Denoising and Prompt-Tuning for Multi-Behavior Recommendation

Chi Zhang
Harbin Engineering University
Harbin, China
zhangchi20@hrbeu.edu.cn

Rui Chen*
Harbin Engineering University
Harbin, China
ruichen@hrbeu.edu.cn

Xiangyu Zhao*
City University of Hong Kong
Hong Kong
xianzhao@cityu.edu.hk

Qilong Han*
Harbin Engineering University
Harbin, China
hanqilong@hrbeu.edu.cn

Li Li
University of Delaware
Newark, United States
lilee@udel.edu

ABSTRACT

In practical recommendation scenarios, users often interact with items under multi-typed behaviors (e.g., click, add-to-cart, and purchase). Traditional collaborative filtering techniques typically assume that users only have a single type of behavior with items, making it insufficient to utilize complex collaborative signals to learn informative representations and infer actual user preferences. Consequently, some pioneer studies explore modeling multi-behavior heterogeneity to learn better representations and boost the performance of recommendations for a target behavior. However, a large number of auxiliary behaviors (i.e., click and add-to-cart) could introduce irrelevant information to recommenders, which could mislead the target behavior (i.e., purchase) recommendation, rendering two critical challenges: (i) denoising auxiliary behaviors and (ii) bridging the semantic gap between auxiliary and target behaviors. Motivated by the above observation, we propose a novel framework–**D**enoising and **P**rompt-**T**uning (DPT) with a three-stage learning paradigm to solve the aforementioned challenges. In particular, DPT is equipped with a pattern-enhanced graph encoder in the first stage to learn complex patterns as prior knowledge in a data-driven manner to guide learning informative representation and pinpointing reliable noise for subsequent stages. Accordingly, we adopt different lightweight tuning approaches with effectiveness and efficiency in the following stages to further attenuate the influence of noise and alleviate the semantic gap among multi-typed behaviors. Extensive experiments on two real-world datasets demonstrate the superiority of DPT over a wide range of state-of-the-art methods. The implementation code is available online at https://github.com/zc-97/DPT.

CCS CONCEPTS

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

*Corresponding authors.

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583513

KEYWORDS

Multi-behavior recommendation, auxiliary behavior denoising, prompt tuning, graph neural networks

ACM Reference Format:
Chi Zhang, Rui Chen, Xiangyu Zhao, Qilong Han, and Li Li. 2023. Denoising and Prompt-Tuning for Multi-Behavior Recommendation. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3543507.3583513

1 INTRODUCTION

Recommender systems have been widely used in various online applications (e.g., e-commerce [52], news [55], and social media [18, 54]) to alleviate information overload and meet user personalized preferences. Traditional collaborative filtering (CF) techniques [7, 15, 19, 32, 37, 45] typically assume that users only interact with items under a single type of behavior, which is insufficient to learn behavior heterogeneity for making accurate recommendations, leading to the problem of *multi-behavior recommendation.*

In practice, users often interact with items under multi-typed behaviors (e.g., click, add-to-cart, and purchase). Consequently, the mainstream multi-behavior recommendation studies leverage different deep models (e.g., neural collaborative filtering [2], attention mechanisms [5, 36, 39, 40] and graph neural networks [1, 8, 11, 33, 41]) to learn complex relations from numerous interactions under *auxiliary behaviors* (i.e., click and add-to-cart). Thus, the learned knowledge could serve as additional information to enhance the relatively sparse *target behavior* (i.e., purchase) and better infer users' preferences for target behavior recommendation. Despite existing methods' effectiveness of enhancing target behavior recommendation via multi-behavior heterogeneity, ignoring numerous auxiliary behaviors could also introduce noisy information and irrelevant semantics, leading to inherent sub-optimal representation learning for target behavior recommendation. To this end, as illustrated in Figure 1, the numerous uncontrollable auxiliary behaviors could render two non-trivial challenges:

Noisy Interactions under Auxiliary Behaviors. Auxiliary behaviors (i.e., click) typically contain some *inherently* noisy user-item interactions (e.g., accidental interactions [24, 36]), which cannot accurately reflect user interests. Without fine control, the learned multi-behavior knowledge could be inevitably affected by noise in such data. Therefore, when transferring such knowledge to target behavior recommendation, the uncontrollable noise's influence

would be further magnified with increasing auxiliary behaviors in a dataset (i.e., the larger the difference of the sizes of auxiliary and target behaviors, the greater the influence is). Despite its practical value, lacking supervised labels to indicate noisy interactions leads to a unique challenge in solving the denoising problem.

Semantic Gap among Multi-Typed Behaviors. While some user-item interactions overlap under multi-typed behaviors, the target behavior still significantly differs from auxiliary behaviors from a semantic feature perspective. For example, a large number of clicks cannot lead to purchases in e-commerce [34, 43]. Therefore, numerous auxiliary behaviors could inevitably make the learned knowledge over-squash into the semantic space of auxiliary behaviors. Thus, the inherent challenge of bridging the semantic gap between auxiliary and target behaviors lies in how to effectively transfer sufficient target-specific semantics from learned knowledge of multi-typed behaviors to target recommendation without jeopardizing knowledge informativeness.

Contribution. In view of the above challenges, we propose a novel framework–**D**enoising and **P**rompt-**T**uning (DPT) with a *three-stage* learning paradigm for multi-behavior recommendation. Specifically, we derive a pattern-enhanced graph encoder in the first stage to learn complex patterns in a data-driven manner. The patterns are served as prior knowledge to learn informative representations and guide the denoising module to pinpoint inherent noise in auxiliary behaviors. In the second stage, we adopt a re-initializing technique to attenuate the influence of noise further. Finally, we adopt a deeply continuous prompt-tuning paradigm to alleviate the semantic gap among multi-typed behaviors, thus adaptively extracting target-specific information for target behavior recommendation. We provide a case study in Section 4.4 to show how the three-stage learning paradigm affects recommendations.

We summarize our main technical contributions as follows.

- To the best of our knowledge, this is the first paper that proposes to ameliorate multi-behavior recommendation from a new perspective that attenuates auxiliary behaviors' negative influences on target behavior recommendation. We identify two critical challenges in existing multi-behavior recommendation studies rendered by the numerous auxiliary behaviors, which have been less explored in literature.
- We propose a novel multi-behavior recommendation framework named DPT, which is powered by a three-stage learning paradigm to attenuate the influence of noise and alleviate the semantic gap among multi-typed behaviors for target behavior recommendation. In particular, we pinpoint noisy interactions without requiring supervised signals (i.e., labels to indicate noise). We also effectively and efficiently bridge the semantic gap between auxiliary and target behaviors without requiring additional prior knowledge (e.g., knowledge graph).
- We perform extensive experiments on two real-world public recommendation datasets and show that our DPT model consistently outperforms a large number of state-of-the-art competitors.

2 PRELIMINARIES

Let $\mathcal{U} = \{u_1, u_2, \cdots, u_{|\mathcal{U}|}\}$ and $\mathcal{I} = \{i_1, i_2, \cdots, i_{|\mathcal{I}|}\}$ denote the set of users and items, respectively. Following previous studies [1, 11, 33, 41], we take purchase as the target behavior and others serving as auxiliary behaviors (i.e., click, add-to-favorite, and add-to-cart). For simplicity, we use b^a and b^t to mean auxiliary behaviors and the target behavior in the remainder, respectively. Accordingly, we define the multi-behavior data as a set of interaction matrices $\mathcal{A} = \{A^a, A^t\}$, where $A^a, A^t \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$. Each element in A^* indicates whether a user u has interacted with item i under behavior $*$ (i.e., $A^*_{ui} = 1$) or not (i.e., $A^*_{ui} = 0$), where $* \in \{a, t\}$.

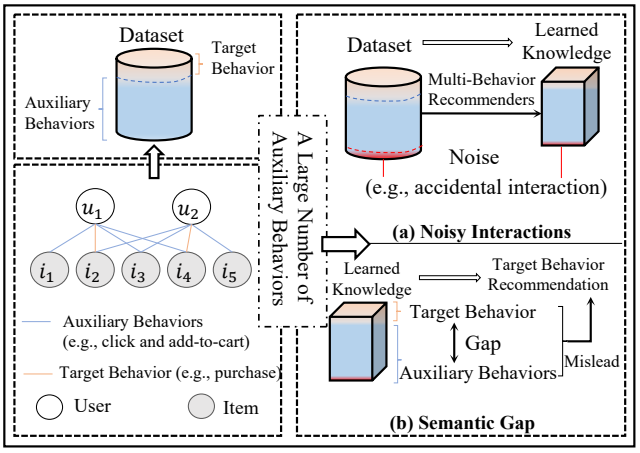

Figure 1: An illustration of the two limitations.

User-Item Multi-Behavior Graph. We construct $\mathcal{G} = \{\mathcal{G}^a, \mathcal{G}^t\}$ to represent interactions under auxiliary and target behaviors. For each graph $\mathcal{G}^* = (\mathcal{V}, \mathcal{E}^*) \in \mathcal{G}$, there is an edge ε^*_{ui} between user u and item i in $\mathcal{E}^*$ iff $A^*_{ui} = 1$, where $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ and $* \in \{a, t\}$.

Problem Statement. We formally describe our task as follows: **Input**: constructed multi-behavior graph $\mathcal{G}$. **Output**: a noiseless multi-behavior graph $\mathcal{G}' = \{\mathcal{G}'^a, \mathcal{G}^t\}$, where $\mathcal{G}'^a = \{\mathcal{V}, \mathcal{E}'^a\}$ and $\mathcal{E}'^a \subseteq \mathcal{E}^a$, and the predictive function $\mathcal{F}(u, i|\mathcal{G}', \Theta)$, which estimates the likelihood of user u adopting the item i of the target behavior type, where Θ is the set of model parameters.

3 METHODOLOGY

As illustrated in Figure 2, DPT is learned in a three-stage learning paradigm. Specifically, the first stage is equipped with a pattern-enhanced graph encoder to learn informative representations and guide the denoising module to generate a noiseless multi-behavior graph for subsequent stages. After that, we adopt different light-weight tuning approaches to further attenuate noise's influence and alleviate the semantic gap among multi-typed behaviors.

3.1 User/Item Relation Graph Construction

Recall that lack of labels for noise identification makes the denoising problem challenging. In practice, there typically exists multiple patterns between user-user relations (e.g., social relations [28]) and item-item relations (e.g., knowledge-driven relations [27, 29]), which could serve as prior knowledge to improve the representation learning ability. Hence, we construct two types (i.e., user and item) of relation graphs to mine patterns in a data-driven manner.

3.1.1 User Relation Graph. In practice, user-side information could be well reflected by their co-interactions [9, 23]. Consequently, we utilize users' co-interactive behaviors (e.g., co-view and co-purchase) to learn the complex patterns. Mathematically, there is

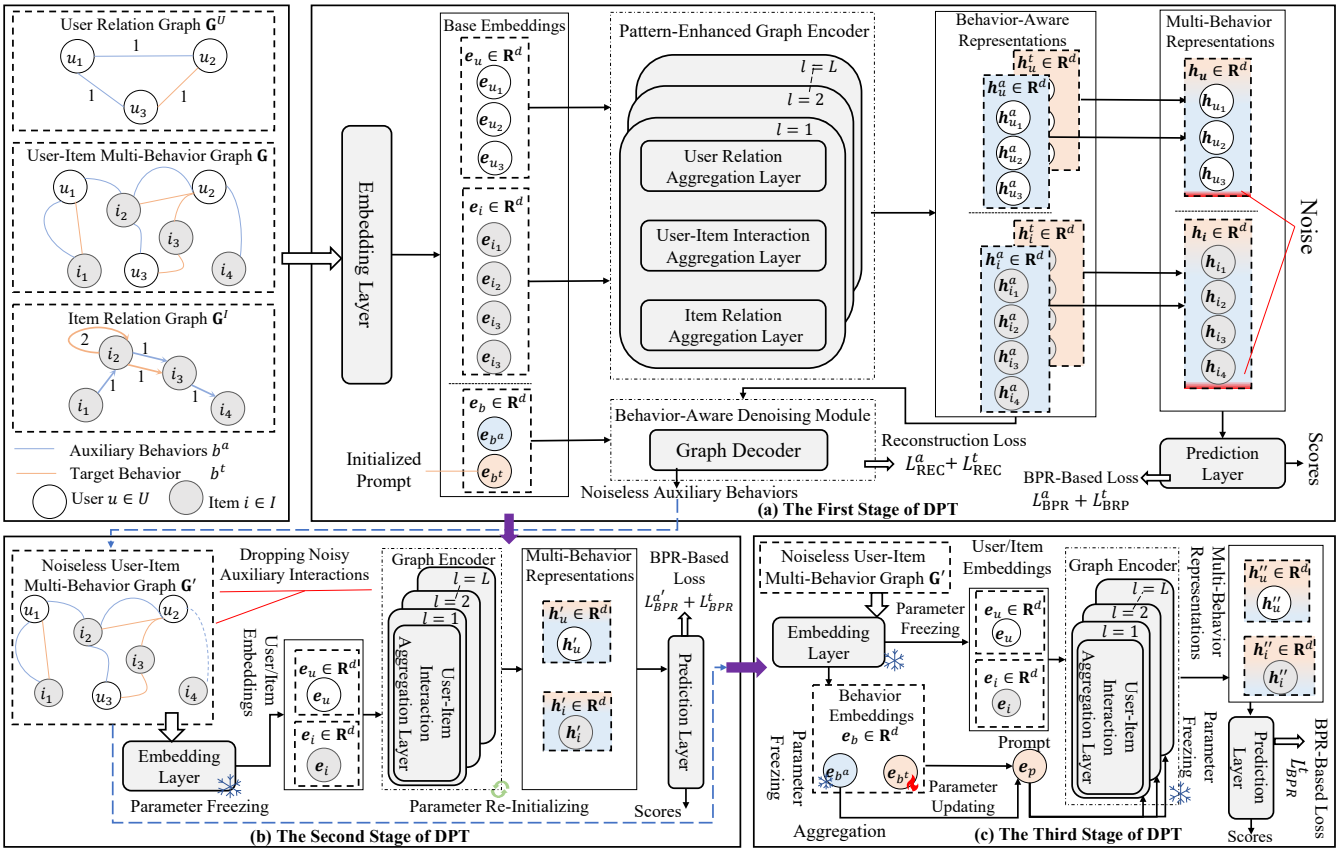

Figure 2: The architecture of the proposed DPT model. (a) The first stage of DPT takes different relation graphs as input and outputs a noiseless graph $\mathcal{G}'$ for subsequent stages; (b) The second stage of DPT re-initializes partially learned parameters to further attenuate noise's influence; (c) The third stage of DPT uses continuous prompts to bridge the behavioral semantic gap.

an edge $\varepsilon^{U,*}_{uv}$ between user $u, v \in \mathcal{U}$ in the user relation graph $\mathcal{G}^U$ under the behavior $*$ iff $\mathcal{E}^*_{u\cdot} \cap \mathcal{E}^*_{v\cdot} \neq \emptyset$ and $\mathcal{E}^*_{u\cdot} \cup \mathcal{E}^*_{v\cdot} \neq \emptyset$. The weight $w^{U,*}_{uv}$ of $\varepsilon^{U,*}_{uv}$ is calculated by Jaccard similarity as $\frac{|\mathcal{E}^*_{u\cdot} \cap \mathcal{E}^*_{v\cdot}|}{|\mathcal{E}^*_{u\cdot} \cup \mathcal{E}^*_{v\cdot}|}$ to indicate the relevance strength between the users.

3.1.2 Item Relation Graph. As suggested by previous studies [6, 40], unlike user relations, items possess directed relations due to the existence of temporal information. Consequently, we construct a directed item relation graph for better representation learning. Specifically, we sort each user u's interactions into a sequence S_u according to interaction timestamps, and count the number of occurrences of each pair of items (i, j) in a particular order (e.g., $i \xrightarrow{*} j$) from all sequences. Formally, there is an edge $\varepsilon^{I,*}_{ij}$ from item i to j in the item relation graph $\mathcal{G}^I$ under behavior $*$ iff $\text{Cnt}(i \xrightarrow{*} j) > 0$, where $\text{Cnt}(\cdot)$ is a counter. The weight $w^{I,*}_{ij}$ of $\varepsilon^{I,*}_{ij}$ is calculated by Jaccard similarity as $\frac{\text{Cnt}(i \xrightarrow{*} j)}{\text{Cnt}(i \xrightarrow{*} j) + \text{Cnt}(j \xrightarrow{*} i)}$ to indicate the sequential strength.

3.2 Embedding Layer

To train DPT, we first learn the embeddings of users, items, and behaviors by mapping their IDs to dense embedding vector $\boldsymbol{e} \in \mathbb{R}^d$. Formally, we build four embedding look-up tables for initialization:

$$\boldsymbol{e}_u = \boldsymbol{x}_u \boldsymbol{W}_u \; ; \; \boldsymbol{e}_i = \boldsymbol{x}_i \boldsymbol{W}_i \; ; \; \boldsymbol{e}_{b^a} = \boldsymbol{x}_{b^a} \boldsymbol{W}_{b^a} \; ; \; \boldsymbol{e}_{b^t} = \boldsymbol{x}_{b^t} \boldsymbol{W}_{b^t}, \tag{1}$$

where $\boldsymbol{W}_u \in \mathbb{R}^{|\mathcal{U}| \times d}$, $\boldsymbol{W}_i \in \mathbb{R}^{|\mathcal{I}| \times d}$, $\boldsymbol{W}_{b^a} \in \mathbb{R}^{(K-1) \times d}$, and $\boldsymbol{W}_{b^t} \in \mathbb{R}^d$ are trainable matrices. $\boldsymbol{x}_u$, $\boldsymbol{x}_i$, $\boldsymbol{x}_{b^a}$ and $\boldsymbol{x}_{b^t}$ are the one-hot encoding vectors of the IDs of user u, item i, auxiliary behavior b^a, and target behavior b^t, respectively.

3.3 Pattern-Enhanced Graph Encoder

As illustrated in Figure 3, we design a pattern-enhanced graph encoder, consisting of three disentangled aggregation layers to individually learn multi-view patterns based on constructed graphs.

3.3.1 User Relation Aggregation Layer. Since the user relation graph $\mathcal{G}^U$ is an undirected graph, we encode such relations without distinguishing directions. Formally, for each user u^*, we aggregate the information from his/her neighbor's information and himself/herself as per $\mathcal{G}^U$ to learn similar behavior patterns among users and generate layer-specific behavior-aware encoded vectors as follows

$$\boldsymbol{e}^{U,(l)}_{u^*} = f^{U,(l)}_{u \to u}\Big(\sum_{u \in N^U_{u^*}} \boldsymbol{e}^{(l)}_u, \boldsymbol{e}^{(l)}_{u^*}, \mathcal{G}^{U,*}, \Theta^{(l)}_U \Big), \tag{2}$$

where $f^{U,(l)}_{u \to u}(\cdot)$ is the aggregation function to encode user relations, $l \in \mathbb{R}$ is the number of layer, $N^U_{u^*}$ is the neighbor of u^* in $\mathcal{G}^{U,*}$ under

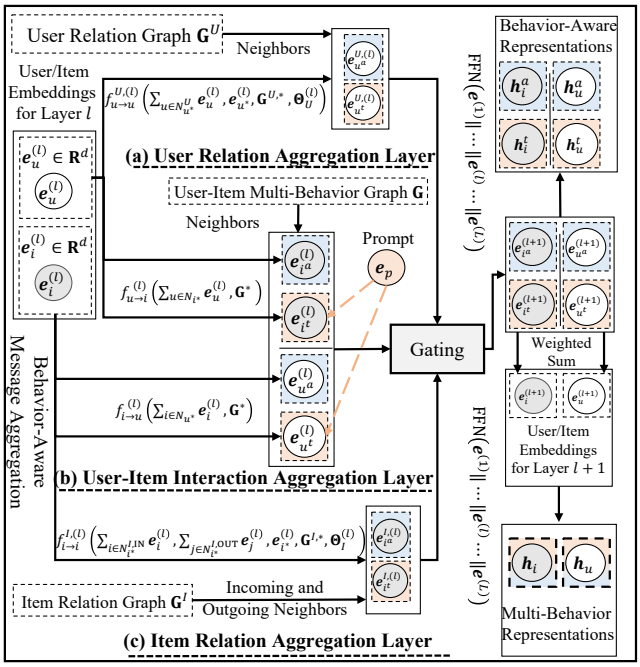

Figure 3: The proposed pattern-enhanced graph encoder.

behavior $* \in \{a, t\}$, and $\Theta_U^{(l)}$ is the set of trainable parameters. Motivated by [6, 16], which utilize a convolution operator to preserve dimension-wise information and effectively encode homogeneous relations, we implement $f^{U,(l)}(\cdot)$ and generate $\boldsymbol{e}_{u^*}^{U,(l)}$ via

$$\boldsymbol{e}_{u^*}^{U,(l)} = \text{Conv}^{U,(l)}([\sum_{u \in N_{u^*}^U} (w_{uu^*}^{U,*} \boldsymbol{e}_u^{(l)}) \| \boldsymbol{e}_{u^*}^{(l)}], \bar{\Theta}_U^{(l)}), \tag{3}$$

where $\text{Conv}^{U,(l)}(\cdot)$ is a layer-specific convolution operator with stride 1 and filter size 2×1, $w_{uu^*}^{U,*}$ is the normalized weight of edge $\varepsilon_{uu^*}^{U,*}$ in $\mathcal{G}^{U,*}$, $\|$ is the concatenation operation, and $\bar{\Theta}_U^{(l)}$ is the set of trainable weight parameters of the convolution operator.

3.3.2 Item Relation Aggregation Layer. Since items typically have directed patterns, we perform an attention-based message passing scheme [26] (i.e., $f_{i \to i}^{I,(l)}(\cdot)$) for adaptively encoding. For each item i^* in the item relation graph $\mathcal{G}^{I,*}$ under behavior $*$, we first assign a 2-dimensional attention vector $\boldsymbol{\alpha}^*$ to weight i^*'s incoming (i.e., $i \in N_{i^*}^{I,\text{IN}}$) and outgoing (i.e., $j \in N_{i^*}^{I,\text{OUT}}$) neighbors. Formally, we implement $f_{i \to i}^{I,(l)}(\cdot)$ and generate $\boldsymbol{e}_{i^*}^{I,(l)}$ as follows

$$\begin{aligned}
\boldsymbol{e}_{i^*}^{I,(l)} &= f_{i \to i}^{I,(l)}(\sum_{i \in N_{i^*}^{I,\text{IN}}} \boldsymbol{e}_i^{(l)}, \sum_{j \in N_{i^*}^{I,\text{OUT}}} \boldsymbol{e}_j^{(l)}, \boldsymbol{e}_{i^*}^{(l)}, \mathcal{G}^{I,*}, \Theta_I^{(l)}), \\
\boldsymbol{\alpha}^* &= [\alpha_{\text{IN}}^*, \alpha_{\text{OUT}}^*] = \frac{\exp(\boldsymbol{e}_{i^*}^\top \boldsymbol{W}_{I,n}^{(l)} / \sqrt{d})}{\sum_{n \in \{\text{IN,OUT}\}} \exp(\boldsymbol{e}_{i^*}^\top \boldsymbol{W}_{I,n}^{(l)} / \sqrt{d}))}, \\
\boldsymbol{e}_{N_{i^*}^I}^{(l)} &= \alpha_{\text{IN}}^* \sum_{i \in N_{i^*}^{I,\text{IN}}} (w_{ii^*}^{I,*} \boldsymbol{e}_i^{(l)}) + \alpha_{\text{OUT}}^* \sum_{i^* \in N_{i^*}^{I,\text{OUT}}} (w_{i^*j}^{I,*} \boldsymbol{e}_j^{(l)}), \\
\boldsymbol{e}_{i^*}^{I,(l)} &= \text{Conv}^{I,(l)}(\boldsymbol{e}_{N_{i^*}^I}^{(l)} \| \boldsymbol{e}_{i^*}^{(l)}, \bar{\Theta}_I^{(l)}),
\end{aligned} \tag{4}$$

where $\text{Conv}^{U,(l)}(\cdot)$ is a similar convolution used in Eq. (3) with different parameters, $w_{ii^*}^{I,*}, w_{i^*j}^{I,*}$ are the normalized weights of edge in $\mathcal{G}^{I,*}$, and $\Theta_I^{(l)}$ is the set of trainable parameters, including attention matrices $W_{I,n}^{(l)} \in \mathbb{R}^{d \times d}$ and the convolutional filter $\bar{\Theta}_I^{(l)}$.

3.3.3 User-Item Interaction Aggregation Layer. For each user u^* and item i^*, we formally present a lightweight message passing scheme based on LightGCN [7] as follows

$$\begin{aligned}
\boldsymbol{e}_{u^*}^{(l)} &= f_{i \to u}^{(l)}(\sum_{i \in N_{u^*}} \boldsymbol{e}_i^{(l)}, \mathcal{G}^*); \boldsymbol{e}_{i^*}^{(l)} = f_{u \to i}^{(l)}(\sum_{u \in N_{i^*}} \boldsymbol{e}_u^{(l)}, \mathcal{G}^*), \\
\boldsymbol{e}_{u^*}^{(l)} &= \sum_{i \in N_{u^*}} \boldsymbol{e}_i^{(l)}; \boldsymbol{e}_{i^*}^{(l)} = \sum_{u \in N_{i^*}} \boldsymbol{e}_u^{(l)}.
\end{aligned} \tag{5}$$

To encoder multiple patterns from different graphs into embeddings, we design a gating operator, which could adaptively balance and fuse the different views generated from the (l)- to $(l+1)$-th layers. More specifically, we generate the pattern-enhanced user/item embedddings $\boldsymbol{e}_{u^*}^{(l+1)}$ and $\boldsymbol{e}_{i^*}^{(l+1)}$ under the type of behavior $*$ via

$$\begin{aligned}
\beta_u^* &= \sigma([\boldsymbol{e}_{u^*}^{(l)} \| \boldsymbol{e}_{u^*}^{U,(l)}] \boldsymbol{W}_{UU}^{(l)}); \beta_i^* = \sigma([\boldsymbol{e}_{i^*}^{(l)} \| \boldsymbol{e}_{i^*}^{I,(l)}] \boldsymbol{W}_{II}^{(l)}), \\
\boldsymbol{e}_{u^*}^{(l+1)} &= \boldsymbol{e}_{u^*}^{(l)} + \frac{1 - \beta_u^*}{\beta_u^*} \boldsymbol{e}_{u^*}^{U,(l)}; \boldsymbol{e}_{i^*}^{(l+1)} = \boldsymbol{e}_{i^*}^{(l)} + \frac{1 - \beta_i^*}{\beta_i^*} \boldsymbol{e}_{i^*}^{I,(l)},
\end{aligned} \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function, $\beta_u^*, \beta_i^* \in \mathbb{R}$ are weight scalars, and $\boldsymbol{W}_{UU}^{(l)}, \boldsymbol{W}_{II}^{(l)} \in \mathbb{R}^{2d \times 1}$ are trainable parameters. Accordingly, we generate multi-behavior embeddings in the $(l+1)$-th layer via

$$\boldsymbol{e}_u^{(l+1)} = \frac{1}{|\{a, t\}|} \sum_{* \in \{a,t\}} \boldsymbol{e}_{u^*}^{(l)}; \boldsymbol{e}_i^{(l+1)} = \frac{1}{|\{a, t\}|} \sum_{* \in \{a,t\}} \boldsymbol{e}_{i^*}^{(l)}. \tag{7}$$

We concatenate user/item embeddings across each layer to generate the final behavior-aware (i.e., $\boldsymbol{h}_u^*, \boldsymbol{h}_i^* \in \mathbb{R}^d$) and multi-behavior (i.e., $\boldsymbol{h}_u, \boldsymbol{h}_i \in \mathbb{R}^d$) representations:

$$\begin{aligned}
\boldsymbol{h}_u^* &= f_U(\boldsymbol{e}_{u^*}^{(1)} \| \boldsymbol{e}_{u^*}^{(2)} \| \cdots \| \boldsymbol{e}_{u^*}^{(L)}); \boldsymbol{h}_i^* = f_I(\boldsymbol{e}_{i^*}^{(1)} \| \boldsymbol{e}_{i^*}^{(2)} \| \cdots \| \boldsymbol{e}_{i^*}^{(L)}), \\
\boldsymbol{h}_u &= f_U(\boldsymbol{e}_u^{(1)} \| \boldsymbol{e}_u^{(2)} \| \cdots \| \boldsymbol{e}_u^{(L)}); \boldsymbol{h}_i = f_I(\boldsymbol{e}_i^{(1)} \| \boldsymbol{e}_i^{(2)} \| \cdots \| \boldsymbol{e}_i^{(L)}),
\end{aligned} \tag{8}$$

where f_U and f_I are feedforward layers activated by the ReLU function with different trainable parameters $\boldsymbol{W}_U, \boldsymbol{W}_I \in \mathbb{R}^{(L \times d) \times d}$.

3.4 Behavior-Aware Denoising Module

In multi-behavior graph $\mathcal{G}$, normal interactions usually exhibit high consistency with the graph structure from a local-global unified perspective, while noisy interactions do not. Thus, once we learn the pattern-enhanced user/item representations to parameterize $\mathcal{G}$, it should be difficult to reconstruct noisy interactions from such informative representations. It follows that we can compare difficulty levels of information reconstruction to pinpoint inherent noise.

3.4.1 Graph Decoder. We design a behavior-aware graph decoder as a discriminator to perform information reconstruction. It takes the learned behavior-aware representations as input and generates the probability graph via

$$\boldsymbol{h}_{\mathcal{G}^*} = \sigma(\boldsymbol{H}_U^* \boldsymbol{e}_{b^*}^\top \boldsymbol{e}_{b^*} \boldsymbol{H}_I^{*\top}), \tag{9}$$

where $\boldsymbol{h}_{\mathcal{G}^*} \in [0, 1]^{|\mathcal{U}| \times |\mathcal{I}|}$ is the parameterized graph under behavior $*$, $\boldsymbol{H}_U^* \in \mathbb{R}^{|\mathcal{U}| \times d}, \boldsymbol{H}_I^* \in \mathbb{R}^{|\mathcal{I}| \times d}$ are the learned representations

of the user/item set, and $\boldsymbol{e}_{b^*} \in \mathbb{R}^d$ is the behavior embedding. Thus, we formulate the information reconstruction task as a binary classification task to predict user-item interactions in graph $\mathcal{G}$:

$$\mathcal{L}^*_{\text{REC}} = -\frac{1}{|\mathcal{U}||\mathcal{I}|}\sum_{u\in\mathcal{U},i\in\mathcal{I}}(\mathcal{G}^*_{ui}\log(\boldsymbol{h}_{\mathcal{G}^*_{ui}}) + (1-\mathcal{G}^*_{ui})\log(1-\boldsymbol{h}_{\mathcal{G}^*_{ui}})),$$
$$\mathcal{L}_{\text{REC}} = \frac{1}{|\{a,t\}|}\sum_{*\in\{a,t\}}\mathcal{L}^*_{\text{REC}}. \tag{10}$$

Note that we also enhance the decoder (i.e., Eq. (9)) to reconstruct the target behavior graph since target behavior (e.g., purchase) is more reliable than auxiliary behaviors (e.g., click) in real-world scenarios and can help avoid incorrect noise identification. Moreover, it is reasonable to assume that most interactions should be noiseless. Therefore, we could gradually learn informative interactions' distribution by minimizing Eq. (10). In this case, interactions with higher loss scores can be identified as noisy interactions.

3.5 Three-Stage Learning Paradigm

Based on the above components, we can obtain the learned pattern-enhanced representations and noiseless auxiliary behaviors. However, the learned parameters in the encoder may be affected by inherent noise in the original data, and the semantic gap may still exist, leading to sub-optimal knowledge learning. Consequently, we propose a *three-stage learning paradigm* in DPT. In the first stage, DPT leverages pattern-enhanced representations to guide noise identification for the following stages. In the second stage, DPT adopts a re-initializing method to attenuate the noise's influence on learned knowledge. In the third stage, DPT integrates target-specific prompt-tuning to bridge the semantic gap.

3.5.1 The First Stage of DPT. As shown in Figure 2(a), DPT takes the constructed relation graphs as input and encodes patterns into embeddings, which helps pinpoint noise. Specifically, we generate base embeddings by Eq. (1) and input them to the pattern-enhanced graph encoder (i.e., Section 3.3) to encode multi-view patterns so as to generate behavior-aware and multi-behavior representations by Eq. (8). We leverage the obtained behavior-aware representations to guide the denoising module to learn parameterized graph $\boldsymbol{h}_{\mathcal{G}^*}$ by Eq. (9) and calculate reconstruction loss $\mathcal{L}_{\text{REC}}$ by Eq. (10). We convert $\boldsymbol{h}_{\mathcal{G}^a}$ into a hard-coding binary (i.e., 0 vs. 1) distribution, which indicates whether the interaction under auxiliary behaviors is noisy or not, to generate the denoised multi-behavior graph $\mathcal{G}'$. Here we could adopt differentiable methods (e.g., Gumbel-softmax function [17, 30, 46, 50]) to binarize the real-valued graph $\boldsymbol{h}_{\mathcal{G}^a}$ in each training iteration. However, reconstructing and generating such a large-scale graph according to Eq. (9) and Eq. (10) leads to large computational costs. Motivated by sub-graph sampling [40, 51] and hyper-parameter optimizing [12, 44] strategies, we first sample a mini-batch $\mathcal{B}$ of user/item interactions to optimize Eq. (10), thus endowing the denoising module with the ability of handling large-scale graphs. Then we leverage the optimized parameters to generate the probability graph instead of generating it iteratively. Specifically, we adopt a simple threshold strategy (i.e., $\boldsymbol{h}_{\mathcal{G}^a_{ui}} = 0$ if $\boldsymbol{h}_{\mathcal{G}^a_{ui}} < 0.5 - \delta$, $\boldsymbol{h}_{\mathcal{G}^a_{ui}} = 1$ otherwise) to binarize $\boldsymbol{h}_{\mathcal{G}^a}$, where δ is a disturber to control reliability of the denoising discriminator. Note that $\delta \to 0^+$ and $\delta \to 0.5^-$ lead to more and less identified noise, respectively. After that, we can generate $\mathcal{G}' = \{\mathcal{G}^a \odot \boldsymbol{h}_{\mathcal{G}^a}, \mathcal{G}^t\}$ for the following stages of DPT, where $\odot$ is the element-wise dot product operator.

However, noise identification highly depends on the reliability of the graph encoder. A pure unsupervised task for denoising may make the optimization process irrelevant to multi-behavior recommendation. Inspired by the co-guided learning scheme [49], we incorporate a multi-behavior recommendation task into the first stage, and thus can avoid unreliable learning (e.g., false noise identification) in the early stage of the training process. Specifically, we formulate the multi-behavior recommendation task as minimizing the Bayesian Personalized Ranking (BPR) objective function:

$$\mathcal{L}_{BPR} = \frac{1}{|\{a,t\}||\mathcal{B}|}\sum_{*\in\{a,t\}}\sum_{(u,i^*,j*)\in\mathcal{B}}\mathcal{L}^*_{\text{BRP}}(u,i^*,j^*),$$
$$\mathcal{L}^*_{\text{BRP}}(u,i^*,j^*) = -\log(\sigma(\text{sim}(\boldsymbol{h}_u,\boldsymbol{h}_{i^*}) - \text{sim}(\boldsymbol{h}_u,\boldsymbol{h}_{j^*}))), \tag{11}$$

where $\text{sim}(\cdot)$ is a similarity function (e.g., inner product or a neural network), $u \in \mathcal{U}$, $i^* \in \mathcal{I}$, $\mathcal{G}^*_{ui^*} = 1$, and j^* is a randomly sampled item from $\mathcal{I}$ with $\mathcal{G}^*_{uj^*} = 0$ in each mini-batch $\mathcal{B}$.

3.5.2 The Second Stage of DPT. While representations learned in the first stage are informative, noise still inevitably affects the learned model parameters (e.g., the embedding layer) because the representation-based denoising module can capture only reliable, but not all, noise. Intuitively, we could leverage $\mathcal{G}'$ to retrain (e.g., full-tuning) the entire model. However, it is less desirable due to the large additional computational costs of model re-training. In addition, the optimized parameters in the embedding layer are informative enough to reflect pattern information, and thus there is no need to encode relation graphs in the second stage. Accordingly, we can fine-tune a few parameters instead of full-tuning the entire model to further attenuate noise's influence. Due to the disentangled design choice of aggregation layers, we can efficiently aggregate user-item interactions (i.e., by Eq. (5) and Eq. (7)) without repeatedly encoding patterns. More specifically, as shown in Figure 2(b), we freeze the parameters of the embedding layer to generate the base embeddings for users and items. Then, we input them to the user-item interaction aggregation layer (i.e., Eq. (5) and Eq. (7)) to iteratively generate multi-behavior embeddings, and learn the final multi-behavior representations (i.e., $\boldsymbol{h}'_u$ and $\boldsymbol{h}'_i$) by Eq. (8) with re-initialized parameters (i.e., $\boldsymbol{W}_U, \boldsymbol{W}_I \in \mathbb{R}^{Ld\times d}$). According to the learned representations, we rewrite the BPR loss based on the noiseless graph as follows

$$\mathcal{L}^*_{\text{BRP}}(u,i^*,j^*) = -\log(\sigma(\text{sim}(\boldsymbol{h}'_u,\boldsymbol{h}'_{i^*}) - \text{sim}(\boldsymbol{h}'_u,\boldsymbol{h}'_{j^*}))), \tag{12}$$

where $u \in \mathcal{U}$, $i^*, j^* \in \mathcal{I}$, $\mathcal{G}'_{ui^*} = 1$, and $\mathcal{G}'_{uj^*} = 0$. By minimizing Eq. (12), we denote the parameter set as $\Theta_2 = \{\boldsymbol{W}_U, \boldsymbol{W}_I\}$, which contains $2Ld \times d$ parameters. It can be seen that we only tune a small amount of parameters $2Ld \times d \ll (|\mathcal{U}| + |\mathcal{I}|) \times d$, instead of tuning the embedding layer.

3.5.3 The Third Stage of DPT. After the above stages, we can generate noiseless auxiliary behaviors and informative knowledge. However, bridging the semantic gap among multi-typed behaviors is still challenging because it requires transferring sufficient target-specific semantics without jeopardizing learned multi-behavioral knowledge. Inspired by the effectiveness of the prompt-tuning

paradigm [3, 10, 21, 38, 42], we adopt a deep continuous prompt-tuning approach in the third stage to alleviate the semantic gap between auxiliary and target behaviors. Specifically, as illustrated in Figure 2(c), we utilize the aggregation of multi-typed behavior embeddings (i.e., $\boldsymbol{e}_{b^a}$ and $\boldsymbol{e}_{b^t}$) to generate a prompt embedding $\boldsymbol{e}_p \in \mathbb{R}^d$, instead of initializing it randomly, to better understand prompt semantics [21]. Note that, during the initializing process, we freeze auxiliary behaviors' embedding parameters (i.e., $\boldsymbol{W}_{b^a}$) and update the target behavior's parameters (i.e., $\boldsymbol{W}_{b^t} \in \mathbb{R}^d$). Therefore, to adopt prompt-tuning in the third stage, we freeze not only the parameters of the embedding layer (except $\boldsymbol{W}_{b^t}$) but also the parameter set Θ_2 learned in the second stage. More specifically, to generate embeddings of the graph encoder's (l)-th layer, we only adopt prompt under the target behavior via

$$\begin{aligned} \boldsymbol{e}_{u^t}^{(l)} &= f_{i\to u}^{(l)}\Big(\sum_{i\in N_{u^t}} \boldsymbol{e}_i^{(l)}, \mathcal{G}^t, \boldsymbol{e}_p\Big); \boldsymbol{e}_{i^t}^{(l)} = f_{u\to i}^{(l)}\Big(\sum_{u\in N_{i^t}} \boldsymbol{e}_u^{(l)}, \mathcal{G}^t, \boldsymbol{e}_p\Big), \\ \boldsymbol{e}_{u^a}^{(l)} &= f_{i\to u}^{(l)}\Big(\sum_{i\in N_{u^a}} \boldsymbol{e}_i^{(l)}, \mathcal{G}^a\Big); \boldsymbol{e}_{i^t}^{(l)} = f_{u\to i}^{(l)}\Big(\sum_{u\in N_{i^a}} \boldsymbol{e}_u^{(l)}, \mathcal{G}^a\Big), \end{aligned} \tag{13}$$

where we can leverage various ways to adopt the prompt, e.g., add, concatenate, and projection operators. Inspired by VPT [10], we use the simple yet effective *add* operator to adopt the prompt in each layer. We further conduct experiments in Section 4.3 over these variants to justify our design choice. We formulate the above process via

$$\boldsymbol{e}_{u^t}^{(l)} = \boldsymbol{e}_p + \sum_{i\in N_{u^t}} \boldsymbol{e}_i^{(l)}\ ;\ \boldsymbol{e}_{i^t}^{(l)} = \boldsymbol{e}_p + \sum_{u\in N_{i^t}} \boldsymbol{e}_u^{(l)}. \tag{14}$$

We can leverage the identical process to learn the final multi-behavior representations by Eq. (8) and rewrite the BPR loss based on the noiseless graph for the target behavior via

$$\mathcal{L}_{BPR} = \frac{1}{|\mathcal{B}|} \sum_{(u,i^t,j^t)\in\mathcal{B}} -\log(\sigma(\text{sim}(\boldsymbol{h}''_u, \boldsymbol{h}''_{i^t}) - \text{sim}(\boldsymbol{h}''_u, \boldsymbol{h}''_{j^t}))). \tag{15}$$

We only optimize the objective function under the target behavior, and the prompt only has $d \ll 2L \times d \times d$ trainable parameters.

3.5.4 Model Complexity Analysis. We analyze the size of the trainable parameters in DPT at each stage. In this first stage, we update all parameters of the embedding layer and the pattern-enhanced graph encoder, which are denoted by Θ_1. We have $|\Theta_1| = (|\mathcal{U}| + |\mathcal{I}|) \times d + 4L \times (d \times d + 1)$. In the second stage, we only train a small amount of parameters, which are denoted by Θ_2. We have $|\Theta_2| = 2L \times (d \times d)$. In the third stage, we train the parameters of the prompt, denoted by Θ_3. Its size is $|\Theta_3| = d$. In conclusion, the proposed DPT can achieve comparable space complexity with state-of-the-art multi-behavior recommendation methods (e.g., CML [33]). The adopted lightweight approaches (i.e., the second and third stages of DPT) only tune/add a small number of parameters compared with the ones in the embedding layer for encoding users and items (i.e., $\mathcal{U}$ and $\mathcal{I}$ are typically of large sizes).

4 EVALUATION

In this section, we aim to answer the following research questions:

- **RQ1**: Dose the proposed DPT model outperforms other state-of-the-art multi-behavior recommendation methods?
- **RQ2**: How do different stages and behaviors of DPT contribute to the performance of target behavior recommendation?
- **RQ3**: How is the interpretation ability of the three-stage learning paradigm in DPT for denoising and target behavior recommendation?

Table 1: Statistics of experimented datasets.

Dataset	User#	Item#	Interaction#	Interactive Behavior Type
IJCAI	17,435	35,920	799,368	{Click,Favorite,Cart,Purchase}
Tmall	31,882	31,232	1,451,219	{Click,Favorite,Cart,Purchase}

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Metrics. To evaluate the effectiveness of the proposed DPT model, we conduct experiments on two public recommendation datasets: (1) **Tmall** that is collected from the Tmall E-commerce platform, and (2) **IJCAI-Contest** that is adopted in IJCAI15 Challenge from a business-to-customer retail system (referred to as IJCAI for short). These datasets have same types of behaviors, including *click*,*add-to-favorite*,*add-to-cart*, and *purchase*. Identical to previous studies [11, 33, 39], we set the purchase behavior as the target behavior, and others are considered as auxiliary behaviors. Then we filter out users whose interactions are less than 3 under the purchase behavior. Moreover, we adopt the widely used *leave-one-out* strategy by leaving users' last interacted items under the purchase behavior as the test set. Two evaluation metrics, HR (Hit Ratio) and NDCG (Normalized Discounted Cumulative Gain, N for short) @ 10, are used for performance evaluation. The statistics of the two datasets are summarized in Table 1.

4.1.2 Baselines. We compare DPT with various representative recommendation methods, including single- and multi-behavior recommendation models. *Single-behavior recommendation*: (1) **BPR** [19] is a matrix factorization model with the BPR optimization objective. (2) **PinSage** [45] uses an importance-based method to pass the message on paths constructed by a random walk. (3) **NGCF** [32] utilizes a standard convolutional message passing method to learn representations. (4) **LightGCN** [7] is a lightweight yet effective graph convolution network for representation learning. (5) **SGL** [37] performs a self-supervised learning paradigm with multi-view graph augmentation and discrimination. *Multi-behavior recommendation*: (1) **NMTR** [2] integrates prior knowledge of behavior relations into a multi-task learning framework. (2) **MATN** [39] uses memory-enhanced self-attention mechanism for multi-behavior recommendation. (3) **MBGCN** [11] leverages convolutional graph neural network to learn high-order multi-behavioral patterns. (4) **EHCF** [1] conducts knowledge transferring among heterogeneous behaviors and uses a new positive-only loss for model optimization. (5) **KHGT** [40] incorporates temporal information and item-side knowledge into the multi-behavior modeling. (6) **CML** [33] adopts meta-learning and contrastive learning paradigms to learn distinguishable behavior representations. In addition, we compare DPT with two state-of-the-art lightweight methods by applying them in multi-behavior recommendation: (1) **ADT** [31] adaptively

Table 2: Experimental results on the two datasets. The best results are boldfaced, and the second-best results are underlined.

Dataset	Metric	BPR	PinSage	NGCF	LightGCN	SGL	NMTR	MBGCN	MATN	KHGT	EHCF	CML	ADT	NoisyTune	DPT	Imprv.
IJCAI	HR	0.163	0.176	0.256	0.257	0.249	0.294	0.304	0.369	0.317	0.409	0.477	0.475	0.473	**0.490**	2.62%
	N	0.085	0.091	0.124	0.122	0.123	0.161	0.160	0.209	0.182	0.237	0.283	0.286	0.275	**0.294**	2.65%
Tmall	HR	0.243	0.274	0.322	0.342	0.350	0.362	0.381	0.406	0.391	0.433	0.543	0.542	0.545	**0.554**	1.58%
	N	0.143	0.151	0.184	0.205	0.210	0.215	0.213	0.225	0.232	0.260	0.327	0.324	0.325	**0.330**	0.92%

Table 3: Impact of different prompt-tuning methods.

Dataset	IJCAI		Tmall	
Metrics	HR	NDCG	HR	NDCG
DPT-shallow	0.472	0.274	0.536	0.316
DPT-projection	0.489	0.291	0.548	0.329
DPT-add	**0.490**	**0.294**	**0.554**	**0.330**

prunes noisy interactions for implicit feedback denoising. (2) **NoisyTune** [35] uses a matrix-wise perturbing method to empower the traditional fine-tuning paradigm.

4.1.3 Implement Details. Identical to the previous study [33], we initialize the trainable parameters with Xavier [4]. The AdamW optimizer [13] and the Cyclical Learning Rate (CLR) strategy [20] are adopted with a default base learning rate of $1e^{-3}$ and a max learning rate of $5e^{-3}$. We set the default mini-batch size to 8192. The dimension d of trainable parameters is set to 16 and 32 for IJCAI and Tmall, respectively. The L_2 regularization coefficient is searched in $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$. As suggested by CML [33], we adopt the dropout operation with a default ratio of 0.8, and set the maximum layer of the graph encoder to $L = 3$ to learn high-order information without falling into over-fitting and over-smoothing issues. Inspired by NoisyTune [35], we set the default disturber $\delta = 0.2$ to control noise identification reliability. The hyper-parameters of all competing models either follow the suggestions from the original papers or are carefully tuned, and the best performances are reported. We implement DPT in PyTorch 1.7.1, Python 3.8.3 on a workstation with an Intel Xeon Platinum 2.40GHz CPU, an NVIDIA Quadro RTX 8000 GPU, and 754GB RAM.

4.2 Performance Comparison (RQ1)

We present the main experimental results in Table 2. *Imprv* stands for the average improvements, and all improvements are significant by performing a two-sided t-test with $p < 0.05$ over the strongest baselines. We can draw a few key observations as follows:

- DPT consistently yields the best performance on all datasets. In particular, its relative improvements over the strongest baselines are 2.62% and 1.58% in terms of HR and 2.65%, 0.92% in terms of NDCG on IJCAI and Tmall, respectively. Such results generally demonstrate the superiority of our solution.
- Compared with the single-behavior recommendation methods, multi-behavior recommendation models consistently improve performance by a significant margin, which confirms the inherent inadequacy of learning from only a single type of behavior.
- Among the multi-behavior methods, DPT consistently achieves the best performance. We attribute such improvements to learning noiseless auxiliary behaviors and bridging the behavioral semantic gap, which can better understand behavior-specific information and thus generate more accurate representations.
- Compared with the denoising and fine-tuning methods, DPT can reliably learn inherent noise under auxiliary behaviors and transfer more suitable semantics to target behavior recommendation, demonstrating its effectiveness.

4.3 Ablation Study (RQ2)

To verify the contribution of each stage of DPT, we conduct an ablation study with various variants over the two datasets, including (1) *DPT-1* using only the first stage, (2) *DPT-2* using only the first and second stages, and (3) *DPT-3* (or DPT) using all the three stages. Figure 4 shows the performances of different variants in terms of HR and NDCG on IJCAI and Tmall. Red/purple dotted lines represent HR/NDCG of the strongest baselines. It can be observed that each stage positively contributes to performance. With the three-stage learning paradigm, DPT can consistently outperform the other variants. Each stage is better than the previous stage, which validates our motivation that noise and the semantic gap may mislead the target recommendation. It is also worth noting that the forward operations are more efficient than expected. Specifically, DPT-2/3 are 6x/12x faster than DPT-1 per epoch, which confirms the high efficiency of the proposed lightweight tuning approaches. While the first stage is more costly, it is a pre-training process and does not need to be performed frequently.

We further investigate different variants discussed in Eq. (14) in Table 3, including (1) *DPT-shallow* that uses prompt in the first layer of the graph encoder, (2) *DPT-projection* that generates user/item embedding vector projection by the prompt, and (3) *DPT-add* which is our choice. In all cases, DPT-add consistently outperforms the others, confirming the reasonableness of our design choice.

Moreover, we separately remove each type of behaviors to study different behaviors' importance of making recommendations. Specifically, HR (NDCG) @10 on the IJCAI dataset is: w/o click: 0.351 (0.206), w/o add-to-favorite: 0.423 (0.237), w/o add-to-cart: 0.481 (0.285). Such results demonstrate that each type of behaviors contributes to model performance. In particular, clicks are the most important signal. While being noisy, the large number of clicks can contribute the most useful information.

4.4 Case Studies of DPT's Explainability (RQ3)

Finally, we conduct a case study to illustrate how the three-stage learning paradigm of DPT can affect the target behavior recommendation. In Figure 5, we show a user whose ID is 58 and whose future purchase item ID is 2130. The user has clicked 52 items $\{186, \cdots, 2134\}$, added 4 items $\{1950, \cdots, 2125\}$ to favorite, added 4 item $\{54, \cdots, 2135\}$ to cart, and purchased 6 item $\{54, \cdots, 2129\}$ from the Tmall dataset. After the first stage of DPT, we explicitly

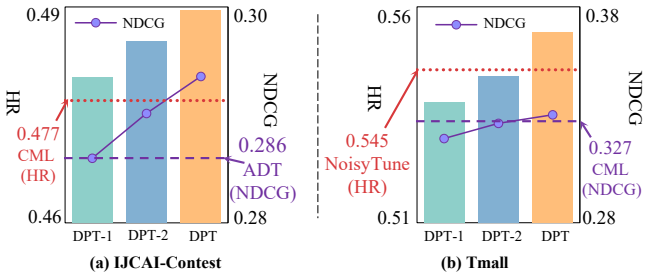

Figure 4: Impact of different stages of DPT.

remove noisy item {2102, 2129, 2120, 2100} under click and add-to-cart behaviors. After that, we utilize the rest of interactions for the following stages. Each stage of DPT consistently yields higher scores than the prior stage (i.e., scores of 1.42, 1.56, and 1.86 in each stage) to recommend item 2130 to the user under the target behavior. This aligns with our motivation that denoising auxiliary behaviors and bridging the semantic gap can boost the performance of target behavior recommendation. Moreover, after denoising, the interaction ratio and number we drop on Tmall are: click: 1.7% (18,859), favorite: 0.1% (38), cart: 0.3% (352), which shows DPT's capability of eliminating noise.

5 RELATED WORK

In view of the limited heterogeneity learning capabilities of traditional CF methods [7, 15, 19, 32, 37, 45], which typically assume that users have a single type of behaviors, multi-behavior recommendation studies [1, 2, 5, 8, 11, 33, 36, 39–41] mainly focus on learning distinguishable and representative knowledge from auxiliary behaviors to enhance the relatively sparse target behaviors. We can categorize existing studies into two lines: side-information enhanced (e.g., knowledge-driven [1, 40]) and behavior-aware balanced methods (e.g., meta-learning [33, 41]). The first line proposes to leverage additional information as prior knowledge to learn more informative representations. Despite its effectiveness, the subtle differences among multi-typed behaviors should be tackled carefully, which otherwise causes unbalanced learning of multi-behavior representations. Another line of research resorts to unsupervised learning to improve target behavior recommendation. The intuition is that a standard multi-task learning paradigm is insufficient to learn distinguishing representations under multi-typed behaviors. Therefore, how to adaptively balance the weights of multi-typed behaviors renders a crucial problem. Compared with the first line of research, behavior-aware balanced methods usually perform better since they consider the inherently unbalanced distribution of multi-typed behaviors. While these two types of methods can outperform the traditional CF approaches due to the consideration of behavior heterogeneity, the numerous uncontrollable auxiliary behaviors may introduce irrelevant (i.e., bridging semantic gap) or noisy (i.e., denoising auxiliary behaviors) information into recommenders, which deserves an in-depth exploration.

Existing denoising studies tackle the challenge of lacking supervised labels in sequential recommendations by comparing items' relevancy with a target item to explicitly remove irrelevant items [17, 22, 25, 47, 48], which are unsuitable for multi-behavior recommendation due to a different recommendation purpose. In

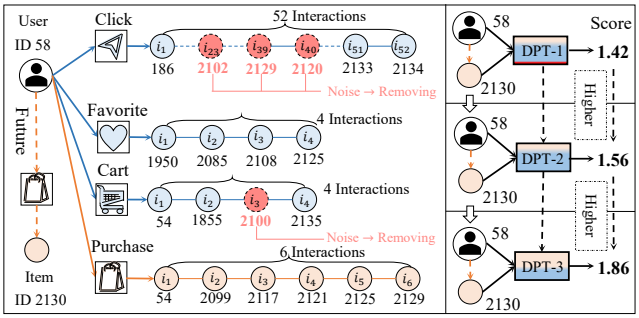

Figure 5: A case study to show how the three-stage learning paradigm affects target behavior recommendation.

contrast, some other studies [14, 31, 53] explore how to reduce noise's influence by assigning lower weights to learn representations. However, noisy interactions still exist in auxiliary behaviors and may jeopardize target behavior recommendation performance.

Prompt-tuning techniques have been widely used in various scenarios (e.g., sequential recommendation [42], fair recommendation [38], and multiple recommendation tasks [3]) to incorporate large-scale pre-trained language models (e.g., Transformer) into recommendations. Existing prompt-based recommendation methods typically focus on prompt designing and specific language model tuning, which do not undermine our technical contributions of bridging the semantic gap among multi-typed behaviors. Moreover, the typically required corpus (e.g., user reviews) hinders the adoption of such methods for multi-behavior recommendations. In contrast, DPT focuses on pinpointing noise and bridging the semantic gap in unbalanced data without requiring additional labels for multi-behavior recommendation. Thus, DPT can be seamlessly integrated into existing multi-behavior recommendation models.

6 CONCLUSION

In this paper, we studied the problem of multi-behavior recommendation from a new perspective – how to reduce the negative influences raised by the large amount of auxiliary behaviors on target behavior recommendation. We identified two critical challenges in multi-behavior recommendation: denoising auxiliary behaviors and bridging the semantic gap among multi-typed behaviors. We devised a novel DPT framework with a three-stage learning paradigm to solve the above challenges effectively and efficiently. We conducted comprehensive experiments on multiple datasets to show that our solution can consistently achieve the best performance compared with various state-of-the-art methods.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China under Grant No. 2020YFB1710200 and the National Natural Science Foundation of China under Grant No. 62072136. Xiangyu Zhao was supported by APRC-CityU New Research Initiatives (No. 9610565, Start-up Grant for New Faculty of City University of Hong Kong), SIRG-CityU Strategic Interdisciplinary Research Grant (No. 7020046, No. 7020074), HKIDS Early Career Research Grant (No. 9360163), Huawei Innovation Research Program and Ant Group (CCF-Ant Research Fund).

REFERENCES

[1] Chong Chen, Min Zhang, Yongfeng Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Efficient Heterogeneous Collaborative Filtering without Negative Sampling for Recommendation. In *AAAI*, Vol. 34. 19–26.

[2] 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 *ICDE*. IEEE, 1554–1557.

[3] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). *RecSys*.

[4] Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *AISTATS*. 249–256.

[5] Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, and Bin Cui. 2019. Buying or Browsing?: Predicting Real-Time Purchasing Intent Using Attention-Based Deep Network with Multiple Behavior. In *KDD*. 1984–1992.

[6] Qilong Han, Chi Zhang, Rui Chen, Riwei Lai, Hongtao Song, and Li Li. 2022. Multi-Faceted Global Item Relation Learning for Session-Based Recommendation. In *SIGIR*. 1705–1715.

[7] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution network for Recommendation. In *SIGIR*. 639–648.

[8] Chao Huang. 2021. Recent Advances in Heterogeneous Relation Learning for Recommendation. In *IJCAI*.

[9] Chao Huang, Huance Xu, Yong Xu, Peng Dai, Lianghao Xia, Mengyin Lu, Liefeng Bo, Hao Xing, Xiaoping Lai, and Yanfang Ye. 2021. Knowledge-Aware Coupled Graph Neural Network for Social Recommendation. In *AAAI*, Vol. 35. 4115–4122.

[10] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual Prompt Tuning. In *ECCV*.

[11] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-Behavior Recommendation with Graph Convolutional Networks. In *SIGIR*. 659–668.

[12] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2019. Darts: Differentiable Architecture Search. In *ICLR*.

[13] Ilya Loshchilov and Frank Hutter. 2017. Decoupled Weight Decay Regularization. In *ICLR*.

[14] Anjing Luo, Pengpeng Zhao, Yanchi Liu, Fuzhen Zhuang, Deqing Wang, Jiajie Xu, Junhua Fang, and Victor S. Sheng. 2020. Collaborative Self-Attention Network for Session-based Recommendation. In *IJCAI*. 2591–2597.

[15] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: Ultra Simplification of Graph Convolutional Networks for Recommendation. In *CIKM*. 1253–1262.

[16] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A Novel Embedding Model for Knowledge Base completion Based on Convolutional Neural Network. In *NAACL-HLT*. 327–333.

[17] Yuqi Qin, Pengfei Wang, and Chenliang Li. 2021. The world is Binary: Contrastive Learning for Denoising Next Basket Recommendation. In *SIGIR*. 859–868.

[18] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social Collaborative Viewpoint Regression with Explainable Recommendations. In *WSDM*. 485–494.

[19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and ST Lars. 2009. Bpr: Bayesian Personalized Ranking from Implicit Feedback. *UAI* (2009), 452–461.

[20] Leslie N Smith. 2017. Cyclical Learning Rates for Training Neural Networks. In *WACV*. IEEE, 464–472.

[21] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. 2022. GPPT: Graph Pre-Training and Prompt Tuning to Generalize Graph Neural Networks. In *KDD*. 1717–1727.

[22] Yatong Sun, Bin Wang, Zhu Sun, and Xiaochun Yang. 2021. Does Every Data Instance Matter? Enhancing Sequential Recommendation by Eliminating Unreliable Data. In *IJCAI*. 1579–1585.

[23] Ye Tao, Ying Li, Su Zhang, Zhirong Hou, and Zhonghai Wu. 2022. Revisiting Graph based Social Recommendation: A Distillation Enhanced Social Graph Network. In *WWW*. 2830–2838.

[24] Gabriele Tolomei, Mounia Lalmas, Ayman Farahat, and Andrew Haines. 2019. You Must Have Clicked on This Ad by Mistake! Data-Driven Identification of Accidental Clicks on Mobile Ads with Applications to Advertiser Cost Discounting and Click-Through Rate Prediction. *IJDSA* 7, 1 (2019), 53–66.

[25] Xiaohai Tong, Pengfei Wang, Chenliang Li, Long Xia, and Shaozhang Niu. 2021. Pattern-Enhanced Contrastive Policy Learning Network for Sequential Recommendation. In *IJCAI*. 1593–1599.

[26] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[27] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Make it a Chorus: Knowledge-and Time-Aware Item Modeling for Sequential Recommendation. In *SIGIR*. 109–118.

[28] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. 2021. HyperSoRec: Exploiting Hyperbolic user and Item Representations with Multiple Aspects for Social-Aware Recommendation. *TOIS* 40, 2 (2021), 1–28.

[29] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-Aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In *KDD*. 968–977.

[30] Qinyong Wang, Hongzhi Yin, Hao Wang, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2019. Enhancing Collaborative Filtering with Generative Augmentation. In *KDD*. 548–556.

[31] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *WSDM*. 373–381.

[32] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.

[33] Wei Wei, Chao Huang, Lianghao Xia, Yong Xu, Jiashu Zhao, and Dawei Yin. 2022. Contrastive Meta Learning with Behavior Multiplicity for Recommendation. In *WSDM*. 1120–1128.

[34] Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. 2020. Entire Space Multi-Task Modeling via Post-Click Behavior Decomposition for Conversion Rate Prediction. In *SIGIR*. 2377–2386.

[35] Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2022. NoisyTune: A Little Noise Can Help You Finetune Pretrained Language Models Better. In *ACL*.

[36] Chuhan Wu, Fangzhao Wu, Tao Qi, Qi Liu, Xuan Tian, Jie Li, Wei He, Yongfeng Huang, and Xing Xie. 2022. FeedRec: News Feed Recommendation with Various User Feedbacks. In *WWW*. 2088–2097.

[37] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-Supervised Graph Learning for Recommendation. In *SIGIR*. 726–735.

[38] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Fuzhen Zhuang, Ao Xiang, Xu Zhang, Leyu Lin, and Qing He. 2022. Selective Fairness in Recommendation via Prompts. In *SIGIR*. 2657–2662.

[39] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Bo Zhang, and Liefeng Bo. 2020. Multiplex Behavioral Relation Learning for Recommendation via Memory Augmented Transformer Network. In *SIGIR*. 2397–2406.

[40] 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 *AAAI*, Vol. 35. 4486–4493.

[41] Lianghao Xia, Yong Xu, Chao Huang, Peng Dai, and Liefeng Bo. 2021. Graph Meta Network for Multi-Behavior Recommendation. In *SIGIR*. 757–766.

[42] Xin Xin, Tiago Pimentel, Alexandros Karatzoglou, Pengjie Ren, Konstantina Christakopoulou, and Zhaochun Ren. 2022. Rethinking Reinforcement Learning for Recommendation: A Prompt Perspective. In *SIGIR*. 1347–1357.

[43] Chen Xu, Quan Li, Junfeng Ge, Jinyang Gao, Xiaoyong Yang, Changhua Pei, Fei Sun, Jian Wu, Hanxiao Sun, and Wenwu Ou. 2020. Privileged Features Distillation at Taobao Recommendations. In *KDD*. 2590–2598.

[44] Huaxiu Yao, Yu Wang, Ying Wei, Peilin Zhao, Mehrdad Mahdavi, Defu Lian, and Chelsea Finn. 2021. Meta-Learning with an Adaptive Task Scheduler. *NIPS*, 7497–7509.

[45] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.

[46] Junliang Yu, Min Gao, Hongzhi Yin, Jundong Li, Chongming Gao, and Qinyong Wang. 2019. Generating Reliable Friends via Adversarial Training to Improve Social Recommendation. In *ICDM*. IEEE, 768–777.

[47] Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual Sparse Attention Network For Session-Based Recommendation. In *AAAI*, Vol. 35. 4635–4643.

[48] Chi Zhang, Yantong Du, Xiangyu Zhao, Qilong Han, Rui Chen, and Li Li. 2022. Hierarchical Item Inconsistency Signal Learning for Sequence Denoising in Sequential Recommendation. In *CIKM*. 2508–2518.

[49] Xiaokun Zhang, Bo Xu, Liang Yang, Chenliang Li, Fenglong Ma, Haifeng Liu, and Hongfei Lin. 2022. Price DOES Matter! Modeling Price and Interest Preferences in Session-based Recommendation. In *SIGIR*.

[50] Xiangyu Zhao, Haochen Liu, Hui Liu, Jiliang Tang, Weiwei Guo, Jun Shi, Sida Wang, Huiji Gao, and Bo Long. 2021. AutoDim: Field-aware Embedding Dimension Searchin Recommender Systems. In *WWW*. 3015–3022.

[51] Jiawei Zheng, Qianli Ma, Hao Gu, and Zhenjing Zheng. 2021. Multi-View Denoising Graph Auto-Encoders on Heterogeneous Information Networks for Cold-Start Recommendation. In *KDD*. 2338–2348.

[52] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *KDD*. 1059–1068.

[53] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-Enhanced MLP is All You Need for Sequential Recommendation. In *WWW*. 2388–2399.

[54] Xiangmin Zhou, Dong Qin, Xiaolu Lu, Lei Chen, and Yanchun Zhang. 2019. Online Social Media Recommendation over Streams. In *ICDE*. IEEE, 938–949.

[55] Qiannan Zhu, Xiaofei Zhou, Zeliang Song, Jianlong Tan, and Li Guo. 2019. Dan: Deep Attention Neural Network for News Recommendation. In *AAAI*, Vol. 33. 5973–5980.