

# Combinatorial Optimization Perspective based Framework for Multi-behavior Recommendation

Chenhao Zhai\*<sup>†</sup>  
Shenzhen International Graduate  
School, Tsinghua University  
Shenzhen, China  
dch23@mails.tsinghua.edu.cn

Chang Meng\*  
Kuaishou Technology  
Beijing, China  
mengchang@kuaishou.com

Yu Yang  
Shenzhen International Graduate  
School, Tsinghua University  
Shenzhen, China  
yu-yang23@mails.tsinghua.edu.cn

Kexin Zhang  
Shenzhen International Graduate  
School, Tsinghua University  
Shenzhen, China  
zkx21@mails.tsinghua.edu.cn

Xuhao Zhao  
School of Electronic Information and  
Electrical Engineering, Shanghai Jiao  
Tong University  
Shanghai, China  
zhaoxuhao@sjtu.edu.cn

Xiu Li<sup>‡</sup>  
Shenzhen International Graduate  
School, Tsinghua University  
Shenzhen, China  
li.xiu@sz.tsinghua.edu.cn

## Abstract

In real-world recommendation scenarios, users engage with items through various types of behaviors. Leveraging diversified user behavior information for learning can enhance the recommendation of target behaviors (e.g., buy), as demonstrated by recent multi-behavior methods. The mainstream multi-behavior recommendation framework consists of two steps: fusion and prediction. Recent approaches utilize graph neural networks for multi-behavior fusion and employ multi-task learning paradigms for joint optimization in the prediction step, achieving significant success. However, these methods have limited perspectives on multi-behavior fusion, which leads to inaccurate capture of user behavior patterns in the fusion step. Moreover, when using multi-task learning for prediction, the relationship between the target task and auxiliary tasks is not sufficiently coordinated, resulting in negative information transfer. To address these problems, we propose a novel multi-behavior recommendation framework based on the combinatorial optimization perspective, named COPF. Specifically, we treat multi-behavior fusion as a combinatorial optimization problem, imposing different constraints at various stages of each behavior to restrict the solution space, thus significantly enhancing fusion efficiency (COGCN). In the prediction step, we improve both forward and backward propagation during the generation and aggregation of multiple experts to mitigate negative transfer caused by differences in both feature and label distributions (DFME). Comprehensive experiments on three real-world datasets indicate the superiority of COPF. Further analyses also validate the effectiveness of the COGCN and DFME modules. Our code is available at <https://github.com/1918190/COPF>.

\*Both authors contributed equally to this research.

<sup>†</sup>Work done when he was a research intern at Kuaishou Technology.

<sup>‡</sup>The corresponding author.

## CCS Concepts

• Information systems → Recommender systems.

## Keywords

Combinatorial Optimization; Multi-behavior Recommendation; Multi-task

## ACM Reference Format:

Chenhao Zhai, Chang Meng, Yu Yang, Kexin Zhang, Xuhao Zhao, and Xiu Li. 2025. Combinatorial Optimization Perspective based Framework for Multi-behavior Recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25)*, August 3–7, 2025, Toronto, ON, Canada. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3690624.3709278>

## 1 INTRODUCTION

Recommender system is a crucial technology in today's society [8, 26, 30, 44, 54], delivering high-quality personalized recommendations based on user preferences. Delving into users' historical engagements, traditional collaborative filtering (CF) techniques [39] learn the representations of users and items for improved recommendations. Although somewhat effective, these methods only account for a single type of user-item interaction, limiting their practical effectiveness. In the real world, user behavior is diverse. Beyond target behavior (e.g., *buy*), which is the primary focus for businesses and platforms, users also engage in behaviors such as viewing, adding to cart, and collecting. This behavioral information encompasses different dimensions of user preferences that can be leveraged as auxiliary knowledge to enhance the learning of target behaviors and provide better service for users [10, 21, 47].

To fully exploit auxiliary behaviors, multi-behavior recommendation methods have emerged as solutions. These methods can be divided into two steps: multi-behavior fusion and multi-behavior prediction [18, 31]. In the fusion step, multiple behaviors are combined to learn the representations that capture user preferences. In the prediction step, these representations are applied for model prediction, with multi-task learning (MTL) proving effective [19].

With the explosive development of deep learning, most methods employed for multi-behavior fusion have transitioned from



This work is licensed under a Creative Commons 4.0 International License.

*KDD '25, August 3–7, 2025, Toronto, ON, Canada*

© 2025 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-1245-6/25/08

<https://doi.org/10.1145/3690624.3709278>

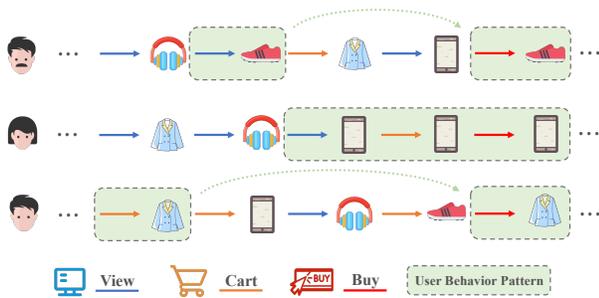


Figure 1: Examples of the user behavior patterns.

traditional matrix factorization [23, 37, 41] to deep neural networks [10, 14, 47]. Among them, graph neural networks (GNNs) [5, 16, 32, 33, 43] have become popular techniques in multi-behavior recommendation due to their ability to model higher-order user-item interactions effectively [4, 31, 46, 48]. For example, MBGCN [21] and CIGF [15] learn different behavioral information synchronously and fuse them through learnable parameters. Additionally, some studies [6, 31, 51] leverage the cascading dependencies between behaviors in the real world (e.g., *view*  $\rightarrow$  *cart*  $\rightarrow$  *buy*) to enhance model learning. BCIPM [52] emphasizes the target behavior by strategically modeling different behaviors. These above methods essentially share the same goal: *capturing richer user preferences by deeply exploring the complex user behavior patterns formed by heterogeneous interactions (shown in Figure 1)*. However, previous methods either simply aggregate the behavior representations without constraints, or define strict sequential relationships for the behaviors, resulting in inadequate modeling of user behavior patterns.

In the multi-behavior prediction step, MTL modules are widely utilized [45, 49, 51, 52] to overcome the limitation of single label [12, 21] in representing diverse user preferences. These modules incorporate auxiliary behavior labels for joint optimization, allowing the model to leverage multi-behavior information for improved accuracy. To better adapt to multi-behavior prediction tasks, CIGF [15] enhances traditional MTL models [29, 40] by further decoupling the inputs of different tasks to mitigate gradient conflicts. PKEF [31] builds on this by introducing a projection mechanism during aggregation, preventing the incorporation of harmful information.

Although the above multi-behavior recommendation methods have demonstrated effectiveness in multi-behavior fusion and prediction steps respectively, the following challenges still exist:

- Combination optimization for Multi-Behavior Fusion.** Combination optimization problems typically involve numerous states and choices, necessitating an optimal solution within manageable complexity. Multi-behavior recommendation is essentially a combination optimization problem. For a given behavioral category, each user has a finite number of possible user behavior patterns. The optimal behavioral pattern can be yielded by exhaustively enumerating all possibilities, but this results in significant spatial and temporal costs, known as "combinatorial explosion." Therefore, the challenge lies in leveraging existing knowledge to restrict the solution space and achieve efficient multi-behavior recommendations. Early approaches [12, 16] focused solely on learning behavior-specific information without considering user behavior patterns, which was improved by methods [15, 21, 52]

that incorporated behavior aggregation during the learning process. However, they still lacked adequate constraints on user behavior patterns. Some recent methods [6, 31, 51] use a cascading paradigm to model user behavioral sequences, resulting in overly strict constraints on user behavior patterns. In conclusion, a paradigm that establishes appropriate constraints to restrict the solution space urgently needs to be proposed.

- Coordination of Correlations between Tasks.** Multi-task learning (MTL) is a commonly used approach for multi-behavior prediction, which models different behaviors as independent tasks. Since each task can affect the final prediction, it is crucial to properly coordinate the correlations between tasks, which is currently limited by two factors:
  - Differences in feature space distribution during forward propagation:** Most existing methods overlook this aspect, leading to biases in information aggregation. While recent approaches have been devoted to mitigating inconsistencies in feature distribution such as the projection-based aggregation mechanism [31], they do not address the dynamics of the representation space during training (details are illustrated in Appendix A.3.2).
  - Differences in label space distribution during backward propagation:** Existing methods learn information from different behaviors but encounter conflicts during gradient updates due to differences in label distributions. Previous methods [15, 31] have attempted to address gradient conflicts using decoupled inputs, with the issue of gradient coupling in aggregation still existing. These factors can lead to negative transfer problems [42].

To address these two challenges, we propose a **Combinatorial Optimization Perspective based Framework for Multi-behavior Recommendation (COPF)**. It consists of the Combinatorial Optimization Graph Convolution Network (COGCN) and the Distributed Fitting Multi-Expert Networks (DFME). To tackle the combinatorial optimization problem in the fusion step, COGCN restricts the solution space of combinatorial optimization by imposing different degrees of constraints to user behavior patterns across various stages (*Pre-behavior, In-behavior, Post-behavior*) based on graph convolutional networks, thus achieving efficient multi-behavior fusion.

To coordinate the correlations between tasks, DFME improves the forward and backward propagation processes during the multi-behavior prediction phase from two perspectives: feature and label. At the feature level, DFME regards different behaviors as independent tasks, and utilizes contrastive learning to adaptively align the distributions of target and auxiliary behaviors. Considering that behavior aggregation can be affected by differences in behavior feature distributions, DFME incorporates a specialized behavior-fitting expert to refine the representation space for each behavior before aggregation, thereby diminishing distribution bias while maintaining spatial generalization. At the label level, DFME further decouples the gradient between the target and auxiliary behaviors during aggregation, avoiding the influence of other tasks on the gradient updates of the target task. The above designs enable the effective utilization of auxiliary tasks to adjust the model-fitted data distribution to be more consistent with the target task's test distribution, alleviating the negative transfer problem caused by uncoordinated task relationships (explained in Appendix A.3.3).

In summary, the main contributions of our works are as follows:

- To our knowledge, we are the first to propose examining the multi-behavior fusion problem from a combinatorial optimization perspective. Specifically, we highlight the benefits of behavioral constraints for multi-behavior recommendation and analyze the limitations of existing methods in this perspective, and then propose Combinatorial Optimization Graph Convolutional Network (COGCN) as a solution. It applies different degrees of constraints to user behavior patterns across various stages, effectively facilitating the process of multi-behavior fusion.
- We investigate the limitations of multi-task methods in multi-behavior recommendation from structural perspectives (i.e., the feature and label perspectives) and propose Distributed Fitting Multi-Expert Networks (DFME). It is designed to coordinate task correlations by improving the forward and backward propagation processes, thus alleviating the negative transfer problem in MTL.
- We conduct comprehensive experiments on three real-world datasets, demonstrating that our proposed COPF has superior performance in multi-behavior recommendation. Further experimental results also verify the rationality and effectiveness of COGCN and DFME.

## 2 RELATED WORK

**Multi-behavior Recommendation.** Multi-behavior recommendation methods aim to leverage the multiple behavior signals of users to alleviate the issue of data sparsity in target behavior. Early multi-behavior methods extend matrix factorization to accommodate multi-behavior data [23, 37, 41], such as CMF [55]. Besides, some other methods use auxiliary behavioral signals to design new sampling strategies [9, 13, 28, 34]. But none of them exploits deep behavioral information.

With the rise of deep learning [7, 25, 27, 53], deep neural networks (DNNs) and graph convolutional networks (GCNs) have been proven to be more suitable approaches for multi-behavior fusion, as they can delve deeper into the multiple information of different behaviors. DNN-based methods often heavily utilize neural networks to extract information from user-item interactions, and the information is embedded within the representations. For example, DIPN [14] and MATN [47] capture the implicit relationship between behaviors through attention mechanism. NMTR [10] treats all behaviors as prediction targets and transfers prediction results between behaviors. However, most DNN-based models fail to capture high-order relationships, resulting in poor performance.

In contrast, GCN-based models can learn higher-order relations between users and items, making them the most popular methods for multi-behavior recommendation currently. This type of method mainly learns behavioral information through graph convolution and considers capturing user behavior patterns by holistic modeling of multiple relationships between users and items. S-MBRec [12] models GCN for each behavior individually, focusing solely on the information of the current behavior without learning user patterns. MBGCN [21], CML [45] and CIGF [15] further aggregate representations between behaviors through learnable parameters, with no constraints on user behavior patterns. Recent methods CRGCN [51] and MB-CGCN [6] take into account the hierarchical correlation between behaviors and employ cascading behavior network, and PKEF [31] further considers the bias in cascading networks, but

this cascading paradigm imposes overly strict constraints on user behavior patterns. BCIPM [52] successively learns global behavior information and target behavior information, which only considers patterns before the target behavior. Therefore, existing methods do not adequately model user behavior patterns.

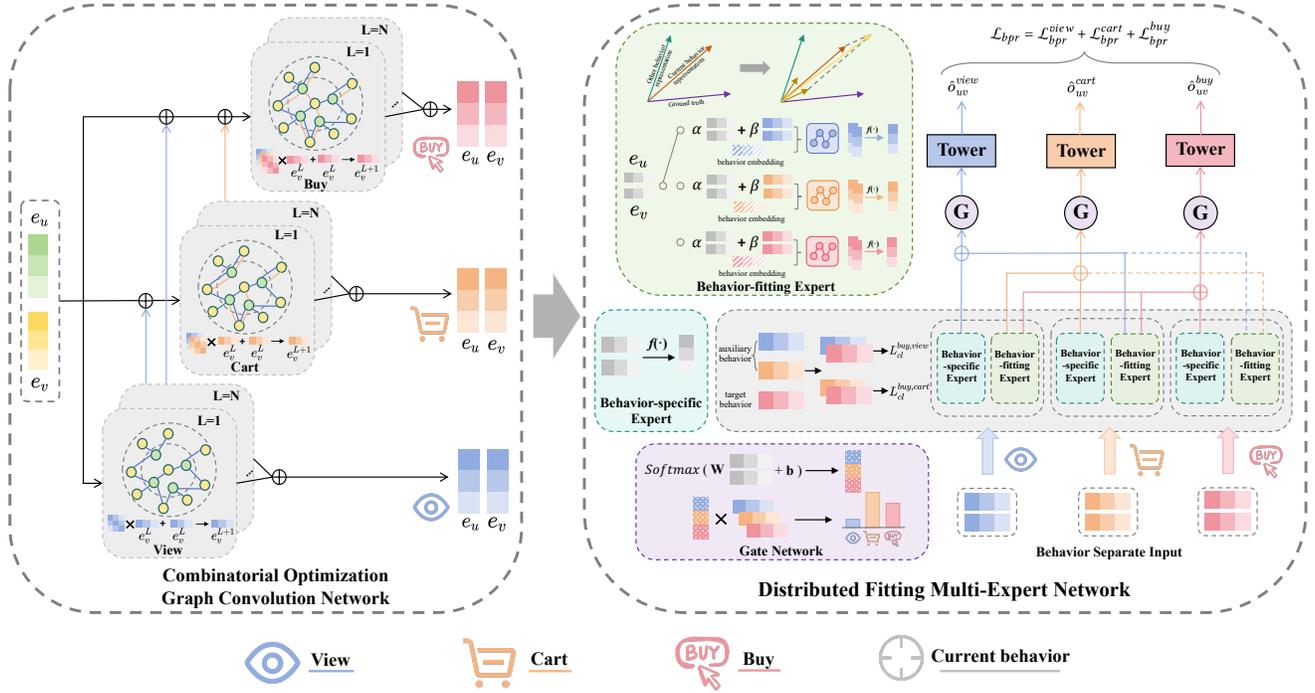
**MTL for Recommendation.** Recent research works have extensively applied multi-task learning (MTL) [20, 24, 38] methods to recommendation systems to leverage heterogeneous user information. The traditional multi-task learning method is the shared bottom [3] structure, which shares the bottom network to learn representations and uses separate tower network to predict each task. While some multi-behavior approaches based on this structure [4, 45, 49, 51] can achieve knowledge sharing between tasks, their effectiveness may be affected by task differences. To better jointly optimize the model, some methods have introduced attention-based gating mechanisms into MTL structures. For example, MOE [20] and MMOE [29] propose a multi-expert structure shared by all tasks, and utilize gating networks to obtain expert fusion weights for each task. PLE [40] further divides experts into shared experts and task-specific experts. However, the above methods all share inputs between tasks, which can lead to negative information transfer due to gradient conflict. MESI in CIGF [15] specifically decouples inputs between tasks, mitigating the negative impact of coupled gradients. PME in PKEF [31] further introduces a projection mechanism during task fusion to eliminate harmful information. Nevertheless, none of these approaches fully alleviate the negative transfer problem caused by the aggregation of tasks with differing feature and label distributions from a structural perspective.

## 3 PRELIMINARY

In our framework, we use  $u$  and  $v$  to represent a user and an item, respectively. The user set and item set are denoted as  $\mathbf{U} = \{u_1, u_2, \dots, u_M\}$  and  $\mathbf{V} = \{v_1, v_2, \dots, v_N\}$ , where  $M$  and  $N$  represent the total number of users and items, respectively. The behavior types  $k \in \{1, 2, \dots, K\}$  maintain a consistent order among behaviors (i.e., 1 and  $K$  correspond to the most upstream and downstream behaviors, respectively). The user-item interaction matrices for the  $K$  behavior types can be represented as  $\mathcal{B} = \{\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_K\}$ , where  $\mathbf{B}_k = [b_{(k)uv}]_{|\mathbf{U}| \times |\mathbf{V}|} \in \{0, 1\}$  indicates whether the user  $u$  interacts with the item  $v$  under behavior  $k$ . Additionally, we define a user-item bipartite graph  $\mathcal{G} = (\mathcal{H}, \mathcal{E}, \mathcal{B})$  to represent the various interaction data between users and items, where  $\mathcal{H} = \mathbf{U} \cup \mathbf{V}$ ,  $\mathcal{E} = \cup_{k=1}^K \mathcal{E}_k$  is the edge set including all interactions. For multi-behavior recommendation, there is a specific target behavior (e.g., buy) to be optimized, and other behaviors are considered auxiliary behaviors to assist in predicting the target behavior. The target behavior is the most downstream behavior (behavior  $K$ ).

## 4 METHODOLOGY

We devise a "Combinatorial Optimization Perspective based Framework" (COPF) for multi-behavior recommendation, which contains two parts: (1) Combinatorial Optimization Graph Convolutional Network (COGCN); (2) Distributed Fitting Multi-Expert Network (DFME). Figure 2 illustrates the overall architecture of the proposed framework.



**Figure 2: Illustration of the proposed COPF framework, and we use three user behavior types as examples: view, cart, and buy. ( $\oplus$ ) denotes the element-wise addition operation,  $f(\cdot)$  represents the function to generate experts, and dotted lines represent stop gradient.**

#### 4.1 Embedding Layer

We first look up the low-dimensional dense embeddings for user  $u$  and item  $v$  from the embedding tables using their one-hot vectors, respectively. Specifically, the process for obtaining these embeddings can be formulated as follows:

$$\mathbf{e}_u = \mathbf{E}_u^T \cdot \text{ID}_u \in \mathbb{R}^d, \mathbf{e}_v = \mathbf{E}_v^T \cdot \text{ID}_v \in \mathbb{R}^d \quad (1)$$

where  $\text{ID}_u$  and  $\text{ID}_v$  denotes the one-hot vectors of user  $u$  and item  $v$ ,  $\mathbf{E}_u \in \mathbb{R}^{|\mathbf{U}| \times d}$  and  $\mathbf{E}_v \in \mathbb{R}^{|\mathbf{V}| \times d}$  are the embedding tables for users and items,  $|\mathbf{U}|$  and  $|\mathbf{V}|$  are the total number of users and items, and  $d$  is the embedding size.

#### 4.2 Combinatorial Optimization Graph Convolution Network

The recent multi-behavior methods have aimed to deeply explore user behavior patterns to enhance model performance. However, they impose constraints on user behavior patterns that are either too strict or too slack during the fusion step, making it difficult for the model to accurately capture these patterns.

To solve the above problem, we examine the multi-behavior fusion process from a combinatorial optimization perspective and propose the Combinatorial Optimization Graph Convolutional Network (COGCN). It imposes different degrees of constraints on behavior patterns at different stages of user behavior, so as to learn the optimal behavioral information.

**4.2.1 Definition of Constraints in Combinatorial Optimization.** To better describe the constraints at different stages of behavior, we first introduce the following definitions:

**Definition 1 (User Behavior Pattern).** User behavior patterns are defined as high-frequency behavior chains between users and all items on the platform, representing the user's personalized habits. Formally, for any user  $u \in \mathbf{U}$ , his/her user behavior pattern is:  $u \rightarrow \dots \rightarrow b_k \rightarrow \dots \rightarrow \mathbf{V}$ , where  $b_k$  is the behavior  $k$  and  $K$  is the number of behavior types. For example, for a user who directly buys items that they find appealing while viewing, his/her user behavior pattern is:  $u \rightarrow \text{view} \rightarrow \text{buy} \rightarrow \mathbf{V}$ .

**Definition 2 (Upstream and Downstream Behavior).** The order of behavior definition is often derived from the mainstream behavior habits of most users in the real world. For example, consider the combination  $\text{view} \rightarrow \text{cart} \rightarrow \text{buy}$ . Upstream behavior refers to the behavior that precedes the current behavior. Similarly, downstream behavior refers to the behavior that follows the current behavior. In the example,  $\text{view}$  is the upstream behavior of  $\text{cart}$ , and  $\text{buy}$  is the downstream behavior of  $\text{cart}$ .

It can be seen that for the number of behavior types  $K$ , the total number of possible user behavior patterns is  $\sum_{i=1}^K \frac{K!}{(K-i)!}$ . To restrict the solution space in the combinatorial optimization of multi-behavior recommendation systems, we impose constraints on user behavior patterns at three stages: pre-behavior, in-behavior, and post-behavior. The details are as follows.

**Definition 3 (Pre-behavior Constraint).** Many users interact with items in a predetermined order of behaviors. Therefore, the upstream behavior information of the current behavior is essential. Formally, for any behavior  $k (k > 1)$ , The input of its encoder  $s_{input}^k$  is denoted as  $s_{input}^k = f(g(\sum_{k'=1}^{k-1} s_{output}^{k'}, s_{init}))$ , where  $s_{init}$  is the initial input, and  $s_{output}^{k'}$  is the information of the behavior  $k'$ ,

which is actually the output of its encoder.  $g(\cdot)$  is the aggregation function (e.g., summation) and  $f(\cdot)$  is the transition function (e.g., matrix multiplication, graph convolution). In this way, we constrain the interaction between behaviors.

**Definition 4 (In-behavior Constraint).** In-behavior Constraints essentially pertain to the modeling of the current behavior (i.e., the transition function  $f(\cdot)$ ). In order to capture more complex or implicit user behavior patterns (for example, a user alternates between *view* and *cart* before *buy*), we similarly utilize GCN with heterogeneous relations [21]. Meanwhile, to prevent poor model generalization performance and overfitting problems caused by information leakage, we define that the current behavior node learning process cannot contain semantic information of downstream behaviors. Specifically, for the current behavior  $k$ , we have the output of its encoder:

$$\mathbf{s}_{output}^k = \text{Agg}(\mathbf{s}_{input}^k, \{\mathbf{B}_{k'} | \mathbf{B}_{k'} \in \mathcal{B}, 1 \leq k' \leq k\}) \quad (2)$$

where  $\mathbf{s}_{input}^k = g(\sum_{k'=1}^{k-1} \mathbf{s}_{output}^{k'}, \mathbf{s}_{init})$ .

**Definition 5 (Post-behavior Constraint).** For each behavior, we design decoupled outputs to partition the solution space, modeling user behavior patterns that end with different behaviors. Through joint optimization, the model can achieve better performance. This is consistent with the perspective of our proposed DFME.

**4.2.2 Graph Convolution.** In the previous part, we outlined the specific constraints for different behavioral stages. As graph convolutional networks (GCNs) can efficiently utilize high-order connectivity between users and items, we use a GCN-based paradigm to model the multi-behavior information fusion in the combinatorial optimization perspective.

Given the adjacency matrices of different behaviors, we modify them to meet the requirements of graph convolution:

$$\mathbf{A}_k = \begin{pmatrix} 0 & \mathbf{B}_k \\ (\mathbf{B}_k)^T & 0 \end{pmatrix} \quad (3)$$

where  $\mathbf{A}_k$  is the adjacency matrix of behavior  $k$  in the graph. For the same purpose, we obtain the embedding matrices for users and items, respectively:

$$\mathbf{E}_u = [ \mathbf{e}_{u_1}, \dots, \mathbf{e}_{u_{|U|}} ], \mathbf{E}_v = [ \mathbf{e}_{v_1}, \dots, \mathbf{e}_{v_{|V|}} ], \quad (4)$$

we then capture the interaction information of behaviors through graph convolution. Inspired by [16, 21], for behavior  $k$ , we have:

$$\mathbf{E}^{k,l+1} = \sum_{k'=1}^k (\mathbf{D}^{-1} \mathbf{A}_{k'} + \mathbf{I}) \mathbf{E}^{k,l} \quad (5)$$

where  $\mathbf{D}$  is the diagonal identity matrix,  $\mathbf{I}$  denotes an identity matrix.  $\mathbf{E}^{k,l} = \mathbf{E}_u^{k,l} || \mathbf{E}_v^{k,l}$ , ( $||$ ) is the concatenate operation and  $l$  denotes the  $l$ -th layer. the initial input of the model  $\mathbf{E}^{1,0} = \mathbf{E}_u || \mathbf{E}_v$ . We utilize the adjacency matrices of the current behavior and its upstream behavior for message propagation and aggregation on each layer  $l$ , thereby implementing the *In-behavior Constraint*.

Further, in order to implement the *Pre-behavior Constraint*, we define the hierarchical information transfer between behaviors as:

$$\mathbf{E}^{k+1,0} = \sum_{k'=1}^k \mathbf{E}^{k',L} + \mathbf{E}^{1,0} \quad (6)$$

where  $L$  denotes the total layers of GCN. Here, we combine the last layer representations of each upstream behavior representation with the initial representation as the input of the current behavior.

Follow the *Post-behavior Constraint*, We independently output the representations of each behavior for subsequent multi-task learning. To be specific, we directly add the outputs of different layers to get relations of different orders. For the embeddings of user  $u$  and item  $v$  in  $\mathbf{E}^{k,l}(\mathbf{e}_u^{k,l}$  and  $\mathbf{e}_v^{k,l})$ , we have:

$$\mathbf{e}_u^{k,*} = \sum_{l=0}^L \mathbf{e}_u^{k,l}, \mathbf{e}_v^{k,*} = \sum_{l=0}^L \mathbf{e}_v^{k,l} \quad (7)$$

where  $L$  is the number of GCN layers.

### 4.3 Distributed Fitting Multi-Expert Network

By employing COGCN in the multi-behavior fusion step, we have obtained representations for user  $u$  and item  $v$  under each behavior  $k$ . The subsequent task is to devise a proper structure for multi-behavior prediction. Many methods [15, 31, 51] have utilized MTL modules to fully leverage multi-behavior information to assist in predicting target behavior, which has demonstrated their effectiveness. However, these MTL methods exhibit insufficient exploration in their structural design, failing to account for the potential negative transfer effects caused by differences in feature and label distributions during the learning process. To handle the drawbacks of the existing MTL modules, we propose the Distributed Fitting Multi-Expert Network(DFME), which controls behavior interactions through both features and labels, thereby coordinating the relationships between tasks. The specific details are as follows.

**4.3.1 Generating of Behavior-specific Experts.** As contrastive learning can alleviate distributional biases between different data sources, we utilize it to adaptively learn the distributional similarity between target behaviors and auxiliary behaviors before generating experts. Take the auxiliary behavior  $k$  as an example, we have:

$$\mathcal{L}_{cl,U}^{K,k} = \frac{1}{|U|} \sum_{u \in U} -\log \frac{\exp(\varphi(\mathbf{e}_u^{K,*}, \mathbf{e}_u^{k,*})/\tau)}{\sum_{u' \in U} \exp(\varphi(\mathbf{e}_u^{K,*}, \mathbf{e}_{u'}^{k,*})/\tau)} \quad (8)$$

where  $\tau$  represents the temperature hyperparameter for the softmax function, and  $\varphi(\cdot)$  is a function for calculating the similarity between two vectors(e.g., inner product). The item side follows the same contrastive learning process. Thus, for behavior  $k$ , the final contrastive loss is  $\mathcal{L}_{cl}^{K,k} = \mathcal{L}_{cl,U}^{K,k} + \mathcal{L}_{cl,V}^{K,k}$ .

Then, we follow previous methods[15] by using decoupled behavior representations to generate behavior-specific experts, thereby preventing gradient conflicts caused by coupled inputs:

$$\mathbf{e}^k = \mathbf{e}_u^{k,*} \circ \mathbf{e}_v^{k,*} \quad (9)$$

where ( $\circ$ ) is the hadamard product operation. Since decoupled behavior representations are utilized to generate experts, we can obtain a total of  $k$  behavior-specific experts.

**4.3.2 Generating of Behavior-fitting Experts.** To address the challenge of mitigating feature distribution bias, we also define a dedicated behavior-fitting expert for each task, whose outputs are used in the subsequent aggregation process. Specifically, for the current

behavior  $k$  and any other behavior  $k'$ , it can be formulated as:

$$\begin{cases} \mathbf{e}_{u,in}^{k,k'} = (\alpha \mathbf{e}_u^{k,*} + \beta \mathbf{e}_u^{k',*})/2 \\ \mathbf{e}_{v,in}^{k,k'} = (\alpha \mathbf{e}_v^{k,*} + \beta \mathbf{e}_v^{k',*})/2 \\ \mathbf{e}_{in}^{k,k'} = \mathbf{e}_{u,in}^{k,k'} \parallel \mathbf{e}_{v,in}^{k,k'} \\ \mathbf{e}_{out}^{k,k'} = \text{Agg}(\mathbf{e}_{in}^{k,k'}, \mathbf{A}_{k'}) \end{cases} \quad (10)$$

where  $\mathbf{e}_{out}^{k,k'} = \mathbf{e}_{u,out}^{k,k'} \parallel \mathbf{e}_{v,out}^{k,k'}$ , ( $\parallel$ ) is the concatenate operation,  $\mathbf{e}_u^{k,*}$  and  $\mathbf{e}_v^{k,*}$  are the user representations of the  $k$ - and  $k'$ -th behavior ( $k' \neq k$ ) respectively, similarly for items.  $\alpha, \beta$  are coefficients that control the scaling of behavioral representations, and  $\text{Agg}(\cdot)$  is a graph convolution operator. In particular, the values of  $\alpha$  and  $\beta$  should be small to achieve the effect of fine-tuning the representation space. We use a graph convolutional network with the  $k'$ -th behavior interaction matrix (i.e.,  $\text{Agg}(\cdot)$ ) to capture the effective information contained within the representation  $\mathbf{e}_{in}^{k,k'}$ . Similar to Equation 5, the graph convolutional operator is defined as follows:

$$\mathbf{E}^{k,k',l+1} = (\mathbf{D}^{-1} \mathbf{A}_{k'} + \mathbf{I})(\mathbf{E}^{k,k',l} \circ \mathbf{R}^{k',l}), \mathbf{R}^{k',l} = \mathbf{W}^l \mathbf{R}^{k',l-1} \quad (11)$$

where ( $\circ$ ) is the hadamard product operation,  $\mathbf{A}_{k'}$  is the adjacency matrix of behavior  $k'$ .  $\mathbf{W}^l$  is the layer specific parameter shared with the layer of GCN. We use a hierarchically updated behavior embedding matrix  $\mathbf{R}^{k',l}$  for fully learning of the  $k'$ -th behavior information, with its initial state is  $\mathbf{R}^{k',1}$ .

As  $\mathbf{e}_{out}^{k,k'}$  contains the representation of users and items, we utilize the hadamard product operation to generate final experts, which is similar to behavior-specific experts:

$$\mathbf{e}^{k,k'} = \mathbf{e}_{u,out}^{k,k'} \circ \mathbf{e}_{v,out}^{k,k'} \quad (12)$$

**4.3.3 Aggregating of Experts.** In order to mitigate the problem of negative transfer caused by distribution differences between features and labels, we improve the task aggregation mechanism in both forward and backward propagation. As we can see, behavior-aware graph convolution operation and representation scaling mechanism help us capture the effective components of other behaviors, which can be further utilized to alleviate the negative transfer caused by feature distribution differences in gated aggregation. To be specific, We define the gate for task  $k$  as:

$$\mathbf{g}^k = \text{Softmax}(\mathbf{W}_g(\mathbf{e}_u^{k,*} \parallel \mathbf{e}_v^{k,*}) + \mathbf{b}_g) \quad (13)$$

where ( $\parallel$ ) is the concatenate operation,  $\mathbf{W}_g \in \mathbb{R}^{K \times 2d}$  and  $\mathbf{b}_g \in \mathbb{R}^{K \times 1}$  are feature transformation matrix and bias matrix, and  $\mathbf{g}^k \in \mathbb{R}^{K \times 1}$  is the attention vector which are used as selector to calculate the weighted sum of all experts. We then take the refined representations  $\mathbf{e}^{j,k}$  ( $j \in \{1, 2, \dots, K\} \cap j \neq k$ ) of other behaviors and  $\mathbf{e}^k$  as targets of aggregation by the  $k$ -th gate. In order to eliminate the negative impact caused by the distribution differences in labels between the auxiliary and target behaviours, we ensure that the parameters of the target task are not updated by the gradient updates from the auxiliary tasks, thus preventing interference from auxiliary behaviors. Formally, we have:

$$\mathbf{o}^k(j) = \begin{cases} \mathbf{g}^k(j) \cdot \mathbf{e}^k, & j = k \\ \mathbf{g}^k(j) \cdot \mathbf{e}^{j,k}, & j \neq k \\ \text{sg}(\mathbf{g}^k(j) \cdot \mathbf{e}^{j,k}), & j \neq k \text{ and } j = K \end{cases} \quad (14)$$

**Table 1: Statistics of evaluation datasets.**

Dataset	#User	#Item	#Interaction	#Target Interaction	#Interactive Behavior Type
Beibei	21,716	7,977	$3.3 \times 10^6$	282,860	{View, Cart, Buy}
Taobao	15,449	11,953	$1.2 \times 10^6$	92,180	{View, Cart, Buy}
Tmall	41,738	11,953	$2.3 \times 10^6$	255,586	{View, Collect, Cart, Buy}

where  $\mathbf{g}^k(j)$  denotes the  $j$ -th element of vector  $\mathbf{g}^k$ .  $\text{sg}(\cdot)$  is the stop gradient operation. The final prediction for task  $k$  is calculated as:

$$\hat{o}_{uv}^k = h^k\left(\sum_{j=1}^K \mathbf{o}^k(j)\right) \quad (15)$$

where  $h^k(\cdot)$  is the tower function. For simplicity, we use average operation as the tower function here.  $\hat{o}_{uv}^k$  is the prediction score of whether user  $u$  will have interaction with item  $v$  under behavior  $k$ .

## 4.4 Joint Optimization

As we have obtained the final prediction  $\hat{o}_{uv}^k$  for each behavior  $k$ , we leverage the Bayesian Personalized Ranking (BPR)[35] loss to optimize the model:

$$\mathcal{L}_{bpr} = - \sum_{k=1}^K \sum_{(u,s,t) \in \mathcal{O}_k} \lambda_k * \ln \sigma(\hat{o}_{us}^k - \hat{o}_{ut}^k) \quad (16)$$

where  $\mathcal{O}_k = \{(u, s, t) | (u, s) \in \mathcal{O}_k^+, (u, t) \in \mathcal{O}_k^-\}$  denotes the training dataset.  $\mathcal{O}_k^+$  and  $\mathcal{O}_k^-$  indicates the observed and unobserved user-item interactions under behavior  $k$ , respectively.  $\lambda_k$  is the coefficient of behavior  $k$ , and  $\sigma$  is the sigmoid function.

In all, the final loss can be formulated as:

$$\mathcal{L}(\Theta) = \mathcal{L}_{bpr} + \gamma \sum_{k=1}^{K-1} \mathcal{L}_{cl}^{K,k} + \mu \|\Theta\|_2^2 \quad (17)$$

where  $\gamma$  is the coefficient of cl loss,  $\Theta$  represents set of all model parameters.  $\mu$  is the  $L_2$  regularization coefficient for  $\Theta$ .

## 5 EXPERIMENTS

### 5.1 Experimental Setting

**5.1.1 Parameter Settings.** Our proposed COPF is implemented in TensorFlow [1]. For a fair comparison, following PKEF [31] and BCIPM [52], we set the embedding size to 64. We initialize the parameters using Xavier [11]. The parameters are optimized by Adam [22], while the learning rate is set to  $10^{-3}$ . We search the number of GCN layers in  $\{1, 2, 3, 4\}$ . Moreover, we adjust the loss coefficients for each behavior in  $\{0, 1/6, 2/6, 3/6, 4/6, 5/6, 1\}$  and fix the sum of the coefficients for all actions as 1. The coefficient of contrastive loss  $\gamma$  and  $L_2$  regularization  $\mu$  are set to 1 and 0.01, respectively. All experiments are run 5 times and average results are reported. For fairness, the parameter settings of the baseline are adjusted and searched by referring to the original work. Furthermore, we conduct parameter analysis experiments, which are shown in Section 5.5.

**5.1.2 Dataset Description.** We use three public datasets (Beibei, Taobao and Tmall) to validate the effectiveness of our proposed COPF model. The pre-processing of these datasets is consistent with the previous methods [15, 31]. Specifically, we eliminate duplicate user-item interactions through retaining the earliest one.

**Table 2: The overall performance comparison. Boldface denotes the highest score and underline indicates the results of the best baselines.  $\star$  represents significance level  $p$ -value  $< 0.05$  of comparing COPF with the best baseline.**

Model	Beibei		Taobao		Tmall	
	HR	NDCG	HR	NDCG	HR	NDCG
MF-BPR	0.0191	0.0049	0.0076	0.0036	0.0230	0.0207
NeuMF	0.0232	0.0135	0.0236	0.0128	0.0124	0.0062
LightGCN	0.0391	0.0209	0.0411	0.0240	0.0393	0.0209
RGCN	0.0363	0.0188	0.0215	0.0104	0.0316	0.0157
GNMR	0.0413	0.0221	0.0368	0.0216	0.0393	0.0193
NMTR	0.0429	0.0198	0.0282	0.0137	0.0536	0.0286
MBGCN	0.0470	0.0259	0.0509	0.0294	0.0549	0.0285
S-MBRec	0.0489	0.0253	0.0498	0.0269	0.0694	0.0362
KMCLR	0.0531	0.0263	0.1185	0.0659	0.0603	0.0310
MB-CGCN	0.0579	0.0381	0.1233	0.0677	0.0984	0.0558
CML	0.0542	0.0268	0.1203	0.0661	0.0448	0.0227
CRGCN	0.0459	0.0324	0.0855	0.0439	0.0840	0.0442
CIGF	0.0809	0.0400	0.0897	0.0474	0.1150	0.0636
PKEF	<u>0.1130</u>	<u>0.0582</u>	<u>0.1385</u>	<u>0.0785</u>	0.1277	0.0721
BCIPM	0.0458	0.0221	0.1201	0.0656	0.1414	0.0741
<b>COPF</b>	<b>0.1694<math>\star</math></b>	<b>0.0903<math>\star</math></b>	<b>0.1552<math>\star</math></b>	<b>0.0838<math>\star</math></b>	<b>0.1755<math>\star</math></b>	<b>0.0967<math>\star</math></b>
Rel Impr.	49.91%	55.15%	12.06%	6.75%	24.12%	30.50%

The statistical information of these three datasets is summarized in Table 1.

**5.1.3 Evaluation Metrics.** To evaluate the performance of COPF and baseline methods in top-k item recommendation, we use two metrics: Hit Ratio ( $HR@K$ ) and Normalized Discounted Cumulative Gain ( $NDCG@K$ ). In all our experiments, we set  $K = 10$ .

**5.1.4 Baseline Models.** To validate the effectiveness of COPF, we compared it with numerous baseline models in recent years, which can be divided into three categories: **(1) Single-behavior methods:** MF-BPR [35], NeuMF [17] and LightGCN [16], **(2) Multi-behavior methods without MTL:** RGCN [36], GNMR [46], NMTR [10], MBGCN<sup>1</sup> [21], S-MBRec [12], KMCLR [50] and MB-CGCN<sup>2</sup> [6], **(3) Multi-behavior methods with MTL:** CML [45], CRGCN [51], CIGF [15], PKEF<sup>3</sup> [31] and BCIPM<sup>4</sup> [52].

## 5.2 Performance Comparison

Table 2 shows the performance of methods on three datasets with respect to  $HR@10$  and  $NDCG@10$ . We have the following findings:

- Our proposed COPF model achieves the best performance on all three datasets. Specifically, COPF improves the best baselines by **49.91%**, **12.06%**, and **24.12%** in terms of HR ( **55.15%**, **6.75%**, and **30.50%** in terms of NDCG) on Beibei, Taobao, and Tmall datasets, respectively. Due to the varying user behavior patterns across different datasets, the superior performance on all datasets further demonstrates the applicability and effectiveness of COPF for multi-behavior recommendation.
- In single-behavior methods, LightGCN achieves better performance than MF-BPR and NeuMF, while in multi-behavior methods, MBGCN also outperforms NMTR. This demonstrates the

<sup>1</sup><https://github.com/tsinghua-fib-lab/MBGCN>

<sup>2</sup><https://github.com/SS-00-SS/MBGCN>

<sup>3</sup><https://github.com/MC-CV/PKEF>

<sup>4</sup><https://github.com/MingshiYan/BIPN>

**Table 3: Performances of different COPF variants.**

Model	Beibei		Taobao		Tmall	
	HR	NDCG	HR	NDCG	HR	NDCG
w/o COGCN	0.0601	0.0294	0.0801	0.0417	0.0783	0.0437
COPF-P	0.1282	0.0661	0.1041	0.0540	0.1201	0.0639
COPF-A	0.1649	0.0885	0.1389	0.0740	0.1592	0.0883
COPF-D	0.0333	0.0163	0.1052	0.0595	0.1064	0.0606
COPF-F	0.1285	0.0665	0.0996	0.0515	0.1167	0.0620
COPF-C	0.1514	0.0861	0.0913	0.0511	0.0785	0.0446
COPF-B	0.1622	0.0870	0.1531	0.0817	0.1714	0.0957
COPF-H	0.1199	0.0649	0.0841	0.0478	0.0809	0.0475
w/o DFME	0.0821	0.0400	0.1120	0.0607	0.1146	0.0643
w/o con.	0.1154	0.0581	0.1476	0.0788	0.1425	0.0791
w/o for.	0.1649	0.0886	0.0471	0.0247	0.1391	0.0769
w/o back.	0.1676	0.0886	0.1446	0.0781	0.1601	0.0888
all sg.	0.1656	0.0888	0.1508	0.0809	0.1553	0.0866
w/o fit.	0.1182	0.0614	0.0827	0.0438	0.1346	0.0738
<b>COPF</b>	<b>0.1694</b>	<b>0.0903</b>	<b>0.1552</b>	<b>0.0838</b>	<b>0.1755</b>	<b>0.0967</b>

advantage of GCNs in capturing high-order interactive information. Furthermore, most of the multi-behavior recommendation methods, such as MBGCN, perform better than single-behavior methods on all three datasets, which highlights the superiority of leveraging multi-behavior information for learning. Finally, the excellent performance of CML and KMCLR also illustrates the effectiveness of contrastive learning.

- Although models vary in network structure, the multi-behavior methods with MTL generally perform better overall compared to those without MTL. For example, PKEF consistently outperforms all the multi-behavior methods without MTL. It is worth noting that KMCLR and MB-CGCN perform better among the multi-behavior methods without MTL. The possible reasons are that KMCLR enhances the original multi-behavioral information by introducing external knowledge graph information; Meanwhile, MB-CGCN reduces the solution space of the multi-behavior fusion problem through cascade constraints. Even though this constraint is overly strict, it still achieves relatively better results in the biased space.
- MBGCN outperforms RGCN by considering the contribution of each behavior during behavior fusion. Compared to them, CIGF utilizes multi-task learning in the prediction process, which further improves the performance. However, they still lacked proper constraints or imposed overly relaxed constraints on user behavior patterns. Recent approaches like CRGCN, MB-CGCN, and PKEF use cascading paradigm to constrain the learning of user behavior patterns; BCIPM further relaxes the constraints within the cascading paradigm and highlights the significance of the target behavior. As we can see, PKEF achieves second performance only to our model on Beibei and Taobao, while BCIPM does the same on Tmall. This indicates the necessity of considering user behavior pattern constraints from a combinatorial optimization perspective.

## 5.3 Ablation Study

**5.3.1 Impact of the Key Components.** To evaluate the effectiveness of sub-modules in our COPF framework, we conducted ablation experiments on COGCN and DFME respectively. For COGCN, we mainly consider the constraints of each stage. Specifically, we

**Table 4: Performances of different aggregation schemes.**

Model	Beibei		Taobao		Tmall	
	HR	NDCG	HR	NDCG	HR	NDCG
No Aggregation	0.1103	0.0560	0.1143	0.0634	0.1165	0.0655
Summation	0.1163	0.0605	0.0789	0.0410	0.1323	0.0732
Linear Trans.	0.0724	0.0342	0.1317	0.0712	0.1408	0.0771
Vanilla Fusion	0.1228	0.0637	0.1080	0.0589	0.1337	0.0749
Projection Fusion	0.1299	0.0672	0.1153	0.0611	0.1442	0.0791
<b>COPF</b>	<b>0.1694</b>	<b>0.0903</b>	<b>0.1552</b>	<b>0.0838</b>	<b>0.1755</b>	<b>0.0967</b>

first define two strict constraints: (1) *strict pre-behavior constraint*: Change the pre-behavior constraint to "only receive information about the most recent upstream behavior"; (2) *strict in-behavior constraint*: Change the in-behavior constraint to "only learn the current behavior signal". Then we design the following variants: (1) **w/o COGCN**: Replace COGCN with multiple LightGCN.(2) **COPF-P**: Remove pre-behavior constraint.(3) **COPF-A**: Remove in-behavior constraint.(4) **COPF-D**: Remove post-behavior constraint(and no DFME either).(5) **COPF-F**: Remove pre-behavior and in-behavior constraints.(6) **COPF-C**:Use a strict cascading paradigm.(7) **COPF-B**: Use strict pre-behavior constraint instead. (8) **COPF-H**: Use strict in-behavior constraint instead. For DFME, we have: (1) **w/o DFME**: Replace DFME with the bilinear module.(2) **w/o con.**: Remove contrastive learning.(3) **w/o for.**: Remove improvement in forward propagation during aggregation.(4) **w/o back.**: Remove improvement in backward propagation during aggregation.(5) **all sg.**: Use simple stop-gradient strategy in backward propagation during aggregation.(6) **w/o fit.**: Remove improvements in both propagation during aggregation. The results in Table 3 lead to the following conclusions:

- Comparing the performance of COPF with other variants that modify constraints in COGCN (for fairness, **COPF-D** is compared with **w/o DFME**), we observe that removing or altering the constraints of any stage leads to varying degrees of performance degradation. Additionally, **w/o COGCN** achieves the worst performance on the three datasets compared to other variants. These demonstrate the effectiveness of addressing multi-behavior fusion from a combinatorial optimization perspective and validate the rationality of the constraints established at each stage of COGCN.
- The performance of each variant modified for DFME is also affected to varying degrees, with the **w/o DFME** variant performing the worst. This demonstrates the rationality and effectiveness of our proposed DFME.

**5.3.2 Impact of the Aggregation Schemes.** To further explore the optimal approach for coordinating tasks, we compare the proposed scheme with several other alternatives: (1) **No Aggregation**: No aggregation process between tasks. (2) **Summation**: Simply add the representation of different tasks. (3) **Linear Trans.**: Apply a linear transformation to transfer the task representation. (4) **Vanilla Fusion** [31]: Utilize a variant of vanilla attention[56]. (5) **Projection Fusion** [2]: Explicitly extract the information through projection mechanism. It is worth noting that in order to directly compare the performance of the aggregation methods, all schemes have incorporated the contrastive learning between behaviors. As shown in Table 4, We can observe that No Aggregation performs the worst

**Table 5: Performances of different MTL module.**

Model	Beibei		Taobao		Tmall	
	HR	NDCG	HR	NDCG	HR	NDCG
COGCN+SB	0.0328	0.0159	0.0594	0.0305	0.0740	0.0403
COGCN+MMOE	0.0557	0.0280	0.0610	0.0317	0.0838	0.0440
COGCN+PLE	0.0546	0.0269	0.0649	0.0338	0.0801	0.0422
COGCN+Bilinear	0.0629	0.0306	0.0986	0.0522	0.1559	0.0858
COGCN+MESI	0.0885	0.0439	0.1039	0.0540	0.1487	0.0811
COGCN+PME	0.1137	0.0572	0.1525	0.0800	0.1572	0.0862
<b>COGCN+DFME</b>	<b>0.1694</b>	<b>0.0903</b>	<b>0.1552</b>	<b>0.0838</b>	<b>0.1755</b>	<b>0.0967</b>

overall among all the schemes, which fully demonstrates the importance of aggregation between tasks. Summation negatively impacts the distribution of representations, leading to relatively poor performance on the three datasets. Additionally, both Linear Trans. and Vanilla Fusion methods ignore the noise that may be introduced during aggregation, which can result in negative information transfer. Projection Fusion utilizes a projection mechanism, avoiding the introduction of harmful information while also mitigating the impact of distribution differences in representations. Finally, our proposed method demonstrates the best performance across all three datasets, highlighting the effectiveness of our scheme.

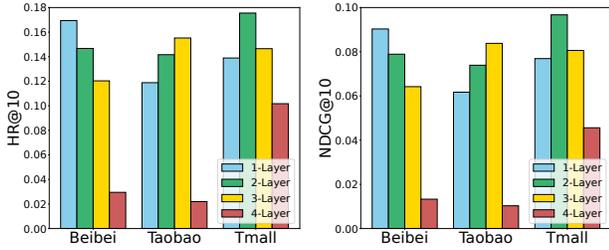
**5.3.3 Impact of the MTL module.** To further demonstrate the superiority of our proposed DFME in MTL, we compare it with some other MTL models: Shared Bottom [3], Bilinear [4], MMOE [29], PLE [40], MESI [15] and PME [31]. These MTL models are applied on top of COGCN for multi-behavior recommendation, which are named COGCN+SB, COGCN+MMOE, COGCN+PLE, COGCN+Bilinear, COGCN+MESI, and COGCN+PME respectively. In particular, We weight the  $K$  separate representations generated by COGCN to meet the shared input requirement of the classical MTL models (i.e., Shared Bottom, MMOE, and PLE). The experimental results are presented in Table 5. COGCN+SB achieves the poorest performance among all MTL models on all datasets. COGCN+MMOE and COGCN+PLE outperform COGCN+SB under the same conditions through gating mechanism, which demonstrates the necessity of task aggregation. COGCN+Bilinear shows better performance by using decoupled input and streamlined task prediction tower functions. COGCN+MESI further combines both decoupled inputs and task aggregation but performs worse than COGCN+Bilinear on Tmall dataset, which is possibly due to differences in feature distributions during aggregation. COGCN+PME further enforces alignment of task representation spaces during aggregation, significantly enhancing performance. Finally, our DFME consistently outperforms all other models on all datasets, verifying its effectiveness for MTL.

## 5.4 Compatibility Analysis

Our proposed DFME can serve as a general module applicable to most existing multi-behavior methods, and we validate this through a compatibility analysis. Specifically, we select some representative multi-behavior methods with MTL, like CRGCN, CIGF and PKEF, as well as multi-behavior methods without MTL, like LightGCN<sub>M</sub>(LightGCN enhanced with the multi-behavioral graph) and MB-CGCN. Then, we replace their prediction modules with DFME, and compare them with the corresponding original models. The results are shown in Table 6. As we can see, our proposed

**Table 6: Compatibility performance of DFME with different models as backbones ("X+DFME" means using X to replace the COGCN in COPF).**

Model	Beibei		Taobao		Tmall	
	HR	NDCG	HR	NDCG	HR	NDCG
LightGCN <sub>M</sub>	0.0456	0.0224	0.0452	0.0246	0.0489	0.0297
LightGCN <sub>M</sub> +DFME	<b>0.0928</b>	<b>0.0461</b>	<b>0.0928</b>	<b>0.0522</b>	<b>0.0865</b>	<b>0.0493</b>
MB-CGCN	0.0579	0.0381	0.1233	0.0677	0.0984	0.0558
MB-CGCN+DFME	<b>0.1356</b>	<b>0.0747</b>	<b>0.1400</b>	<b>0.0764</b>	<b>0.1094</b>	<b>0.0585</b>
CRGCN	0.0459	0.0324	0.0855	0.0439	0.0840	0.0442
CRGCN+DFME	<b>0.1253</b>	<b>0.0663</b>	<b>0.1322</b>	<b>0.0698</b>	<b>0.1308</b>	<b>0.0736</b>
CIGF	0.0809	0.0400	0.0897	0.0474	0.1150	0.0636
CIGF+DFME	<b>0.0851</b>	<b>0.0435</b>	<b>0.1097</b>	<b>0.0600</b>	<b>0.1219</b>	<b>0.0704</b>
PKEF	0.1130	0.0582	0.1385	0.0785	0.1277	0.0721
PKEF+DFME	<b>0.1320</b>	<b>0.0701</b>	<b>0.1425</b>	<b>0.0794</b>	<b>0.1419</b>	<b>0.0810</b>

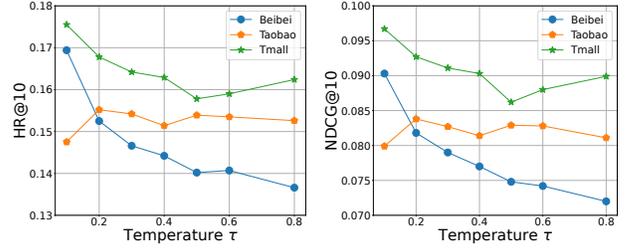
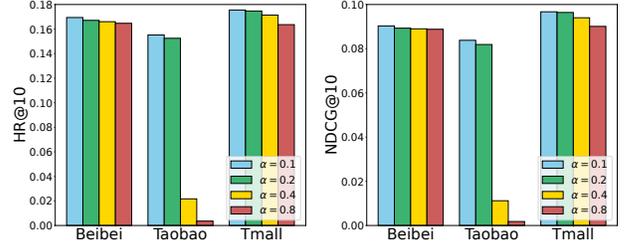
**Figure 3: Impact of GCN layers.**

DFME improves the performance of all original models. The original LightGCN<sub>M</sub> and MB-CGCN benefit significantly from DFME due to their lack of MTL modules. Among multi-behavior methods with MTL, CRGCN exhibits more significant improvement. This is likely due to its original MTL module being relatively basic, which allows for greater compatibility. In contrast, CIGF and PKEF already have sizable MTL modules, resulting in a less pronounced improvement. Overall, the results fully demonstrate the wide compatibility and general applicability of our proposed DFME, which can be integrated into the multi-behavior methods to improve their performance.

## 5.5 Parameter Analysis

**5.5.1 Impact of the number of layers.** We investigate the impact of high-order interaction information on model performance by varying the number of GCN layers within the range of {1, 2, 3, 4}. As shown in Figure 3, the optimal number of layers varies by dataset, which is determined by the relative amounts of noise and useful signals in the high-order information. However, when the number of layers exceeds three, performance significantly declines due to the over-smoothing problem of GCN.

**5.5.2 Impact of temperature hyperparameter.** We adjust the temperature hyperparameter in contrastive learning within the range of {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8} and plot the resulting curves (shown in Figure 4). For all three datasets, excessively large temperature coefficients lead to poorer performance, which indicates that a large temperature value will reduce the ability to distinguish negative samples. Additionally, in the Taobao dataset, performance also declines when the temperature coefficient is too small (e.g., 0.1). This may be due to the imbalanced contribution of the samples.

**Figure 4: Impact of temperature hyperparameter.****Figure 5: Impact of the scaling size.**

**5.5.3 Impact of the scaling size.** With  $\beta$  fixed at 0.001, we investigate the impact of the scaling factor  $\alpha$  on the forward propagation in multi-task learning. Figure 5 shows the results of our search in the range of {0.1, 0.2, 0.4, 0.8}. Across all datasets, COPF achieves the highest performance when the scaling factor is small (i.e., 0.1). Besides, the performance gradually decreases as the scaling factor increases, which is particularly evident in the Taobao dataset. These observations highlight the effectiveness of "fine-tuning" the representation space.

## 6 CONCLUSION

In this paper, we propose the Combinatorial Optimization Perspective based Framework (COPF) for multi-behavior recommendation. To avoid incorrect modeling of user behavior patterns due to limited perspectives, we propose the Combinatorial Optimization Graph Convolutional Network (COGCN), which treats multi-behavior fusion as a combinatorial optimization problem. COGCN imposes different constraints at various stages of each behavior to restrict the solution space, thereby effectively facilitating the multi-behavior fusion process. To better coordinate the correlations between tasks in MTL methods, we design the Distribution Fitting Expert Network (DFME), which improves both forward and backward propagation. By reducing the distribution differences in features and labels between tasks, DFME alleviates negative information transfer caused by uncoordinated task relationships. We conduct comprehensive experiments on three real-world datasets to verify the superiority of our proposed COPF. Further analysis demonstrates the rationality and effectiveness of the designed COGCN and DFME modules.

## Acknowledgments

This work was supported by the STI 2030-Major Projects under Grant 2021ZD0201404.

## References

- [1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*. 265–283.
- [2] 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 *Fifteenth ACM Conference on Recommender Systems*. 400–410.
- [3] Rich Caruana. 1997. Multitask learning. *Machine learning* 28, 1 (1997), 41–75.
- [4] Chong Chen, Weizhi Ma, Min Zhang, Zhaowei Wang, Xiuqiang He, Chenyang Wang, Yiqun Liu, and Shaoping Ma. 2021. Graph Heterogeneous Multi-Relational Recommendation. In *AAAI*, Vol. 35. 3958–3966.
- [5] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting Graph Based Collaborative Filtering: A Linear Residual Graph Convolutional Network Approach. In *AAAI*, Vol. 34. 27–34.
- [6] 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*. 1181–1189.
- [7] Yizhou Dang, Enneng Yang, Guibing Guo, Linying Jiang, Xingwei Wang, Xiaoxiao Xu, Qinghui Sun, and Hong Liu. 2023. Uniform sequence better: Time interval aware data augmentation for sequential recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 4225–4232.
- [8] Yizhou Dang, Jiahui Zhang, Yuting Liu, Enneng Yang, Yuliang Liang, Guibing Guo, Jianzhe Zhao, and Xingwei Wang. 2024. Augmenting Sequential Recommendation with Balanced Relevance and Diversity. *arXiv preprint arXiv:2412.08300* (2024).
- [9] Jingtao Ding, Guanghui Yu, Xiangnan He, Yuhuan Quan, Yong Li, Tat-Seng Chua, Depeng Jin, and Jiajie Yu. 2018. Improving Implicit Recommender Systems with View Data. In *IJCAI*. 3343–3349.
- [10] 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.
- [11] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *ALSTATS*. 249–256.
- [12] Shuyun Gu, Xiao Wang, Chuan Shi, and Ding Xiao. 2022. Self-supervised Graph Neural Networks for Multi-behavior Recommendation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. 2052–2058.
- [13] Guibing Guo, Huihui Qiu, Zhenhua Tan, Yuan Liu, Jing Ma, and Xingwei Wang. 2017. Resolving data sparsity by multi-type auxiliary implicit feedback for recommender systems. *Knowledge-Based Systems* 138 (2017), 202–207.
- [14] 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 *SIGKDD*. 1984–1992.
- [15] Wei Guo, Chang Meng, Enming Yuan, Zhicheng He, Hui Feng Guo, Yingxue Zhang, Bo Chen, Yaochen Hu, Ruiming Tang, Xiu Li, et al. 2023. Compressed Interaction Graph based Framework for Multi-behavior Recommendation. In *Proceedings of the ACM Web Conference 2023*. 960–970.
- [16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *arXiv preprint arXiv:2002.02126* (2020).
- [17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
- [18] Zhicheng He, Weiwen Liu, Wei Guo, Jiarui Qin, Yingxue Zhang, Yaochen Hu, and Ruiming Tang. 2023. A Survey on User Behavior Modeling in Recommender Systems. *arXiv preprint arXiv:2302.11087* (2023).
- [19] Chao Huang. 2021. Recent Advances in Heterogeneous Relation Learning for Recommendation. *arXiv preprint arXiv:2110.03455* (2021).
- [20] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.
- [21] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-behavior recommendation with graph convolutional networks. In *SIGIR*.
- [22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [23] 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 ACM international conference on Web search and data mining*. 173–182.
- [24] Pengcheng Li, Runze Li, Qing Da, An-Xiang Zeng, and Lijun Zhang. 2020. Improving multi-scenario learning to rank in e-commerce by exploiting task relationships in the label space. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2605–2612.
- [25] Xiang Li, Chaofan Fu, Zhongying Zhao, Guanjie Zheng, Chao Huang, Junyu Dong, and Yanwei Yu. 2024. Dual-Channel Multiplex Graph Neural Networks for Recommendation. *arXiv preprint arXiv:2403.11624* (2024).
- [26] Xiaodong Li, Jiawei Sheng, Jiangxia Cao, Wenyuan Zhang, Quangang Li, and Tingwen Liu. 2024. CDRNP: Cross-Domain Recommendation to Cold-Start Users via Neural Process. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 378–386.
- [27] Ke Liang, Yue Liu, Sihang Zhou, Wenxuan Tu, Yi Wen, Xihong Yang, Xiangjun Dong, and Xinwang Liu. 2023. Knowledge Graph Contrastive Learning Based on Relation-Symmetrical Structure. *IEEE Transactions on Knowledge and Data Engineering* (2023).
- [28] Babak Loni, Roberto Pagano, Martha Larson, and Alan Hanjalic. 2016. Bayesian personalized ranking with multi-channel user feedback. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 361–364.
- [29] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *SIGKDD*. 1930–1939.
- [30] Chang Meng, Chenhao Zhai, Xueliang Wang, Shuchang Liu, Xiaoqiang Feng, Lantao Hu, Xiu Li, Han Li, and Kun Gai. 2024. Coarse-to-fine Dynamic Uplift Modeling for Real-time Video Recommendation. *arXiv preprint arXiv:2410.16755* (2024).
- [31] 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*. 1797–1806.
- [32] Chang Meng, Hengyu Zhang, Wei Guo, Hui Feng 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*. 4649–4660.
- [33] Chang Meng, Ziqi Zhao, Wei Guo, Yingxue Zhang, Haolun Wu, Chen Gao, Dong Li, Xiu Li, and Ruiming Tang. 2022. Coarse-to-Fine Knowledge-Enhanced Multi-Interest Learning Framework for Multi-Behavior Recommendation. *arXiv preprint arXiv:2208.01849* (2022).
- [34] Huihui Qiu, Yun Liu, Guibing Guo, Zhu Sun, Jie Zhang, and Hai Thanh Nguyen. 2018. BPRH: Bayesian personalized ranking for heterogeneous implicit feedback. *Information Sciences* 453 (2018), 80–98.
- [35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
- [36] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.
- [37] Ajit P 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*. 650–658.
- [38] Liangcai Su, Junwei Pan, Ximei Wang, Xi Xiao, Shijie Quan, Xihua Chen, and Jie Jiang. 2024. STEM: Unleashing the Power of Embeddings for Multi-task Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 9002–9010.
- [39] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).
- [40] Hongyan Tang, Junming Liu, Ming Zhao, and Xudong Gong. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *RecSys*. 269–278.
- [41] 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*. 283–292.
- [42] Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 242–264.
- [43] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*. 165–174.
- [44] Xiaobei Wang, Shuchang Liu, Xueliang Wang, Qingpeng Cai, Lantao Hu, Han Li, Peng Jiang, Kun Gai, and Guangming Xie. 2024. Future Impact Decomposition in Request-level Recommendations. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5905–5916.
- [45] Wei Wei, Chao Huang, Lianghao Xia, Yong Xu, Jiashu Zhao, and Dawei Yin. 2022. Contrastive meta learning with behavior multiplicity for recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1120–1128.
- [46] 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 *ICDE*. IEEE, 1931–1936.
- [47] 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.
- [48] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4486–4493.
- [49] Lianghao Xia, Yong Xu, Chao Huang, Peng Dai, and Liefeng Bo. 2021. Graph meta network for multi-behavior recommendation. In *SIGIR*. 757–766.

- [50] Hongrui Xuan, Yi Liu, Bohan Li, and Hongzhi Yin. 2023. Knowledge Enhancement for Contrastive Multi-Behavior Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 195–203.
- [51] Mingshi Yan, Zhiyong Cheng, Chen Gao, Jing Sun, Fan Liu, Fuming Sun, and Haojie Li. 2022. Cascading Residual Graph Convolutional Network for Multi-Behavior Recommendation. *arXiv preprint arXiv:2205.13128* (2022).
- [52] Mingshi Yan, Fan Liu, Jing Sun, Fuming Sun, Zhiyong Cheng, and Yahong Han. 2024. Behavior-Contextualized Item Preference Modeling for Multi-Behavior Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 946–955.
- [53] Chengqing Yu, Guangxi Yan, Chengming Yu, Yu Zhang, and Xiwei Mi. 2023. A multi-factor driven spatiotemporal wind power prediction model based on ensemble deep graph attention reinforcement learning networks. *Energy* 263 (2023), 126034.
- [54] Xuhao Zhao, Yanmin Zhu, Chunyang Wang, Mengyuan Jing, Jiadi Yu, and Feilong Tang. 2023. Task-difficulty-aware meta-learning with adaptive update strategies for user cold-start recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 3484–3493.
- [55] Zhe Zhao, Zhiyuan Cheng, Lichan Hong, and Ed H Chi. 2015. Improving user topic interest profiles by behavior factorization. In *Proceedings of the 24th International Conference on World Wide Web*. 1406–1416.
- [56] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiushi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

## A APPENDIX

### A.1 Complexity Analysis

**A.1.1 Time Complexity.** The main time consumption of COPF lies in the graph convolution operations. The computational complexity of the both steps is  $O(L|\mathcal{E}|d)$ , where  $|\mathcal{E}|$  represents the number of edges in the bipartite graph  $\mathcal{G}$ ,  $L$  denotes the number of GNN layers, and  $d$  denotes the embedding size. Since the computational complexity does not introduce parameters outside the GNN module, the time complexity of COPF is comparable to that of existing GNN-based methods.

**A.1.2 Space Complexity.** The learnable parameters in COPF mainly come from the embeddings of users and items, resulting in a space complexity of  $O((|U| + |V|)d)$ , which is similar to existing methods. Additionally, as the dense graphs  $\mathcal{G}_k$  ( $k \in \{1, 2, \dots, K\}$ ) required for graph convolution operation are pre-converted into sparse behavior-specified matrices, there is no need to pay additional space to store these graphs in the computational process. Overall, The memory usage of the model remains within a manageable range during training.

### A.2 Analysis of Data Distribution

Figure 6 shows the data distribution on the three datasets. 1/0 indicates the presence or absence of the corresponding type of behavior. For example, 1001 represents users who only have view and buy behaviors with items. As we can see, there are significant differences in the behavior distribution between datasets, and each dataset contains rich information about user behavior patterns that can be used for learning. For instance, the user behaviors in the Beibei dataset strictly follow the dependency requirements, so the COPF-C variant performs significantly better on Beibei than the other two datasets due to its cascade paradigm (shown in Table 3). Moreover, the user behavior patterns are not explicitly available within most datasets since we do not know the specific timing of each user-item interaction. The process of learning user behavior is also implicit. This highlights the necessity of using a general framework (i.e., COGCN) to restrict the solution space by imposing

constraints from the perspective of combinatorial optimization to efficiently capture user behavior patterns.

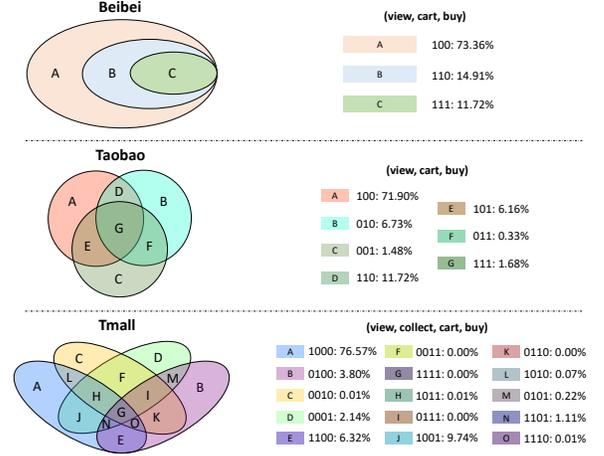


Figure 6: Data distribution of three datasets

### A.3 Analysis of the Proposed DFME

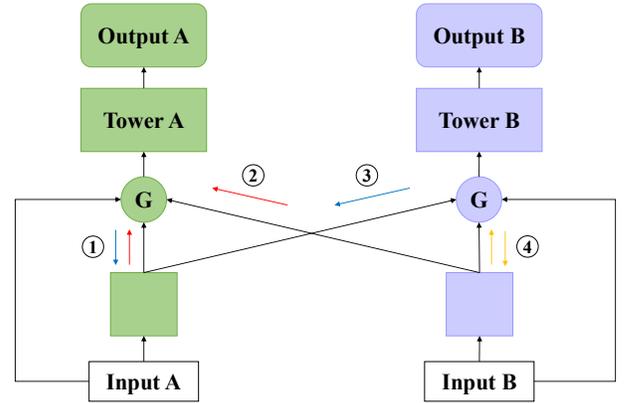


Figure 7: The structure of transfer-based MTL. Red line represents the forward propagation that affects the target task A, blue line denotes the backward propagation that affects the target task A, and yellow line represents the irrelevant process.

**A.3.1 The structure of transfer-based MTL.** A basic transfer-based MTL network is shown in Figure 7. The task corresponding to the target behavior is designated as the target task (Task A), while other tasks are auxiliary tasks (Task B and other tasks). As we can see, there are four paths associated with these two tasks. For target task A, path 1 is the main path for this task, and path 4 becomes an irrelevant path due to the decoupling of task inputs [31]. Paths 2 and 3 are information interaction paths resulting from the aggregation between tasks, which are the main correlations that DFME needs to coordinate. Specifically, path 2 affects the target task A at the feature level during forward propagation, and path 3 affects the target task A at the label level during backward propagation.

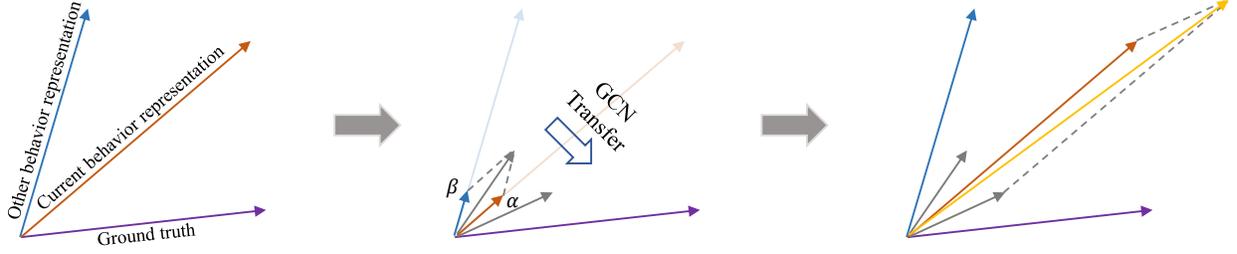


Figure 8: Refine with other behavior representations.

**A.3.2 Shortcomings of existing optimal MTL.** PKEF introduces a projection mechanism during aggregation to disentangle the shared and unique parts for other behavioral experts. The shared part is used for aggregation, while the unique part is used for auxiliary learning. While somewhat effective, this method has two main issues: **1)** For instance, with the behaviors "cart" and "buy", PKEF assumes that the shared part is the information related to both behaviors occurring together (e.g., cart & buy), while the unique part is information related to only a single behavior (e.g., only cart), thus further sets an auxiliary loss for the unique part accordingly (e.g., labels for cart w/o buy). In reality, "not buy" is also valuable information for learning behavior buy, which should exist in the shared part, and the unique part should be completely irrelevant information. This incorrect projection mechanism can compromise the effectiveness of the information during aggregation. Additionally, the projection mechanism ensures that the representation space of the auxiliary behavior remains consistent with that of the current behavior during aggregation. However, since the model is still in the training phase, the representation space for the target behavior is unstable and may be inaccurate. Enforcing such alignment could introduce detrimental information, which can potentially prevent the model from converging to the optimal distribution and adversely affect its generalization performance. **2)** PKEF overlooks the impact of gradient coupling during aggregation, where gradient updates from auxiliary tasks affect the target task. The above two problems may cause the model's inability to accurately fit the target behavior distribution, leading to negative transfer problem.

**A.3.3 Method of the proposed DFME.** Our proposed MTL network, named DFME, coordinates the relationship between target and auxiliary tasks in two key aspects to control negative information transfer. Specifically, in forward propagation, contrastive learning between tasks is utilized to enable the target behavior to obtain effective information from the auxiliary behaviors, thereby reducing the distribution gap between the target and auxiliary behaviors when generating behavior-specific experts. Meanwhile, we spatially adapt the representation of the current behavior to fit the aggregation process by generating behavior-fitting experts for each task, thereby preventing interference among different task behaviors. As shown in Figure 8, for the current behavior  $k$  and one of the other behaviors  $k'$ , we fuse the representations of behavior  $k$  and  $k'$  after multiplying them by small weight coefficients respectively. This yields a fitted representation that is appropriately sized and

slightly different in direction from the representation of behavior  $k'$ . We then obtain the final task output representation by capturing the effective information about behavior  $k'$  contained in this fitted representation through a graph convolutional network. Finally, we use behavior-specific expert for the current behavior and behavior-fitting experts for other behaviors during aggregation. In summary, the above steps can be outlined as follows: the effective information contained in the interaction between behaviors is used to refine the representation space of the current behavior while ensuring that the generalization performance of the model is not affected.

During backward propagation, for a transfer-based MTL using decoupled input  $(\mathbf{e}_u^{k'}, \mathbf{e}_v^{k'})$  for each task  $k' \in \{1, 2, \dots, K\}$ , the aggregation process of the auxiliary task  $k$  is:

$$\mathbf{o}^k = \sum_{k'=1}^K \mathbf{g}^k(k') \cdot \mathbf{e}^{k'}$$

where  $\mathbf{e}^{k'}$  is the output of the expert,  $\mathbf{g}^k(k')$  indicates the  $k'$ -th element of the gating vector  $\mathbf{g}^k$ . Due to the introduction of behavior-fitting experts (i.e.,  $f_k(\cdot)$ ) in the forward propagation to refine the representation space for each behavior, the aggregation process can be further represented as:

$$\mathbf{o}^k = \sum_{k'=1, k' \neq k}^K \mathbf{g}^k(k') \cdot \mathbf{e}^{k, k'} + \mathbf{g}^k(k) \cdot \mathbf{e}^k$$

where  $\mathbf{e}^{k, k'} = f_k(\mathbf{e}_u^k, \mathbf{e}_v^k, \mathbf{e}_u^{k'}, \mathbf{e}_v^{k'})$  denotes the output of the behavior-fitting expert. And the loss function for task  $k$  can be defined as:

$$\mathcal{L}_k = L(h^k(\mathbf{o}^k) - \hat{o}_{uv}^k)$$

where  $h^k(\cdot)$  is the tower function,  $L(\cdot)$  denotes the loss function. Then we can obtain the gradient of the auxiliary task  $k$  with respect to the behavior target behavior  $t$  as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_k}{\partial \mathbf{e}_*^t} &= \frac{\partial h^k(\mathbf{g}^k(t) \cdot \mathbf{e}^{k, t})}{\partial \mathbf{e}_*^t} * L'(h^k(\mathbf{o}^k) - \hat{o}_{uv}^k) \\ &= \frac{\partial h^k(\mathbf{g}^k(t) \cdot f_k(\mathbf{e}_u^k, \mathbf{e}_v^k, \mathbf{e}_u^t, \mathbf{e}_v^t))}{\partial \mathbf{e}_*^t} * L'(h^k(\mathbf{o}^k) - \hat{o}_{uv}^k) \end{aligned}$$

where  $\mathbf{e}_*^t \in \{\mathbf{e}_u^t, \mathbf{e}_v^t\}$ . It can be seen that the gradient update of the auxiliary task still affects the target behavior. Therefore, we stop the gradient updates from the auxiliary loss to the target behavior in order to alleviate the potential negative transfer caused by gradient coupling in the multi-behavior prediction step.