng B (2021) Explicit semantic cross feature learning via pre-trained graph neural networks for ctr prediction. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 2161–2165
23. Shan Y, Hoens TR, Jiao J, Wang H, Yu D, Mao J (2016) Deep crossing: web-scale modeling without manually crafted combinatorial features. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 255–262
24. Tang L, Long B, Chen B-C, Agarwal D (2016) An empirical study on recommendation with multiple types of feedback. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 283–292
25. Singh AP, Gordon GJ (2008) Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp 650–658
26. Zhao Z, Cheng Z, Hong L, Chi EH (2015) Improving user topic interest profiles by behavior factorization. In: Proceedings of the 24th international conference on world wide web, pp 1406–1416
27. Guo L, Hua L, Jia R, Zhao B, Wang X, Cui B (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 and data mining, pp 1984–1992
28. Zhang W, Mao J, Cao Y, Xu C (2020) Multiplex graph neural networks for multi-behavior recommendation. In: Proceedings of the 29th ACM international conference on information & knowledge management, pp 2313–2316
29. Xia L, Xu Y, Huang C, Dai P, Bo L (2021) Graph meta network for multi-behavior recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 757–766
30. Wu Y, Xie R, Zhu Y, Ao X, Chen X, Zhang X, Zhuang F, Lin L, He Q (2022) Multi-view multi-behavior contrastive learning in recommendation. In: Database systems for advanced applications: 27th international conference, DASFAA 2022, Virtual Event, April 11–14, 2022, Proceedings, Part II. Springer, pp 166–182
31. Xuan H, Liu Y, Li B, Yin H (2023) Knowledge enhancement for contrastive multi-behavior recommendation. In: Proceedings of the sixteenth ACM international conference on web search and data mining, pp 195–203
32. Yang H, Chen H, Li L, Philip SY, Xu G (2021) Hyper meta-path contrastive learning for multi-behavior recommendation. In: 2021 IEEE international conference on data mining (ICDM). IEEE, pp 787–796
33. Xin X, Liu X, Wang H, Ren P, Chen Z, Lei J, Shi X, Luo H, Jose JM, de Rijke M (2023) Improving implicit feedback-based

recommendation through multi-behavior alignment. In: Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval, pp 932–941

34. Chen C, Zhang M, Zhang Y, Ma W, Liu Y, Ma S (2020) Efficient heterogeneous collaborative filtering without negative sampling for recommendation. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 19–26
35. Chen C, Ma W, Zhang M, Wang Z, He X, Wang C, Liu Y, Ma S (2021) Graph heterogeneous multi-relational recommendation. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, pp 3958–3966
36. Gan M, Xu G, Ma Y (2023) A multi-behavior recommendation method exploring the preference differences among various behaviors. Expert Syst Appl 228:120316
37. He Y, Wang Z, Cui P, Zou H, Zhang Y, Cui Q, Jiang Y (2022) Causpref: causal preference learning for out-of-distribution recommendation. In: Proceedings of the ACM web conference 2022, pp 410–421
38. Huang Junjie, Xie Ruobing, Cao Qi, Shen Huawei, Zhang Shaoliang, Xia Feng, Cheng Xueqi (2023) Negative can be positive: signed graph neural networks for recommendation. Inf Process Manag 60(4):103403
39. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. Stat 1050(20):10–48550
40. Xia L, Huang C, Xu Y, Dai P, Bo L (2022) Multi-behavior graph neural networks for recommender system. IEEE Trans Neural Netw Learn Syst
41. Yuan G, Yuan F, Li Y, Kong B, Li S, Chen L, Yang M, Yu C, Hu B, Li Z, Xu Y (2022) Tenrec: a large-scale multipurpose benchmark dataset for recommender systems. Adv Neural Inf Process Syst 35:11480–11493
42. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
43. Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. Adv Neural Inf Process Syst 30
44. Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, Grohe M (2019) Weisfeiler and leman go neural: higher-order graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 4602–4609
45. Ranjan E, Sanyal S, Talukdar P (2020) Asap: adaptive structure aware pooling for learning hierarchical graph representations. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 5470–5477