# MuLe: Multi-Grained Graph Learning for Multi-Behavior Recommendation

Seunghan Lee*
Dept. of CSAI,
Jeonbuk Nat'l Univ.
Jeonju, South Korea
juajua56@jbnu.ac.kr

Geonwoo Ko*
School of Software,
Soongsil University
Seoul, South Korea
geonwooko@soongsil.ac.kr

Hyun-Je Song†
Dept. of CSAI,
Jeonbuk Nat'l Univ.
Jeonju, South Korea
hyunje.song@jbnu.ac.kr

Jinhong Jung†
School of Software,
Soongsil University
Seoul, South Korea
jinhong@ssu.ac.kr

## Abstract

Multi-behavior recommender systems, rapidly advancing across various domains, utilize plentiful auxiliary interactions on a variety of user behaviors to enhance recommendations for the target behavior, such as purchases. While previous methods have made strides in leveraging such interactions with advanced machine learning methods, they still face challenges in adequately using multi-faceted relationships among behaviors and handling uncertain auxiliary interactions that could potentially lead to purchases or not. In this paper, we propose MuLe (Multi-Grained Graph Learning), a novel graph-based model designed to address these limitations. We design a multi-grained graph learning strategy to capture diverse aspects of behaviors, ranging from unified to specific, and then to target-related behavior interactions. To handle uncertain interactions, we use graph attention, weighting the importance of those interactions related to the target behavior. Afterward, we use an attention mechanism to effectively aggregate diverse behavior embeddings obtained from the multi-grained graph encoders. Extensive experiments show that MuLe significantly outperforms the state-of-the-art methods, achieving improvements of up to 44.6% in HR@10 and 52.9% in NDCG@10, respectively. Our code and datasets are available at https://github.com/geonwooko/MULE.

## CCS Concepts

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

## Keywords

Multi-behavior recommendation, Multi-grained graph learning, Target-guided denoising attention

**ACM Reference Format:**
Seunghan Lee, Geonwoo Ko, Hyun-Je Song, and Jinhong Jung. 2024. MuLe: Multi-Grained Graph Learning for Multi-Behavior Recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3627673.3679709

*Two first authors have contributed equally to this work.
†Co-corresponding authors



*CIKM '24, October 21–25, 2024, Boise, ID, USA*

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679709

## 1 Introduction

Recommender systems, indispensable for navigating the information overload, have significantly advanced over the past decades to enhance personalized user experiences across various domains such as streaming services, e-commerce, social media, content aggregation, etc. Collaborative filtering (CF) [29] suggests items to a user based on interactions of similar users, and most traditional CF-based methods exploit matrix factorization (MF) [10, 19, 28, 32] or graph-based models [8, 18, 21, 27]. However, those methods rely on a single behavior on items, leading to data sparsity and limited performance [11, 41]. In practice, user interactions involve multiple behaviors, including viewing, adding to cart, and purchasing [24]. To fully exploit such behaviors, *multi-behavior recommendation* has gained attention from data mining communities [6, 11, 25, 26, 40, 43, 45]. It aims to predict a target behavior, such as purchasing, by using other behaviors as auxiliary information (Figure 1a).

Many researchers have proposed innovative methods to effectively make recommendations using multi-behavior interactions by leveraging advanced machine learning methods, including neural MF [9], graph convolution network (GCN) [16], and transformer [34]. As the multi-behavior interactions can be modeled as a unified multigraph, GCN-based methods [11, 30, 39] learn node embeddings from the unified graph, and some [2, 42, 45] further adopt multi-task learning (MTL) to predict auxiliary behaviors as well. Inspired by the cascading sequence such as viewing, adding to cart, and purchasing items, some studies use the sequence to model embeddings within the framework of either MF [6, 7, 36] or GCN [3, 43]. Recently, several researches [25, 26, 44] have pointed out that multi-behavior interactions are heavily skewed, leading them to separately learn embeddings on each behavior-specific graph to reduce bias from predominant interactions. Especially, Meng et al. [26] further refine the behavior embeddings into unique and shared components via a projection technique [1] to fully exploit supervisory signals from non-overlapping interactions, such as viewing exclusively without purchasing.

However, the performance of existing methods for multi-behavior recommendation still remains limited due to the following:

- **Inadequate use of multi-faceted relationships.** When dealing with multi-behavior interactions, diverse aspects can be considered for learning embeddings of users and items. For instance, a unified graph can reveal global user-item relationships, while specific behavior graphs capture distinct behavioral signals [44]. Particularly, as pointed in [26], the relationships between different behaviors require detailed consideration, e.g., "`only view`" and "`view and buy`" interactions carry distinct implications. However, the previous methods are limited in their

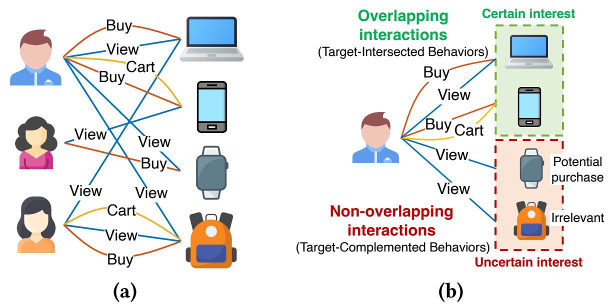



**Figure 1: Examples of (a) user-item interactions with multiple behaviors and (b) certain and uncertain interest related to the target behavior $t$ (e.g., buy).**

ability to comprehensively integrate such multi-faceted relationships when encoding embeddings.

- **Uncertain interactions of auxiliary behaviors.** Note auxiliary interactions can be uncertain w.r.t. the target behavior (i.e., buy). As shown in Figure 1b, if a user views an item and then makes a purchase, these clearly indicate their buying intent. However, if a user only views an item, this interaction may lead to a purchase or may not. When predicting the user's target behavior, such an interaction can be helpful if it conveys potential interest; otherwise, it is not informative for learning and acts as noise. Furthermore, as interactions are skewed toward auxiliary behaviors [25], substantial noises from early stages can accumulate through a cascading chain, adversely affecting the learning process.

In this paper, we propose MuLe (**Mu**lti-grained Graph-based **Le**arning), a novel method designed to effectively address the aforementioned limitations in multi-behavior recommendation. For the first issue, we introduce a multi-grained graph learning strategy that transitions from a unified graph to behavior-specific graphs (BSGs), and then to target-related behavior graphs (TRBGs) in order to effectively capture multi-faceted relationships among behaviors. Especially, we distinguish between overlapping (e.g., `view` and `buy`) and non-overlapping (e.g., `only view`) interactions between auxiliary and target behaviors to form TRBGs, enabling MuLe to capture both definite and potential interests from each corresponding TRBG. For the second issue, we propose Target-Guided Denoising Attention (TDA) to denoise uncertain auxiliary interactions by distinctly considering between potential interests and irrelevant ones when learning embeddings. To achieve this, it uses target behavior information of users or items to assign high attention weights to potential interests and low weights to irrelevant ones. Following theses, we design Multi-Grained Aggregator (MGA) to effectively integrate multi-grained behavior embeddings using attention mechanisms, leading to the final embeddings w.r.t. the target behavior.

Our main contributions are summarized as follows:

- We propose MuLe, a novel graph learning method for multi-behavior recommendation. Our method employs a multi-grained graph learning strategy to capture multi-faceted relationships of behaviors, ranging from unified to specific and then to target-related behaviors.
- We design TDA to effectively identify potential interest or denoise uncertain auxiliary interactions w.r.t. the target behavior. Moreover, we develop MGA to integrate multi-grained behavior embeddings using attention mechanisms.
- Experiments on three real-world datasets show that MuLe outperforms state-of-the-art methods, with improvements of up to 44.6% in HR@10 and 52.9% in NDCG@10. Additional experiments further highlight the effectiveness of MuLe, indicating the importance of considering multi-grained graphs and effectively handling uncertain auxiliary interactions.

## 2 Related Work

In this section, we review previous methods for recommendations based on user behavior.

**Single-behavior recommendation.** It recommends items based on a single type of user behavior, e.g., ratings [19], clicks [28], trusts [13, 17], likes [12, 31], etc. Collaborative filtering (CF) [29], a popular approach, predicts a user's preferences based on those of similar users. For this, early models exploit matrix factorization (MF) [10, 19, 32], graph-based similarity [18] or ranking based on random walks [5, 14, 27]. To model positive interactions to score higher than non-observed ones, Bayesian personalized ranking (BPR) loss is proposed, combined with MF [28] or deep neural networks [9]. Recently, graph convolution networks (GCN) [16] have gained attention in CF. As a leading GCN method, LightGCN [8] simplifies the GCN design, removing non-linear activations and feature transformations. Observing that it is hard to train noisy interactions in the implicit feedback, Wang et al. [37] suggested adaptively truncating (or denoising) harder samples to facilitate effective learning. However, the recommendation power of a single-type behavior is limited by the data sparsity issue, which leads to insufficient learning ability [11, 41].

**Multi-behavior recommendation.** To address the issue, the multi-behavior recommendation aims to use auxiliary behaviors (e.g., viewing) to predict a target behavior (e.g., purchasing). In the early stage, researchers focused on exploiting MF [20, 33] or developing effective schemes for negative sampling [4, 24]. More recently, various methods have been proposed that leverage advanced learning methods, including neural network, GCN, and transformer [40, 41]. Most methods can be categorized based on how they handle the behaviors: 1) unified, 2) cascading, and 3) parallel approaches.

The *unified approach* learns behavior embeddings on a single relational multigraph, representing multi-behavior interactions on a unified graph. RGCN [11, 30], MBGCN [11], and GNMR [39] perform their own GCN layers on the unified graph. To further enhance performance, MGNN [45], GHCF [2], and MBGMN [42] integrate a multi-task learning (MTL) framework, predicting auxiliary interactions as well, into their graph learning processes. CML [38] adopts contrastive learning coupled with behavior-aware GCNs to capture knowledge across different behaviors.

The *cascading approach* designs models exploiting the natural sequence of user behaviors, e.g., `view` → `cart` → buy. ChainRec [36] extends MF and optimizes scoring functions by considering the cascading dependency. NMTR [6, 7] extends the MTL technique with neural MF to predict interactions of each behavior in the cascading order. CRGCN [43] introduces a cascading GCN model with residual connections, sequentially propagating behavioral embeddings in the MTL framework. MB-CGCN [3] encodes nodes for

**Table 1: Frequently-used symbols.**

| Symbol | Description |
|---|---|
| $\mathcal{U}$ | set of users, where $M = \lvert\mathcal{U}\rvert$ |
| $\mathcal{I}$ | set of items, where $N = \lvert\mathcal{I}\rvert$ |
| $t$ | target behavior (e.g., buy) |
| $\mathcal{B}$ | set of behaviors |
| $\mathcal{B}_{\mathsf{aux}}$ | set of auxiliary behaviors, where $\mathcal{B}_{\mathsf{aux}} = \mathcal{B} \setminus \{t\}$ |
| $\mathcal{B}_{\mathtt{inter}}$ | set of target-intersected behaviors |
| $\mathcal{B}_{\mathtt{comp}}$ | set of target-complemented behaviors |
| $\mathcal{E}_b$ | set of user-item interactions on behavior $b$ |
| $\mathcal{G}_b = (\mathcal{V}, \mathcal{E}_b)$ | user-item graph on behavior $b$, where $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ |
| $n$ | number of nodes in $\mathcal{G}_b$, i.e., $n = \lvert\mathcal{V}\rvert = M + N$ |
| $\mathbf{A}_b \in \mathbb{R}^{n\times n}$ | adjacency matrix of $\mathcal{G}_b$ in Equation (1) |
| $d$ | embedding dimension |
| $\mathbf{E}_b \in \mathbb{R}^{n\times d}$ | embeddings of users and items w.r.t behavior $b$ |
| $\mathbf{Z}_t \in \mathbb{R}^{n\times d}$ | final embeddings of users and items w.r.t. $t$ |
| $L_{\mathtt{light}}$ & $L_{\mathtt{tda}}$ | numbers of LightGCN and TDA layers, resp. |

each behavior using LightGCNs, and transfers behavior features along a cascading chain through feature transformations.

The *parallel approach* separately learns embeddings on each behavior graph to consider the imbalance in behavior interactions (see Table 2), aiming to reduce bias from dominant ones. MB-HGCN [44] initializes GCNs on individual graphs using global embeddings from a unified graph, allowing the MTL module to effectively use the refined behavior embeddings. HPMR [26] utilizes behavior embeddings obtained in parallel, and disentangles them into unique and shared components through projection [1], thereby mitigating the MTL's negative transfer [23]. Meng et al. [25] claimed the behavior imbalance exacerbates the negative transfer through the cascading scheme, and proposed PKEF, which carefully merges the parallel approach with cascading using the projection-based distillation.

However, as discussed in Section 1, the existing methods are limited to comprehensively capture multi-faceted relationships for learning embeddings, disregarding distinct implications of different behaviors. Furthermore, target-related auxiliary interactions, such as only viewing items without purchasing, can introduce noise into learning target behavior predictions due to their uncertainty. On the other hand, our MuLe effectively addresses these limitations through multi-grained graph learning and target-guided denoising attention, showing significant improvements compared to the existing methods (refer to Table 3).

# 3 Preliminaries

We introduce preliminaries on basic concepts, and LightGCN, and the definition of the problem addressed in this paper.

## 3.1 Concepts and Notations

We describe the basic notations frequently used in this paper, with the related symbols summarized in Table 1.

**User-item interactions.** Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items, where $M = |\mathcal{U}|$ and $N = |\mathcal{I}|$ are the numbers of users and items, respectively. Suppose $\mathcal{B} = \{\mathtt{view}, \mathtt{cart}, \cdots, \mathtt{buy}\}$ is the set of behaviors, and let $t$ denote the target behavior (e.g., buy). Then, $\mathcal{B}_{\mathsf{aux}}$ defines the set of auxiliary behaviors, i.e., $\mathcal{B}_{\mathsf{aux}} = \mathcal{B} \setminus \{t\}$. If user $u$ has interacted with item $i$ on behavior $b \in \mathcal{B}$, a pair $(u, i)$ is included in $\mathcal{E}_b$, the set of user-item interactions on $b$.

**User-item graphs.** A user-item graph on behavior $b$ is denoted by $\mathcal{G}_b = (\mathcal{V}, \mathcal{E}_b)$ where $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$, and $\mathbf{R}_b \in \mathbb{R}^{M\times N}$ is the user-item interaction matrix where $\mathbf{R}_b(u, i)$ is 1 if $(u, i) \in \mathcal{E}_b$; otherwise, 0. Let $\mathbf{A}_b$ and $\tilde{\mathbf{A}}_b$ denote the adjacency matrix of $\mathcal{G}_b$ and its symmetric normalized matrix, respectively, which are defined as follows:

$$\mathbf{A}_b = \begin{bmatrix} \mathbf{0} & \mathbf{R}_b \\ \mathbf{R}_b^\top & \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \tilde{\mathbf{A}}_b = \mathbf{D}^{-1/2}\mathbf{A}_b\mathbf{D}^{-1/2}, \tag{1}$$

where $\mathbf{A}_b \in \mathbb{R}^{n\times n}$, $n = M + N$, and $\mathbf{D}$ is a diagonal degree matrix, in which $\mathbf{D}(i, i)$ is the degree of the $i$-th node in the graph.

## 3.2 LightGCN

As described in Section 2, LightGCN [8] employs simplified GCNs on a graph of single-behavior interactions. As its effectiveness was empirically verified for graph-based collaborative filtering, it has been utilized in various methods [3, 25, 43, 44] including multi-behavior recommendation. In this work, we adopt a variant of LightGCN with normalization [44] as a graph encoder to learn user and item embeddings. Given a bipartite graph represented by $\mathbf{A}$ and an initial embedding matrix $\mathbf{E}^{(0)}$, LightGCN with $L$ layers is defined as follows:

$$\mathbf{E} \leftarrow \mathtt{LightGCN}\big(\mathbf{A}, \mathbf{E}^{(0)}\big) := \mathbf{E}^{(0)} + \sum_{l=1}^{L} \frac{\mathbf{E}^{(l)}}{l} \tag{2}$$

where $\mathbf{E}^{(l)} \in \mathbb{R}^{n\times d}$ is the embedding matrix, and $d$ is the dimension. Note $\mathbf{E}^{(l)} \leftarrow \mathtt{normalize}(\tilde{\mathbf{A}}\mathbf{E}^{(l-1)})$, the result of the message-passing on the graph at the $l$-th layer, where $\leftarrow$ is the assignment operator, $\tilde{\mathbf{A}}$ is the symmetric normalized matrix similarly defined in Equation (1), and $\mathtt{normalize}(\mathbf{M})$ does row-wise $L_2$-normalization of $\mathbf{M}$, i.e., each $i$-th row of $\mathbf{M}$ is normalized as $\mathbf{M}(i)/\|\mathbf{M}(i)\|_2$. In the model, the only trainable parameters are in $\mathbf{E}^{(0)}$.

## 3.3 Problem Definition

We define the problem of multi-behavior recommendation as:

Problem 1 (Multi-behavior Recommendation).

- ***Input***: *the sets $\mathcal{U}$ and $\mathcal{I}$ of users and items, and the set of multi-behavior interactions, i.e., $\mathcal{E} = \{\mathcal{E}_b \mid b \in \mathcal{B}\}$,*
- ***Output***: *a ranking score $\hat{y}_t(u, i)$ between user $u$ and item $i$, meaning the likelihood of user $u$ performing the target behavior $t$ (e.g.,* buy*) for item $i$.*

For each querying user, the recommendation list is generated by sorting items in descending order based on their similarity scores.

# 4 Proposed Method

In this section, we propose MuLe, a novel and effective graph learning method for multi-behavior recommendation, where the overall architecture of MuLe is depicted in Figure 2.

## 4.1 Overview

The technical challenges for effective multi-behavior recommendation are summarized as follows:

C1. **Multi-faceted relationships.** As discussed in Section 1, multi-behavior interactions involves various aspects, resulting in diverse graphs. How can we effectively learn user and item embeddings across these different levels of graphs?

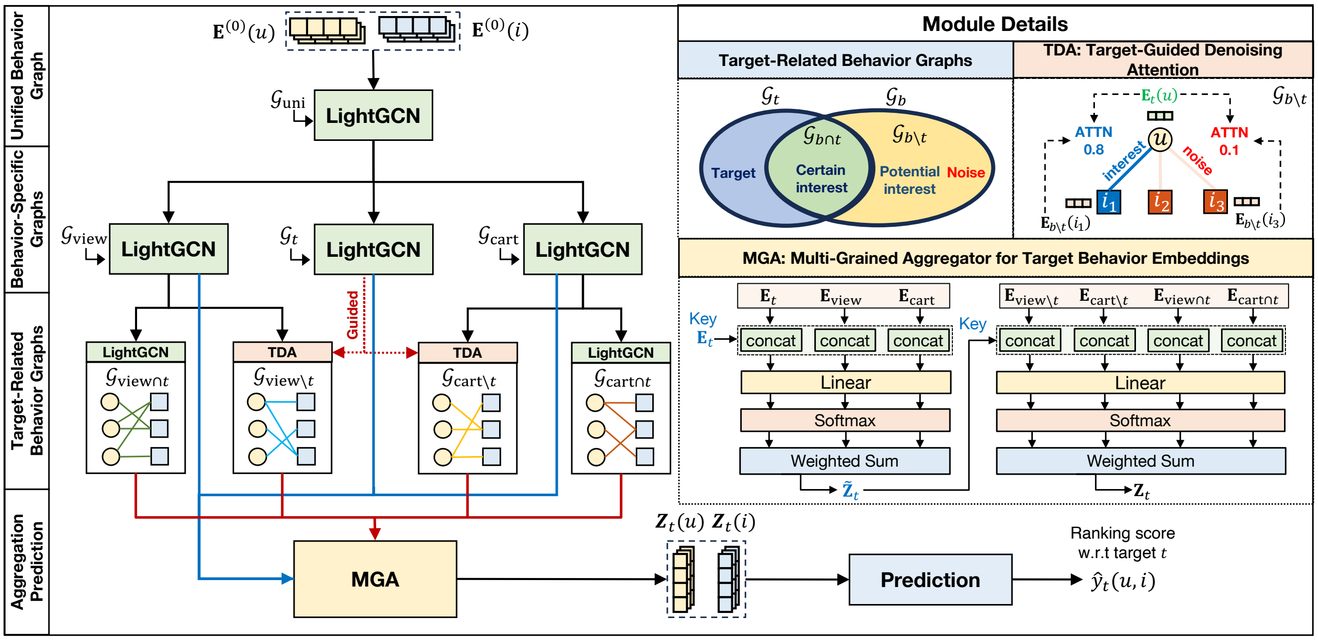



**Figure 2: Overall architecture of MuLe where $t$ denotes the target behavior (e.g., buy). Our method 1) comprehensively considers multi-faceted relationships among behaviors, 2) effectively denoises uncertain interactions using TDAs, and 3) integrates multi-grained behavior embeddings using an MGA, resulting in the final embeddings $\mathbf{Z}_t$.**

C2. **Uncertain auxiliary interactions.** As mentioned in Section 1, auxiliary interactions can be uncertain, as they may or may not intend genuine interest. How can we handle these interactions to facilitate effective learning?

C3. **Diverse behavior embeddings.** Given that different graphs from diverse aspects would yield distinct behavior embeddings, how can we effectively integrate those embeddings for better recommendation?

We propose the following ideas to handle the challenges above:

A1. **Multi-grained graph encoders.** We employ a multi-grained graph learning strategy that learns various behavior embeddings from a unified graph to behavior-specific graphs, and finally to target-related behavior graphs.

A2. **Target-guided denoising attention encoder.** To handle uncertain interactions, we devise a graph attention module, guided by target behavior embeddings, that assigns high scores to potential interests and low scores to irrelevant ones.

A3. **Multi-grained aggregator.** We design an attention-based aggregator that integrates diverse behavior embeddings relative to the target behavior, resulting in the final embeddings.

## 4.2 Multi-Grained Graph Encoders

To fully use multi-faceted relationships of behaviors, we construct multi-grained graphs from multi-behavior interactions, ranging from coarse to fine-grained. Afterwards, we learn user and item embeddings using graph encoders from a unified behavior to specific behaviors, then to target-related behaviors, as shown in Figure 2.

*4.2.1 Unified and Behavior-Specific Graphs.* As suggested by [44], embeddings generated from a unified graph representing merged interactions can serve as good initial embeddings when learning behavior-specific embeddings. Building upon this approach, we also employ a similar strategy to obtain effective warm-start embeddings, which are suitable for initializing subsequent encoders on target-related behavior graphs.

**Unified behavior graph encoder.** We first construct a unified behavior graph by merging interactions. Let $\mathcal{G}_{\texttt{uni}}$ denote the unified behavior graph. Then, $\mathcal{G}_{\texttt{uni}} = (\mathcal{V}, \mathcal{E}_{\texttt{uni}})$ where $\mathcal{E}_{\texttt{uni}} \coloneqq \bigcup_{b\in\mathcal{B}} \mathcal{E}_b$, the merged interactions of all behaviors. Given an initial learnable embedding matrix $\mathbf{E}^{(0)} \in \mathbb{R}^{n\times d}$, MuLe employs LightGCN to learn $\mathcal{G}_{\texttt{uni}}$ as follows:

$$\mathbf{E}_{\texttt{uni}} \leftarrow \texttt{LightGCN}\big(\mathbf{A}_{\texttt{uni}}, \mathbf{E}^{(0)}\big), \tag{3}$$

where $\mathbf{A}_{\texttt{uni}}$ is the adjacency matrix of $\mathcal{G}_{\texttt{uni}}$, and $\mathbf{E}_{\texttt{uni}}$ contains *unified* embeddings of users and items.

**Behavior-specific graph encoder.** This encoder captures distinct graph signals corresponding to each behavior. As discussed in [25], this could help mitigate bias from skewed interactions, particularly since multi-behavior interactions tend to be imbalanced towards auxiliary behaviors (refer to Table 2). Let $\mathcal{G}_b$ be the graph of specific behavior $b \in \mathcal{B}$, which is defined as $\mathcal{G}_b = (\mathcal{V}, \mathcal{E}_b)$. The adjacency matrix of $\mathcal{G}_b$ is denoted by $\mathbf{A}_b$. Initialized by $\mathbf{E}_{\texttt{uni}}$, the learning process for $\mathcal{G}_b$ is as follows:

$$\mathbf{E}_b \leftarrow \texttt{LightGCN}\big(\mathbf{A}_b, \mathbf{E}_{\texttt{uni}}\big), \tag{4}$$

where $\mathbf{E}_b$ has *behavior-specific* embeddings of users and items on behavior $b$. As shown in Figure 2, MuLe repeatedly calculates the embeddings for each behavior, e.g., $\mathbf{E}_{\texttt{view}}$, $\mathbf{E}_{\texttt{cart}}$, $\mathbf{E}_{\texttt{buy}}$ (or $\mathbf{E}_t$), etc.

*4.2.2 Target-related Behavior Graphs.* In this step, MuLe aims to comprehensively learn auxiliary behavior interactions related to the target behavior (e.g., buy). As discussed in Section 1, the relationships between auxiliary and target behaviors should be considered differently when learning a user's target behavior. For example, if the user both views and buys an item, the view interaction strongly conveys the user's interest in the purchase. However, if the user only views the item, it may or may not lead to a purchase. The former is referred to as *target-intersected* behavior (e.g., $\texttt{view} \cap \texttt{buy}$), while the latter is called *target-complemented* behavior (e.g., $\texttt{view} \setminus \texttt{buy}$). These are formally defined as follows:

Definition 1 (Target-related Behaviors). *Suppose $\mathcal{B}_{\texttt{aux}}$ is the set of auxiliary behaviors, and $t$ is the target behavior. Then, $\mathcal{B}_{\texttt{inter}} = \{b \cap t \mid b \in \mathcal{B}_{\texttt{aux}}\}$ is the set of target-intersected behaviors, and $\mathcal{B}_{\texttt{comp}} = \{b \setminus t \mid b \in \mathcal{B}_{\texttt{aux}}\}$ is the set of target-complemented behaviors.*[1]

For each $b \cap t \in \mathcal{B}_{\texttt{inter}}$, the set of its interactions is represented as $\mathcal{E}_{b \cap t} := \{(u,i) \mid (u,i) \in \mathcal{E}_b \wedge (u,i) \in \mathcal{E}_t\}$. On the other hand, for each $b \setminus t \in \mathcal{B}_{\texttt{comp}}$, the set of its interactions is defined as $\mathcal{E}_{b \setminus t} := \{(u,i) \mid (u,i) \in \mathcal{E}_b \wedge (u,i) \notin \mathcal{E}_t\}$. As interactions in $\mathcal{E}_{b \cap t}$ and $\mathcal{E}_{b \setminus t}$ have different implications, we handle them differently in the learning process.

**Target-intersected behavior graph encoder.** An interaction of $b \cap t$ indicates that the auxiliary behavior $b$ has a positive influence on the target behavior $t$. Thus, we utilize the interactions in $\mathcal{E}_{b \cap t}$ to refine behavior embeddings $\mathbf{E}_b$ previously obtained from $\mathcal{G}_b$ on specific behavior $b$, which contains noisy interactions. Let $\mathcal{G}_{b \cap t} = (\mathcal{V}, \mathcal{E}_{b \cap t})$ denote a graph of target-intersected interactions. For each $b \cap t \in \mathcal{B}_{\texttt{comp}}$, MuLe learns the graph with $\mathbf{E}_b$ as follows:

$$\mathbf{E}_{b \cap t} \leftarrow \texttt{LightGCN}(\mathbf{A}_{b \cap t}, \mathbf{E}_b), \tag{5}$$

where $\mathbf{A}_{b \cap t}$ is the adjacency matrix of $\mathcal{G}_{b \cap t}$.

## 4.3 Target-Guided Denoising Attention Encoder

As mentioned in Section 4.2.2, an interaction of target-complemented behavior $b \setminus t$ can be uncertain because the auxiliary behavior $b$ may or may not result in the target behavior $t$. Therefore, it is important to distinguish such an interaction as either informative or non-informative, when modeling user and item embeddings.

For this purpose, we propose *target-guided denoising attention* (TDA) based on graph attention network [35] on $\mathcal{G}_{b \setminus t} = (\mathcal{V}, \mathcal{E}_{b \setminus t})$. Specifically, in the graph $\mathcal{G}_{b \setminus t}$, we aim to assign high weights to interactions that may contain potential interest, and low weights to those that are irrelevant or noisy. This enables aggregated embeddings with those weights to be further refined w.r.t. the target $t$. As a guide for this, we exploit target behavior embeddings $\mathbf{E}_t$ from $\mathcal{G}_t$, which encapsulate genuine interest related to $t$ of users and items.

Given the graph $\mathcal{G}_{b \setminus t}$ with embeddings $\mathbf{E}_t$, TDA with $L$ layers learns embeddings $\mathbf{E}_{b \setminus t}$ as follows:

$$\mathbf{E}_{b \setminus t} \leftarrow \texttt{TDA}(\mathcal{G}_{b \setminus t}, \mathbf{E}_t, \mathbf{E}^{(0)}_{b \setminus t}) := \mathbf{E}^{(0)}_{b \setminus t} + \sum_{l=1}^{L} \frac{\mathbf{E}^{(l)}_{b \setminus t}}{l}, \tag{6}$$

where $\mathbf{E}^{(0)}_{b \setminus t}$ is an initial embedding matrix. The embedding matrix $\mathbf{E}^{(l)}_{b \setminus t}$ at the $l$-th layer is updated as follows:

$$\mathbf{E}^{(l)}_{b \setminus t} \leftarrow \texttt{normalize}(\mathcal{A}^{(l)}_{b \setminus t} \mathbf{E}^{(l-1)}_{b \setminus t}), \tag{7}$$

which indicates the neighborhood aggregation on the adjacent attention matrix $\mathcal{A}^{(l)}_{b \setminus t} \in \mathbb{R}^{n \times n}$ at the $l$-th layer, which is defined as the following. For each interaction $(u,i) \in \mathcal{E}_{b \setminus t}$, the attention probability (or weight) from $u$ to $i$ is obtained as follows:

$$\begin{aligned}\mathcal{A}^{(l)}_{b \setminus t}(u,i) &\leftarrow \texttt{softmax}_i\left(\{a^{(l)}_{b \setminus t}(u,j) \mid j \in \mathcal{N}_{b \setminus t}(u)\}\right) \\ &:= \frac{\texttt{exp}\left(a^{(l)}_{b \setminus t}(u,i)\right)}{\sum_{j \in \mathcal{N}_{b \setminus t}(u)} \texttt{exp}\left(a^{(l)}_{b \setminus t}(u,j)\right)},\end{aligned} \tag{8}$$

where $a^{(l)}_{b \setminus t}(u,i)$ is the attention score from $u$ to $i$, $\mathcal{N}_{b \setminus t}(u)$ is the set of neighboring nodes of $u$ in $\mathcal{G}_{b \setminus t}$, and $\texttt{exp}(\cdot)$ is the exponential function. For denoising, the score should reflect the relevance between a user (or item) in the target context (e.g., buy) and an item (or user) in the auxiliary context complementary to the target (e.g., $\texttt{view} \setminus \texttt{buy}$). Thus, the score is calculated as follows:

$$a^{(l)}_{b \setminus t}(u,i) \leftarrow \left\langle \mathbf{E}_t(u), \mathbf{E}^{(l)}_{b \setminus t}(i) \right\rangle, \tag{9}$$

where $\mathbf{E}_t(u)$ is the target embedding of user $u$, $\mathbf{E}^{(l)}_{b \setminus t}(i)$ is the target-complemented embedding of item $i$, and $\langle \cdot, \cdot \rangle$ is the inner product of two vectors.

The attention probability $\mathcal{A}^{(l)}_{b \setminus t}(i,u)$ from $i$ to $u$ is calculated in a manner similar to that described above, using $\mathbf{E}_t(i)$ and $\mathbf{E}^{(l)}_{b \setminus t}(u)$. For $(u,i) \notin \mathcal{E}_{b \setminus t}$, $\mathcal{A}^{(l)}_{b \setminus t}(u,i) = \mathcal{A}^{(l)}_{b \setminus t}(i,u) = 0$, indicating that the attention matrix is sparse, and its graph structure mirrors that of $\mathcal{G}_{b \setminus t}$. We initialize $\mathbf{E}^{(0)}_{b \setminus t}$ with $\mathbf{E}_b$, as it provides good warm-start embeddings derived from plentiful interactions at coarser levels. Note that our TDA design is well-suited for its purpose, showing its effectiveness through comparisons with alternative GNNs (Figure 4), and attention analyses (Section 5.5.1).

## 4.4 Multi-Grained Aggregator for Target Behavior Embeddings

After obtaining various embeddings through the multi-grained graph encoders, we aim to effectively aggregate them into the final embeddings $\mathbf{Z}_t$ related to the target behavior $t$.

For this purpose, we propose *multi-grained aggregator* (MGA) that effectively aggregates those embeddings based on attention mechanisms. MGA first generates intermediate target embeddings $\tilde{\mathbf{Z}}_t$ that capture the context of behavior-specific embeddings $\mathbf{E}_b$, serving as a warm start for the next step, similar to the multi-grained graph learning. For this, it computes the attention probability between the key $\mathbf{E}_t(v)$ and the query $\mathbf{E}_b(v)$ for each $v \in \mathcal{V}$, weighting their relationship. For each $b \in \mathcal{B}$, let $\mathbf{s}_b$ be an attention score between them, which is calculated as follows:

$$\mathbf{s}_b \leftarrow \texttt{Linear}_1\left(\mathbf{E}_t(v) \parallel \mathbf{E}_b(v)\right). \tag{10}$$

Note that we have $|\mathcal{B}|$ scores of $\mathbf{s}_b$ for all $b$ in $\mathcal{B}$. Then, its attention probability (or weight) $\boldsymbol{\alpha}_b$ is obtained by the softmax function as:

$$\boldsymbol{\alpha}_b \leftarrow \texttt{softmax}_b\left(\{\mathbf{s}_b \mid b \in \mathcal{B}\}\right) := \frac{\texttt{exp}\,(\mathbf{s}_b)}{\sum_{b \in \mathcal{B}} \texttt{exp}\,(\mathbf{s}_b)}. \tag{11}$$

Then, $\tilde{\mathbf{Z}}_t(v) \leftarrow \sum_{b \in \mathcal{B}} \boldsymbol{\alpha}_b \mathbf{E}_b(v)$. Afterwards, we focus on the relationship between target-related behavior embeddings and the above $\tilde{\mathbf{Z}}_t$. Suppose $\mathcal{B}' := \mathcal{B}_{\texttt{inter}} \cup \mathcal{B}_{\texttt{comp}}$ is the set of target-related behaviors. For $b' \in \mathcal{B}'$, the attention score $\mathbf{s}_{b'}$ between the key $\tilde{\mathbf{Z}}_t(v)$ and the query $\mathbf{E}_{b'}(v)$ is obtained as follows:

$$\mathbf{s}_{b'} \leftarrow \texttt{Linear}_2\left(\tilde{\mathbf{Z}}_t(v) \parallel \mathbf{E}_{b'}(v)\right). \tag{12}$$

[1] Although $b$ and $t$ do not represent sets, we use set operations to convey the concepts of intersection and difference between the interactions.

Then, its attention probability $\boldsymbol{\alpha}_{b'}$ is computed by the softmax function among $|\mathcal{B}'|$ scores of $\mathbf{s}_{b'}$, similarly to Equation (11). For all $v \in \mathcal{V}$, the final embedding $\mathbf{Z}_t(v)$ is aggregated as follows:

$$\mathbf{Z}_t(v) \leftarrow \sum_{b' \in \mathcal{B}'} \boldsymbol{\alpha}_{b'} \mathbf{E}_{b'}(v). \tag{13}$$

Note that MGA outputs $\mathbf{Z}_t(u)$ and $\mathbf{Z}_t(i)$ for all users and items, as $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$. Each linear layer has a weight matrix $\mathbf{W} \in \mathbb{R}^{2d \times 1}$ and a bias term $\mathbf{b} \in \mathbb{R}^1$. As shown in Figure 5, this approach is empirically more effective compared to simple methods such as averaging all embeddings, or using an MLP.

## 4.5 Optimization for MuLe

Using the final embeddings $\mathbf{Z}_t$, MuLe computes the score $\hat{y}_t(u,i)$ between user $u$ and item $i$ w.r.t. the target behavior $t$ as follows:

$$\hat{y}_t(u,i) \leftarrow \langle \mathbf{Z}_t(u), \mathbf{Z}_t(i) \rangle, \tag{14}$$

where the score is used for Problem 1. In recommendation tasks using implicit feedback, ranking items is more effective than prediction [28], leading to the standard use of a pairwise learning strategy for model optimization [8, 11, 22]. Following previous studies [3, 25, 43, 44], we apply the Bayesian personalized ranking (BPR) loss [28], which assumes observed items should receive higher scores than unobserved ones. Given the set $\mathcal{E}_t$ of target behavior interactions, the loss function $\mathcal{L}_t$ is as follows:

$$\mathcal{L}_t(\mathcal{E}_t) \coloneqq \sum_{(u,i,j) \in O_t} -\ln \sigma\big(\hat{y}_t(u,i) - \hat{y}_t(u,j)\big) + \lambda \|\Theta\|_2^2, \tag{15}$$

where $\sigma(\cdot)$ is the sigmoid function, $\lambda$ is a hyperparameter for regularization, and $\Theta$ is the set of model parameters of MuLe. In the loss function, $O_t$ is the set of triplets $(u,i,j)$ where $(u,i) \in \mathcal{E}_t$ represents a positive sample, and $(u,j) \notin \mathcal{E}_t$ is a negative sample which is randomly selected per epoch.

## 4.6 Complexity Analysis

In this section, we analyze the computational complexities of MuLe in terms of time and space.

Theorem 1 (Time Complexity for Forward Pass). *The time complexity of MuLe is $O\big(BLd(m+n)\big)$, where $m$ is the total number of interactions, $n = M + N$ is the number of nodes, $L$ is the number of layers, $d$ is embedding dimension, and $B$ is the number of behaviors.*

Proof. Every component utilizing LightGCN incurs a time complexity of $O(Ld(m+n))$ because the adjacency matrix of each behavior graph, which has $O(m)$ non-zeros, undergoes $L$ sparse matrix multiplications, each taking $O(Ldm)$. The normalized sum adds $O(Ldn)$ to it. TDA also requires $O(Ld(m+n))$ time, but it additionally spends $O(dm)$ time to compute each attention matrix. Since MuLe uses $O(B)$ graph encoders, it takes $O(BLd(m+n))$ time. MGA operates in $O(Bdn)$ time, as it spends $O(Bd)$ time per node $v$ for $\mathcal{B}'$, the set of target-related behaviors, where $|\mathcal{B}'| = O(B)$. Putting everything together, we prove the theorem. □

Theorem 2 (Space Complexity on Model Parameters). *The space complexity for the learnable parameters of MuLe is $O\big(dn\big)$.*

Proof. The initial embedding matrix $\mathbf{E}^{(0)}$ requires $O(dn)$ space. Note that the graph encoders of MuLe do not require additional model parameters. MGA includes two linear layers where each linear layer uses $O(d)$ space for a weight matrix and a bias. Hence, MuLe requires a total of $O(dn)$ space for its model parameters. □

**Table 2: Data statistics of multi-behavior interactions. The percentage of each auxiliary behavior is the ratio of target-intersected interactions (e.g., view & buy).**

| Dataset | Users | Items | Views | Collects | Carts | Buys |
|---|---|---|---|---|---|---|
| Taobao | 15,449 | 11,953 | 873,954 ( 9%) | - | 195,476 (10%) | 92,180 |
| Tmall | 41,738 | 11,953 | 1,813,498 (12%) | 221,514 (12%) | 1,996 (15%) | 255,586 |
| Jdata | 93,334 | 24,624 | 1,681,430 (16%) | 45,613 (43%) | 49,891 (57%) | 321,883 |

The above theorems indicate that, assuming the hyperparameters are constant and behaviors are of fixed size, MuLe requires $O(m+n)$ time and uses $O(n)$ model parameters. In other words, MuLe is linearly scalable w.r.t. the total number of interactions and the number of nodes (i.e., users and items), respectively. Note that MuLe lies within the same complexities as other GNN-based methods [3, 25, 26, 36, 43, 44], and MuLe utilizes a few additional model parameters of $O(d)$.

# 5 Experiments

In this section, we conducted experiments to address the following questions:

Q1. **Recommendation performance.** How effective is MuLe for multi-behavior recommendation compared to its competitors?
Q2. **Ablation study.** How does each module of MuLe affect its recommendation performance?
Q3. **Parameter sensitivity.** How does the number of layers in the graph encoders of MuLe affect the performance?
Q4. **Attention analysis.** What insights do the TDA and MGA modules of MuLe provide, respectively?
Q5. **Case study.** How does the TDA of MuLe differentiate target-complemented interactions for specific users?

## 5.1 Experimental Settings

In this section, we explain the setup for our experiments on multi-behavior recommendation.

**Machine and Implementation.** We used a workstation with AMD 5955WX and RTX 4090 (24GB VRAM). Our method MuLe was implemented using Pytorch 2.0 in Python 3.9.

**Datasets.** Following recent studies [25, 43], we performed experiments on three publicly available real-world benchmark datasets: Taobao [25], Tmall [43], and JData [43]. These are sourced from e-commerce platforms in China. Tmall and JData have four behavior types: *view*, *add-to-collect*, *add-to-cart*, and *buy*, while Taobao, apart from add-to-collect, consists of three types. We followed the approach of previous studies [7, 11, 25, 43] for the datasets, where duplicate interactions were preprocessed by retaining only the earliest occurrence for each behavior. The detailed statistics of the datasets are summarized in Table 2.

**Competitors.** We compared MuLe with state-of-the-art models for multi-behavior recommendation to validate its effectiveness. For single-behavior models, we examined MF-BPR [28], NeuMF [9], and LightGCN [8]. For multi-behavior models adopting a unified approach, we investigated RGCN [30], GNMR [39], and MBGCN [11]. Employing a cascading approach, NMTR [6, 7], CRGCN [43], and

**Table 3: Recommendation performance in terms of HR@10 and NDCG@10. The best result is in bold, and the second best is underlined. The "% diff" means the percentage improvement of the best performance over the second best.**

| Model | HR@10 | | | NDCG@10 | | |
|---|---|---|---|---|---|---|
| | Taobao | Tmall | Jdata | Taobao | Tmall | Jdata |
| MF-BPR | 0.0076 | 0.0230 | 0.1850 | 0.0036 | 0.0207 | 0.1238 |
| NeuMF | 0.0236 | 0.0124 | 0.2090 | 0.0128 | 0.0062 | 0.1410 |
| LightGCN | 0.0411 | 0.0393 | 0.2252 | 0.0240 | 0.0209 | 0.1436 |
| RGCN | 0.0215 | 0.0316 | 0.2406 | 0.0104 | 0.0157 | 0.1444 |
| GNMR | 0.0368 | 0.0393 | 0.3068 | 0.0216 | 0.0193 | 0.1581 |
| MBGCN | 0.0509 | 0.0549 | 0.2803 | 0.0294 | 0.0285 | 0.1572 |
| NMTR | 0.0282 | 0.0536 | 0.3142 | 0.0137 | 0.0286 | 0.1717 |
| CRGCN | 0.0855 | 0.0840 | 0.5001 | 0.0439 | 0.0442 | 0.2914 |
| MB-CGCN | 0.1233 | 0.0984 | 0.4349 | 0.0677 | 0.0558 | 0.2758 |
| HPMR | 0.1104 | 0.0956 | – | 0.0599 | 0.0515 | – |
| PKEF | <u>0.1385</u> | 0.1277 | 0.4334 | <u>0.0785</u> | 0.0721 | 0.2615 |
| MB-HGCN | 0.1299 | <u>0.1443</u> | <u>0.5406</u> | 0.0690 | <u>0.0769</u> | <u>0.3555</u> |
| **MuLe** | **0.1918** | **0.2112** | **0.5889** | **0.1103** | **0.1177** | **0.4061** |
| % diff | 38.5% | 44.6% | 10.3% | 40.5% | 52.9% | 25.4% |

–: it indicates out-of-memory errors during training.

MB-CGCN [3] were included. Furthermore, we compared MB-HGCN [44], HPMR [26], and PKEF [25], as parallel approaches. For brief descriptions of each competitor, refer to Section 2. For the competitors, we used their official open-source implementations.

**Training and evaluation protocol.** Following the broadly used leave-one-out setting [3, 7, 11, 42–44], the test set consists of the last interacted item and all uninteracted items for each user. The second most recently interacted item for each user forms the validation set for hyperparameter tuning, while the remaining positive items are used for training. In the evaluation phase, items within the test set for each user are ranked based on predicted scores by recommendation models, where its top-$k$ ranking quality is measured by HR@$k$ and NDCG@$k$ [7, 25, 26, 39, 42–44]. HR@$k$ measures how often relevant items, on average, appear in the recommendation for each user. NDCG@$k$ considers both relevance and order of relevant items in a ranking, averaged across all users.

**Hyperparameter tuning.** According to previous studies [3, 25], we set the final embedding dimension $d$ to 64, the batch size to $2^{10}$ and the number of epochs to 100 for all tested methods. For each metric, we conducted a grid search to validate hyperparameters using the validation set and recorded the test performance with the validated hyperparameters. The hyperparameters of MuLe are the numbers $L_{\texttt{light}}$ and $L_{\texttt{tda}}$ of LightGCN's layers and TDA's layers, where they were tuned in $\{0, 1, 2, 3\}$ and $\{0, 1, 2, 3, 4, 5, 6\}$, respectively. We used the Adam optimizer [15], where its learning rate $\eta$ was tuned in $\{10^{-4}, 5 \cdot 10^{-4}, 10^{-3}\}$. The regularizer $\lambda$ in Equation (15) was searched in $\{0, 10^{-5}\}$. For other methods, we followed the hyperparameter ranges described in the corresponding papers. We repeated each experiment 5 times with different random seeds and report the average test performance.

## 5.2 Recommendation Performance (Q1)

We conducted experiments on the multi-behavior recommendation in Problem 1 to evaluate the performance of each method across the

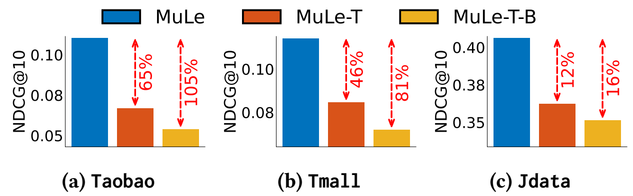



**Figure 3: Effect of multi-grained graphs. As shown by MuLe, it is beneficial to use behavior graphs from all levels.**

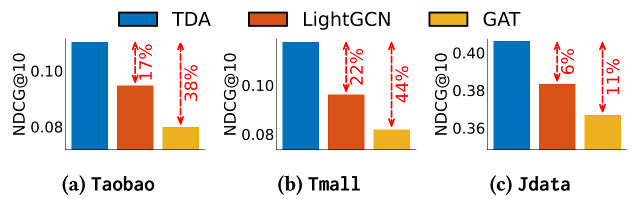



**Figure 4: Effect of target-guided attention for learning $\mathcal{G}_{b \setminus t}$. Our TDA design proves more effective than alternative GNNs.**

benchmark datasets, where $k$ was set to 10 for HR@$k$ and NDCG@$k$. From Table 3, we found the following observations:

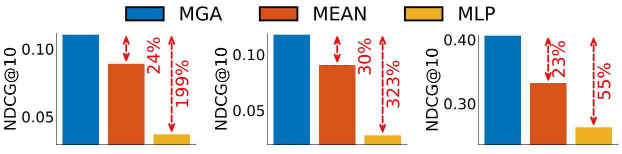
- Our proposed MuLe significantly outperforms the state-of-the-art methods across all datasets, showing **improvements of up to 44.6% in HR@10 and up to 52.9% in NDCG@10 compared to the second-best model**, particularly on the Tmall dataset.
- Using multi-behavior interactions is essential because single-behavior methods, such as LightGCN, which only use the target-behavior graph, significantly underperform compared to most multi-behavior models, including MuLe.
- The unified approaches such as GNMR and MBGCN offer inferior recommendations compared to cascading (e.g., MB-CGCN) and parallel (e.g., PKEF) models, highlighting the importance of separately leveraging behavior-specific interactions.
- While the cascading approaches, such as MB-CGCN, achieve moderate performance, they slightly underperform compared to the parallel approaches such as PKEF and MB-HGCN, including MuLe, due to the noise accumulation issue.
- As a parallel approach, MuLe outperforms others such as HPMR, PKEF, and MB-HGCN. HPMR produced embeddings with limited expressiveness because it indirectly captures target-complemented semantics for each node via projection in MTL. In contrast, our MuLe directly refines the uncertainty of target-complemented interactions through TDA. While MB-HGCN and PKEF use either unified or behavior-specific interactions, they overlook target-related graphs $\mathcal{G}_{b \cap t}$ and $\mathcal{G}_{b \setminus t}$, resulting in lower performance compared to MuLe.

## 5.3 Ablation Study (Q2)

In this section, we investigate the effectiveness of each module or design within MuLe through ablation studies.

*5.3.1 Effect of multi-grained graphs.* We considered the following variants of MuLe to verify the effect of graphs at each level.

- **MuLe** is a version of using the graphs from all levels.
- **MuLe-T** is a version that excludes TRBGs from MuLe.
- **MuLe-T-B** is a version that excludes BSGs from MuLe-T.




(a) Taobao (b) Tmall (c) Jdata

**Figure 5: Effect of multi-grained aggregator for $Z_t$. Our MGA design is better suited than the simple aggregators.**

Figure 3 shows that **using graphs from all levels is beneficial for recommendation performance** across all tested datasets. Especially, MuLe-T significantly underperforms compared to MuLe, highlighting the importance of separating auxiliary behaviors related to the target behavior and treating confident and uncertain (or noisy) interactions differently. On the other hand, MuLe-T-B yields the worst performance, showing that using unified interactions without the semantics of specific behaviors is undesirable.

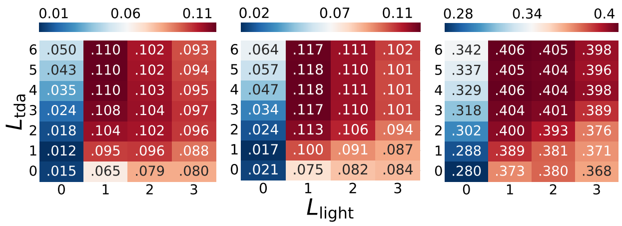
*5.3.2 Effect of TDA.* To validate our design for TDA, we replaced TDA with alternative graph neural networks (GNNs) such as LightGCN [8] or GAT [35] (w/o using the target $\mathbf{E}_t$) for learning the graph $\mathcal{G}_{b\setminus t}$ of each target-complemented behavior $b \setminus t \in \mathcal{B}_{\text{comp}}$. As shown in Figure 4, **our target-guided attention in TDA is more effective than those GNNs**. Note that the performance of LightGCN drops by up to 22% compared to TDA, indicating that treating interactions in $\mathcal{G}_{b\setminus t}$ equally is not beneficial. Moreover, using $\mathbf{E}_t$ is crucial for achieving effective attentions, as GAT without $\mathbf{E}_t$ results in undesirable attentions, decreasing by up to 44%, which is much worse than LightGCN.

*5.3.3 Effect of MGA.* To show MGA's effectiveness for $\mathbf{Z}_t$, given the multi-grained behavior embeddings, we replaced MGA with simple aggregators such as MEAN (averaging the embeddings) or MLP (training a two-layered MLP on their concatenation). Figure 5 demonstrates that **our multi-grained aggregator utilizing an attention mechanism outperforms the alternatives** for obtaining $\mathbf{Z}_t$. The simple aggregators such as MEAN and MLP degrade the performance. Particularly, MLP performs the worst, as it substantially increases model complexity by concatenating $\mathbf{E}_b$, leading to overfitting.

## 5.4 Parameter Sensitivity (Q3)

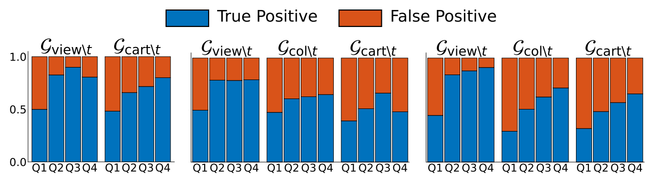
Our MuLe has two main hyperparameters: $L_{\texttt{light}}$ and $L_{\texttt{tda}}$, which represent the numbers of LightGCN and TDA layers, respectively. We investigated how $L_{\texttt{light}}$ and $L_{\texttt{tda}}$ impact the performance of MuLe in terms of NDCG@10 by varying $L_{\texttt{light}}$ from 0 to 3 and $L_{\texttt{tda}}$ from 0 to 6.

Figure 6 indicates that in most cases, **TDA has a greater impact on performance compared to LightGCN**, as the performance improves with increasing $L_{\texttt{tda}}$ while $L_{\texttt{light}}$ decreases. For examples, in the `Tmall` dataset, the best performance is achieved when $L_{\texttt{tda}}$ is 5 and $L_{\texttt{light}}$ is 1. Note that using additional layers for LightGCN was not beneficial, suggesting that a small number of layer is sufficient for obtaining the warm-start embeddings in $\mathcal{G}_{\text{uni}}$ and $\mathcal{G}_b$ or encoding confident interactions in $\mathcal{G}_{b\cap t}$. On the other hand, to effectively denoise interactions in $\mathcal{G}_{b\setminus t}$, TDA considered higher-order neighborhoods, requiring more layers.




(a) Taobao (b) Tmall (c) Jdata

**Figure 6: Effect of $L_{\texttt{tda}}$ and $L_{\texttt{light}}$ of MuLe, where $L_{\texttt{tda}}$ and $L_{\texttt{light}}$ are the numbers of TDA and LightGCN layers, respectively. MuLe tends to provide the best performance when $L_{\texttt{light}}$ is small and $L_{\texttt{tda}}$ is large.**




(a) Taobao (b) Tmall (c) Jdata

**Figure 7: Histogram of the relative ratio of TP to FP attentions in TDA for each $\mathcal{G}_{b\setminus t}$ where $b \in \mathcal{B}_{\text{aux}}$ and $t$ is the target behavior. Our TDA differentiates b.t.w. TP and FP, assigning relatively higher attention weights to TP than to FP.**

## 5.5 Attention Analysis (Q4)

In this section, we discuss the attention results obtained from TDA and MGA of MuLe, respectively.

*5.5.1 Attention analysis for TDA.* We analyze how effectively TDA distinguishes (or denoises) uncertain interactions in $\mathcal{G}_{b\setminus t}$. For evaluation, we first define true or false positive interactions[2] in $\mathcal{G}_{b\setminus t} = (\mathcal{V}, \mathcal{E}_{b\setminus t})$ for the target behavior $t$ as follows:

- **True positive (TP) interaction:** if $(u, i)$ is in training $\mathcal{E}_{b\setminus t}$, and it is also found in $\mathcal{E}_t^{(\texttt{test})}$, then $(u, i)$ is called TP interaction.
- **False positive (FP) interaction:** if $(u, i)$ is in training $\mathcal{E}_{b\setminus t}$, and it is not found in $\mathcal{E}_t^{(\texttt{test})}$, then $(u, i)$ is called FP interaction.

Ideally, TDA should assign low attention weights to FP interactions and high weights to TP ones.

**Attention distribution.** We examined the attention weight of TDA's last layer for each TP or FP interaction in $\mathcal{G}_{b\setminus t}$. For visualization, we divided the attention range $[0, 1]$ into four quartiles, and checked the relative ratio of TP to FP interactions within each bin. As shown in Figure 7, **TDA distinctly differentiates between TP and FP interactions in most cases**. In general, more FP interactions are found in the lower quartiles, while more TP interactions are found in the higher quartiles. Although the absolute value of attention is affected by the number of neighbors (which is why all TP and FP interactions are distributed across the range), FP interactions tend to have relatively lower weights than TP interactions.

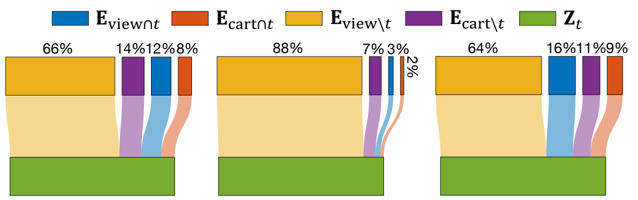
**Statistical difference.** We checked the statistical difference between the TP and FP attention distributions, where $\mu_{\text{tp}}$ and $\mu_{\text{fp}}$ are the mean values of their respective distributions, and $\Delta_\mu$ is their percentage difference. As presented in Table 4, **in most cases, TDA**

[2] Note that TP and FP interactions were only used for the analysis, not for training.

**Table 4: Statistics of TDA attention distributions for TP and FP interactions in $\mathcal{G}_{b\setminus t}$, where $\mu_{\mathtt{tp}}$ and $\mu_{\mathtt{fp}}$ are their mean values, and $\Delta_\mu$ is their percentage difference. In most cases, TDA yields different distributions for TP and FP.**

| Dataset | Taobao | | | Tmall | | | Jdata | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\mu_{\mathtt{tp}}$ | $\mu_{\mathtt{fp}}$ | $\Delta_\mu$ | $\mu_{\mathtt{tp}}$ | $\mu_{\mathtt{fp}}$ | $\Delta_\mu$ | $\mu_{\mathtt{tp}}$ | $\mu_{\mathtt{fp}}$ | $\Delta_\mu$ |
| $\mathcal{G}_{\mathtt{view}\setminus t}$ | 0.033 | 0.019 | 72%$^\dagger$ | 0.041 | 0.026 | 58%$^\dagger$ | 0.183 | 0.054 | 239%$^\dagger$ |
| $\mathcal{G}_{\mathtt{col}\setminus t}$ | | - | | 0.184 | 0.126 | 46%$^\dagger$ | 0.629 | 0.338 | 86%$^\dagger$ |
| $\mathcal{G}_{\mathtt{cart}\setminus t}$ | 0.129 | 0.087 | 49%$^\dagger$ | 0.654 | 0.633 | 3%$^\ddagger$ | 0.638 | 0.425 | 50%$^\dagger$ |

†: it is statistically significant as $p$-value of Student's $t$-test is less than 0.05.
‡: it is not statistically significant as $p$-value is greater than 0.05.




(a) Taobao (b) Tmall (c) Jdata

**Figure 8: Average attention weights for each behavior, used for yielding $\mathbf{Z}_t$. MGA can identify which behaviors are important for $\mathbf{Z}_t$, depending on the datasets.**

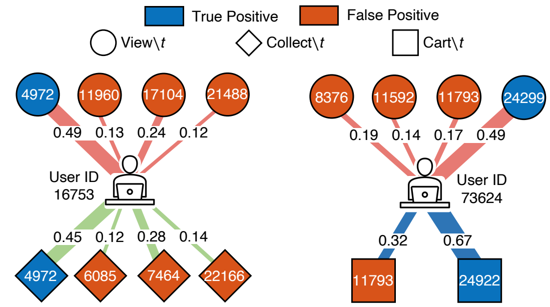
**yields different attention distributions for TP and FP**, with $\mu_{\mathtt{tp}}$ being greater than $\mu_{\mathtt{fp}}$, as well as a large $\Delta_\mu$. For all cases except $\mathcal{G}_{\mathtt{cart}\setminus t}$ of the Tmall dataset, there are statistically significant differences between them, as the $p$-values from Student's $t$-test are less than 0.05. Note that views tend to be noisier (more FPs) than collects, which are noisier than carts, because people usually buy items in their cart rather than those they are just viewing or collecting [43]. According to the table, $\Delta_\mu$ is largest for views, followed by collects, and smallest for carts. This implies that TDA assigns attentions such that TPs and FPs become more distinguishable as interactions get noisier. Thus, TDA effectively reduces the impact of noisy interactions and focuses on potential interests.

*5.5.2 Attention analysis for MGA.* We investigated the attention weights obtained from MGA in Equation (13) to identify which behaviors are important for producing $\mathbf{Z}_t$. For each target-related behavior $b'$, we computed the average of attention weights $\boldsymbol{\alpha}_{b'}$ for all nodes. To simplify visualization, we merged collect into the concept of cart, as they share a similar context in e-commerce.

Figure 8 presents that **MGA effectively captures the importance of each target-related behavior**, where the detailed attention weights depend on the datasets. One notable point is that the attention of $\mathbf{E}_{\mathtt{view}\setminus t}$ takes the largest portion for all datasets. Another point is that the portion of $\mathbf{E}_{b\setminus t}$ is larger than that of $\mathbf{E}_{b\cap t}$ for each behavior $b$. These indicate that MGA effectively harnesses the potential interest in $\mathbf{E}_{b\setminus t}$ refined through TDA, and the refined embeddings are beneficial for the multi-behavior recommendation task, especially when a large number of noisy interactions are given.

## 5.6 Case Study (Q5)

We conducted a case study on how TDA distinctly considers TP and FP interactions, defined in Section 5.5.1. Figure 9 shows the attention weights of TDA's last layer for users with IDs 16753 and 73624 in the Jdata dataset. **In both cases, TDA consistently assigned higher weights to TPs than to FPs**, implying that TDA effectively identifies potential interests from those interactions as intended by its design. Although we demonstrate only successful cases and acknowledge the presence of ambiguous cases, as verified in Figure 7 and Table 4, there were more desirable cases overall, with similar results in other datasets.




(a) Case 1 in Jdata (b) Case 2 in Jdata

**Figure 9: Case studies of TDA's attentions of specific users. It tends to assign high weights to TPs and low weights to FPs, effectively drawing potential interests of users during learning embeddings.**

## 6 Conclusion

In this paper, we propose MuLe, a novel graph learning method for effective multi-behavior recommendation. Our MuLe learns multi-grained graphs to fully leverage the multi-faceted relationships of behaviors. Additionally, MuLe's TDA effectively denoises target-complemented interactions using attention mechanisms to distinguish potential interest from noisy interactions. Finally, MuLe's MGA aggregates diverse behavior embeddings with attention to produce the final embeddings related to the target behavior. Through extensive experiments on three real-world benchmark datasets, we demonstrate that our MuLe significantly outperforms the state-of-the-art methods, improving by up to 44.6% in HR@10 and 52.9% in NDCG@10, respectively. We additionally confirm the effectiveness of TDA and MGA through further experiments, including ablation studies and attention analyses, which highlight the significance of incorporating multi-grained graphs and adeptly managing uncertain auxiliary interactions. Our code and datasets are available at https://github.com/geonwooko/MULE.

## Acknowledgments

We would like to thank all anonymous reviewers for their dedicated reviews and comments. This work was supported by National Research Foundation of Korea (NRF) grant by the Korea government (MSIT) (No.2021R1C1C1008526), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2021-II212068, Artificial Intelligence Innovation Hub), and Cultural Heritage Administration, National Research Institute of Cultural Heritage (No.RS-2024-00398805). This research was partly supported by the MSIT (Ministry of Science, ICT), Korea, under the National Program for Excellence in SW, supervised by the IITP in 2024 (2022-0-01067).

## References


[1] Zhi Bian, Shaojun Zhou, Hao Fu, Qihong Yang, Zhenqi Sun, Junjie Tang, Guiquan Liu, Kaikui Liu, and Xiaolong Li. 2021. Denoising User-aware Memory Network for Recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*. ACM, 400–410. https://doi.org/10.1145/3460231.3474237

[2] Chong Chen, Weizhi Ma, Min Zhang, Zhaowei Wang, Xiuqiang He, Chenyang Wang, Yiqun Liu, and Shaoping Ma. 2021. Graph Heterogeneous Multi-Relational Recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 3958–3966. https://doi.org/10.1609/AAAI.V35I5.16515

[3] Zhiyong Cheng, Sai Han, Fan Liu, Lei Zhu, Zan Gao, and Yuxin Peng. 2023. Multi-Behavior Recommendation with Cascading Graph Convolution Networks. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*. ACM, 1181–1189. https://doi.org/10.1145/3543507.3583439

[4] Jingtao Ding, Guanghui Yu, Xiangnan He, Yuhan Quan, Yong Li, Tat-Seng Chua, Depeng Jin, and Jiajie Yu. 2018. Improving Implicit Recommender Systems with View Data. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 3343–3349. https://doi.org/10.24963/IJCAI.2018/464

[5] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A System for Recommending 3+ Billion Items to 200+ Million Users in Real-Time. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. ACM, 1775–1784. https://doi.org/10.1145/3178876.3186183

[6] 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 *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*. IEEE, 1554–1557. https://doi.org/10.1109/ICDE.2019.00140

[7] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, Lina Yao, Yang Song, and Depeng Jin. 2021. Learning to Recommend With Multiple Cascading Behaviors. *IEEE Trans. Knowl. Data Eng.* 33, 6 (2021), 2588–2601. https://doi.org/10.1109/TKDE.2019.2958808

[8] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong 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, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 639–648. https://doi.org/10.1145/3397271.3401063

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*. ACM, 173–182. https://doi.org/10.1145/3038912.3052569

[10] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. ACM, 549–558. https://doi.org/10.1145/2911451.2911489

[11] 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, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 659–668. https://doi.org/10.1145/3397271.3401072

[12] Jinhong Jung, Woojeong Jin, and U Kang. 2020. Random walk-based ranking in signed social networks: model and algorithms. *Knowl. Inf. Syst.* 62, 2 (2020), 571–610. https://doi.org/10.1007/S10115-019-01364-Z

[13] Jinhong Jung, Woojeong Jin, Lee Sael, and U Kang. 2016. Personalized Ranking in Signed Networks Using Signed Random Walk with Restart. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*. IEEE Computer Society, 973–978. https://doi.org/10.1109/ICDM.2016.0122

[14] Jinhong Jung, Namyong Park, Lee Sael, and U Kang. 2017. BePI: Fast and Memory-Efficient Method for Billion-Scale Random Walk with Restart. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. ACM, 789–804. https://doi.org/10.1145/3035918.3035950

[15] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1412.6980

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[17] Geonwoo Ko and Jinhong Jung. 2024. Learning disentangled representations in signed directed graphs without social assumptions. *Inf. Sci.* 665 (2024), 120373. https://doi.org/10.1016/J.INS.2024.120373

[18] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M. Jose. 2009. On social networks and collaborative recommendation. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*. ACM, 195–202. https://doi.org/10.1145/1571941.1571977

[19] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37. https://doi.org/10.1109/MC.2009.263

[20] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2012. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*. ACM, 173–182. https://doi.org/10.1145/2124295.2124317

[21] Jong-whi Lee and Jinhong Jung. 2023. Time-Aware Random Walk Diffusion to Improve Dynamic Graph Learning. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*. AAAI Press, 8473–8481. https://doi.org/10.1609/AAAI.V37I7.26021

[22] Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. 2021. Interest-aware Message-Passing GCN for Recommendation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*. ACM / IW3C2, 1296–1305. https://doi.org/10.1145/3442381.3449986

[23] Shengchao Liu, Yingyu Liang, and Anthony Gitter. 2019. Loss-Balanced Task Weighting to Reduce Negative Transfer in Multi-Task Learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 9977–9978. https://doi.org/10.1609/AAAI.V33I01.33019977

[24] Babak Loni, Roberto Pagano, Martha A. Larson, and Alan Hanjalic. 2016. Bayesian Personalized Ranking with Multi-Channel User Feedback. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*. ACM, 361–364. https://doi.org/10.1145/2959100.2959163

[25] Chang Meng, Chenhao Zhai, Yu Yang, Hengyu Zhang, and Xiu Li. 2023. Parallel Knowledge Enhancement based Framework for Multi-behavior Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*. ACM, 1797–1806. https://doi.org/10.1145/3583780.3615004

[26] Chang Meng, Hengyu Zhang, Wei Guo, Huifeng Guo, Haotian Liu, Yingxue Zhang, Hongkun Zheng, Ruiming Tang, Xiu Li, and Rui Zhang. 2023. Hierarchical Projection Enhanced Multi-behavior Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*. ACM, 4649–4660. https://doi.org/10.1145/3580305.3599838

[27] Haekyu Park, Jinhong Jung, and U Kang. 2017. A comparative study of matrix factorization and random walk with restart in recommender systems. In *2017 IEEE International Conference on Big Data (IEEE BigData 2017), Boston, MA, USA, December 11-14, 2017*. IEEE Computer Society, 756–765. https://doi.org/10.1109/BIGDATA.2017.8257991

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*. AUAI Press, 452–461. https://www.auai.org/uai2009/papers/UAI2009_0139_48141db02b9f0b02bc7158819ebfa2c7.pdf

[29] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*. ACM, 285–295. https://doi.org/10.1145/371920.372071

[30] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings (Lecture Notes in Computer Science, Vol. 10843)*. Springer, 593–607. https://doi.org/10.1007/978-3-319-93417-4_38

[31] Changwon Seo, Kyeong-Joong Jeong, Sungsu Lim, and Won-Yong Shin. 2024. SiReN: Sign-Aware Recommendation Using Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.* 35, 4 (2024), 4729–4743. https://doi.org/10.1109/TNNLS.2022.3175772

[32] Ajit Paul Singh and Geoffrey J. Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*. ACM, 650–658. https://doi.org/10.1145/1401890.1401969

[33] Liang Tang, Bo Long, Bee-Chung Chen, and Deepak Agarwal. 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, San Francisco, CA, USA, August 13-17, 2016.* ACM, 283–292. https://doi.org/10.1145/2939672.2939690

[34] 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 Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA.* 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[35] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJXMpikCZ

[36] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018.* ACM, 86–94. https://doi.org/10.1145/3240323.3240369

[37] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 373–381.

[38] Wei Wei, Chao Huang, Lianghao Xia, Yong Xu, Jiashu Zhao, and Dawei Yin. 2022. Contrastive Meta Learning with Behavior Multiplicity for Recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022.* ACM, 1120–1128. https://doi.org/10.1145/3488560.3498527

[39] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Mengyin Lu, and Liefeng Bo. 2021. Multi-Behavior Enhanced Recommendation with Cross-Interaction Collaborative Relation Modeling. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021.* IEEE, 1931–1936. https://doi.org/10.1109/ICDE51399.2021.00179

[40] 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, SIGIR 2020, Virtual Event, China, July 25-30, 2020.* ACM, 2397–2406. https://doi.org/10.1145/3397271.3401445

[41] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Xiyue Zhang, Hongsheng Yang, Jian Pei, and Liefeng Bo. 2021. Knowledge-Enhanced Hierarchical Graph Transformer Network for Multi-Behavior Recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021.* AAAI Press, 4486–4493. https://doi.org/10.1609/AAAI.V35I5.16576

[42] Lianghao Xia, Yong Xu, Chao Huang, Peng Dai, and Liefeng Bo. 2021. Graph Meta Network for Multi-Behavior Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021.* ACM, 757–766. https://doi.org/10.1145/3404835.3462972

[43] Mingshi Yan, Zhiyong Cheng, Chen Gao, Jing Sun, Fan Liu, Fuming Sun, and Haojie Li. 2024. Cascading Residual Graph Convolutional Network for Multi-Behavior Recommendation. *ACM Trans. Inf. Syst.* 42, 1 (2024), 10:1–10:26. https://doi.org/10.1145/3587693

[44] Mingshi Yan, Zhiyong Cheng, Jing Sun, Fuming Sun, and Yuxin Peng. 2023. MB-HGCN: A Hierarchical Graph Convolutional Network for Multi-behavior Recommendation. *CoRR* abs/2306.10679 (2023). https://doi.org/10.48550/ARXIV.2306.10679 arXiv:2306.10679

[45] Weifeng Zhang, Jingwen Mao, Yi Cao, and Congfu Xu. 2020. Multiplex Graph Neural Networks for Multi-behavior Recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020.* ACM, 2313–2316. https://doi.org/10.1145/3340531.3412119