Multi-behavior Self-supervised Learning for Recommendation

Jingcao Xu
xjc20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Chaokun Wang*
chaokun@tsinghua.edu.cn
Tsinghua University
Beijing, China

Cheng Wu
c-wu19@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Yang Song
yangsong@kuaishou.com
Kuaishou Inc.
Beijing, China

Kai Zheng
zhengkai@kuaishou.com
Kuaishou Inc.
Beijing, China

Xiaowei Wang
wangxiaowei03@kuaishou.com
Kuaishou Inc.
Beijing, China

Changping Wang
wcpvincent@gmail.com
Kuaishou Inc.
Beijing, China

Guorui Zhou
zhouguorui@kuaishou.com
Kuaishou Inc.
Beijing, China

Kun Gai
gai.kun@qq.com
Unaffiliated
Beijing, China

ABSTRACT

Modern recommender systems often deal with a variety of user interactions, *e.g.*, click, forward, purchase, etc., which requires the underlying recommender engines to fully understand and leverage multi-behavior data from users. Despite recent efforts towards making use of heterogeneous data, multi-behavior recommendation still faces great challenges. Firstly, sparse target signals and noisy auxiliary interactions remain an issue. Secondly, existing methods utilizing self-supervised learning (SSL) to tackle the data sparsity neglect the serious optimization imbalance between the SSL task and the target task. Hence, we propose a Multi-Behavior Self-Supervised Learning (MBSSL) framework together with an adaptive optimization method. Specifically, we devise a behavior-aware graph neural network incorporating the self-attention mechanism to capture behavior multiplicity and dependencies. To increase the robustness to data sparsity under the target behavior and noisy interactions from auxiliary behaviors, we propose a novel self-supervised learning paradigm to conduct node self-discrimination at both inter-behavior and intra-behavior levels. In addition, we develop a customized optimization strategy through hybrid manipulation on gradients to adaptively balance the self-supervised learning task and the main supervised recommendation task. Extensive experiments on five real-world datasets demonstrate the consistent improvements obtained by MBSSL over ten state-of-the-art (SOTA) baselines. We release our model implementation at: https://github.com/Scofield666/MBSSL.git.

CCS CONCEPTS

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

*Chaokun Wang is a corresponding author.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07.
https://doi.org/10.1145/3539618.3591734

KEYWORDS

Multi-Behavior Recommendation, Collaborative filtering, Graph Neural Network, Self-Supervised Learning

ACM Reference Format:
Jingcao Xu, Chaokun Wang, Cheng Wu, Yang Song, Kai Zheng, Xiaowei Wang, Changping Wang, Guorui Zhou, and Kun Gai. 2023. Multi-behavior Self-supervised Learning for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591734

1 INTRODUCTION

Recommender systems have emerged as indispensable means of promoting personalized suggestions for users in a variety of applications, ranging from e-commerce platforms [11], online video websites [3] and location-based services [1]. Collaborative Filtering (CF) is the most extensively adopted paradigm for recommendation, which develops from traditional Matrix Factorization algorithms [15] to novel Neural Network (NN) architectures like Autoencoders [24] or Graph Neural Networks (GNNs) [13, 29].

Nevertheless, most CF solutions are designed based on a singular-type of user-item interaction behavior, whereas the real-world recommendation is more like a multi-behavior setting, *i.e.*, the interaction behaviors between users and items are multi-typed with one target behavior to be optimized [34, 35]. For example, in E-commerce services, users can interact with items in multiple manners, including page view, favorite and purchase, among which purchase behavior is the optimization target as it is directly related to Gross Merchandise Value (GMV).

Up to now, great efforts on multi-behavior recommendation have been made towards modeling the complex semantics of different interaction behaviors. MATN [34] and MBGMN [36] encode the interaction pattern multiplicity and behavior dependencies via memory-augmented Transformer and Graph Meta Network respectively. To characterize discriminative semantics of different behaviors, some research works [4, 6] learn one specific representation for each behavior. In addition, one recent study CML [31] firstly incorporates contrastive learning to tackle the label scarcity problem

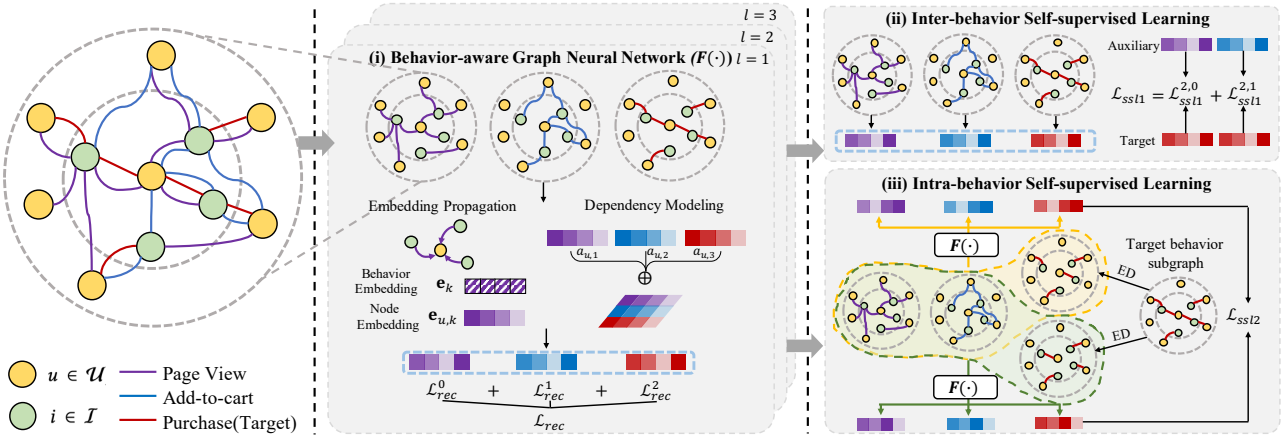


Figure 1: The overall architecture of MBSSL. i) Behavior-aware graph neural network ($F(\cdot)$) which performs behavior-specific embedding propagation and cross-behavior dependency modeling. ii) Inter-behavior SSL contrasting between target and auxiliary subgraphs to alleviate skewed data distribution and data sparsity under the target behavior. iii) Intra-behavior SSL contrasting between target and Edge Dropout (ED) subgraphs to counteract interaction noises from auxiliary behaviors.

for the target behavior. Despite their success, the multi-behavior recommendation still encounters the following challenges.

The robustness against data sparsity and interaction noises. In spite of the fact that interaction data of auxiliary behaviors could offer quite complementary information for recommendation on the target behavior, the data sparsity under the target behavior remains an issue. One possible solution is proposed in CML [31] which makes full use of supervision signals from auxiliary behaviors via conducting contrastive learning between each auxiliary behavior and target behavior pair. However, auxiliary behaviors might contain noisy interactions which are detrimental to the target task at the same time. Hence, simply adopting the contrastive learning paradigm in CML is likely to exacerbate the negative transfer towards noise distributions in auxiliary behaviors, greatly undermining the true semantics of the target behavior. In this regard, an approach to making comprehensive and adaptive use of interaction data plays a vital role in performance enhancement.

The optimization imbalance between auxiliary tasks and the target task. Existing multi-behavior recommendation solutions basically adopt the Multi-task Learning (MTL) paradigm to optimize the auxiliary and target task jointly [4, 6, 9, 18, 34]. However, ignoring the estimation of contributions of each task to the optimization target will suffer from a serious optimization imbalance problem where auxiliary tasks might dominate the network weights, resulting in worse performance on the target task. What's more, existing multi-task learning methods like GradNorm [8] or PCGrad [40] do not apply to the scenario where the self-supervised learning (SSL) task is treated as the auxiliary task, in that the SSL task has a confounding effect on the target task, depending on the particular design of SSL. Therefore, another key problem in multi-behavior recommendation is the elaborated design of the optimization method to mitigate the optimization imbalance between the auxiliary and target tasks.

Towards this end, we propose a **M**ulti-**B**ehavior **S**elf-**S**upervised **L**earning solution, short-handed as **MBSSL**, to ensure the performance for multi-behavior recommendation.

Specifically, we first devise a behavior-aware graph neural network augmented with behavior representation learning and the self-attention mechanism to jointly model the behavior inner-context and behavior inter-dependencies. For dealing with the sparse supervision signals under the target behavior, a comprehensive self-supervised learning paradigm is introduced to contrast nodes from inter-behavior and intra-behavior levels respectively. The inter-behavior SSL transfers informative semantics from auxiliary behaviors to the target behavior via selectively constructing negative node pairs. To further increase the robustness to noisy interactions, the intra-behavior SSL consolidates the self-supervised information in the target behavior to counteract the potential negative transfer brought by inter-behavior SSL. In addition, based on the observation that the SSL task exhibits an arbitrary optimization trend with respect to the target task, we design a multi-behavior optimization method that hybridly rectifies the direction and magnitude of gradients to balance SSL tasks and the target task in optimization.

In a nutshell, our main contributions are as follows:

- We develop a new self-supervised learning framework named MBSSL for multi-behavior recommendation, which embodies a behavior-aware graph neural network (Section 3.1) to uncover latent cross-behavior dependencies and a comprehensive SSL paradigm (Section 3.2) at inter-behavior and intra-behavior levels to alleviate the problem of data sparsity and interaction noises.
- To the best of our knowledge, we are the first to research the optimization imbalance problem between the SSL task and the target recommendation task. Accordingly, we propose a hybrid gradient manipulation method on both the magnitude and direction to adjust the optimization trend (Section 3.3).
- Extensive experiments are conducted on five real-world datasets

compared with ten SOTA baselines (Section 4). The consistent performance uplift on two representative metrics demonstrates the effectiveness of MBSSL.

2 PRELIMINARY

In typical recommender systems, we define the set of users and items as $\mathcal{U}$ ($u \in \mathcal{U}$) and $\mathcal{I}$ ($i \in \mathcal{I}$) respectively, where $|\mathcal{U}|$ and $|\mathcal{I}|$ represent the number of users and items. $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a bipartite graph based on the user-item interactions, where $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ is denoted as the whole node set involving users and items, and the edge set $\mathcal{E}$ represents the observed interactions. In the multi-behavior scenario with K types of behaviors, the whole bipartite graph $\mathcal{G}$ can be segmented into K behavior subgraphs $\{\mathcal{G}_1, \mathcal{G}_2, .., \mathcal{G}_K\}$ based on the interaction behavior type. To reflect the multi-behavior interaction data, we define a tensor $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|\times K}$, where the individual element $x^k_{u,i} \in \mathbf{X}$ equals to 1 if u interacted with i under behavior k, otherwise $x^k_{u,i} = 0$. Therefore, the task of multi-behavior recommendation is formulated as:

Input: The multi-behavior interaction tensor $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|\times K}$ under multiplex K types of behaviors.

Output: A recommendation model that estimates the probability that a user u interacts with item i under the K-th behavior, *i.e.*, the target behavior.

3 METHODOLOGY

In this section, our MBSSL framework is presented in detail. The architecture of MBSSL is depicted in Figure 1, which encapsulates three key components: i) behavior-aware graph neural network which collectively captures behavioral context and dependencies (Section 3.1), ii) inter-behavior self-supervised learning to facilitate knowledge transfer, and iii) intra-behavior self-supervised learning to counteract noisy interactions.

3.1 Behavior-aware Graph Neural Network

Motivated by the message-passing architecture in GNNs, we develop a behavior-aware graph neural network to capture complex CF signals between nodes and among behaviors. More concretely, we first obtain contextualized node embeddings under each behavior subgraph via behavior-specific embedding propagation. Thereafter, we enhance the embeddings with personalized behavioral correlations via cross-behavior dependency modeling.

3.1.1 Behavior-specific Embedding propagation. Under such a multi-behavior setting, we first construct each bipartite behavior subgraph $\mathcal{G}_k$ according to the interaction behavior type k and then we perform embedding propagation on each subgraph to obtain the representation of each node under each behavior. In order to explicitly manifest the discriminative semantic of each behavior and capture the contextualized user preference, we also embed behaviors on top of nodes [4, 6, 23] and incorporate the representations of each behavior into the message-passing paradigm as:

$$\mathbf{e}^{(l+1)}_{u,k} = \text{LeakyRelu}(\mathbf{W}^{(l)} \cdot \text{mean}(\{\mathbf{e}^{(l)}_{i,k} \odot \mathbf{e}^{(l)}_{k} : i \in \mathcal{N}_{u,k}\})), \quad (1)$$

where $\mathbf{e}^{(l+1)}_{u,k}$ denotes the embedding of node u under behavior k in the $(l+1)$-th propagation layer; $\mathcal{N}_{u,k}$ is the set of immediate neighbors of u under behavior type k; $\mathbf{W}^{(l)}$ is layer-specific and $\odot$ denotes the element-wise product of two vectors; $\mathbf{e}^{(l)}_k$ represents the embedding of behavior k in the l-th layer, which is updated by multiplying another layer-specific parameter $\mathbf{W}_b$:

$$\mathbf{e}^{(l+1)}_{k} = \mathbf{W}^{(l)}_b \mathbf{e}^{(l)}_{k}. \quad (2)$$

3.1.2 Cross-behavior dependency modeling. Given that different behaviors would interweave with each other in an implicit manner and the correlations among behaviors vary by user, we leverage the self-attention [20] mechanism to model the cross-behavior dependency.

Specifically, we concatenate the embeddings of node u under all the behaviors as $\boldsymbol{e}_u \in \mathbb{R}^{K\times d}$, and then the coefficients $\mathbf{a}_{u,k} \in \mathbb{R}^K$ reflecting the dependency between behavior k and other behaviors for user u are computed by:

$$\mathbf{a}_{u,k} = \text{softmax}((\mathbf{W}^k_2)^T \tanh((\boldsymbol{e}_u \mathbf{W}^k_1)^T)), \quad (3)$$

where $\mathbf{W}^k_1 \in \mathbb{R}^{d\times d'}, \mathbf{W}^k_2 \in \mathbb{R}^{d'}$ are two behavior-specific parameters and d' is the output dimension size. Hence, the enhanced embedding of node u under behavior k can be readily calculated as:

$$\boldsymbol{e}_{u,k} = \mathbf{a}_{u,k} \boldsymbol{e}_u. \quad (4)$$

Since the embeddings of different layers express different connections, we utilize the mean-pooling to integrate the embeddings of all layers as follows:

$$\mathbf{e}_{u,k} = \frac{1}{L}\sum_{l=0}^{L-1} \mathbf{e}^{(l)}_{u,k}; \quad \mathbf{e}_k = \frac{1}{L}\sum_{l=0}^{L-1} \mathbf{e}^{(l)}_{k}; \quad (5)$$

3.2 Multi-behavior Self-supervised Learning

As mentioned before, sparse supervision signals of the target behavior may lead to severe bias of learned representations compared with those of auxiliary behaviors. Besides, the overlook of noisy interactions brought from auxiliary behaviors would exaggerate the immoderate reliance on certain interactions. Accordingly, we introduce a novel self-supervised learning paradigm to conduct self-discrimination contrastive learning from both inter-behavior and intra-behavior levels.

3.2.1 Inter-behavior Self-supervised Learning. In view of the fact that supervision signals in auxiliary behaviors are much richer than that in the target behavior, we perform selective contrastive learning between auxiliary behaviors and the target behavior to enable knowledge transfer, and thus alleviate the data sparsity at the first step.

In particular, each auxiliary behavior k from $\{1, 2, ..., K-1\}$ is to be contrasted with the target behavior K to provide distinct semantics. The common practice will treat the views of the same node as positive pairs (*i.e.*, $\{(\mathbf{e}_{u,K}, \mathbf{e}_{u,k})|u \in \mathcal{U}\}$), and the views of any different nodes as the negative pairs (*i.e.*, $\{(\mathbf{e}_{u,K}, \mathbf{e}_{v,k})|u, v \in \mathcal{U}, u \neq v\}$). However, in the recommendation setting which encapsulates various interactions, the same two subjects will have some commonalities (*e.g.*, users share similar preferences or items have similar attributes.). In this case, the constructed negative pairs following the common practice are likely to include many false negatives (*i.e.*, highly similar nodes), which will discard true semantic information [17]. Therefore, we propose to find potential false negatives based on the calculated similarity score using swing algorithm [37] and eliminate them when contrasting node pairs.

To be more specific, the similarity score of two users u, v in subgraph $\mathcal{G}_k (k \in \{1, 2, ..., K\})$ using swing algorithm is calculated as:

$$S_{\mathcal{G}_k}(u, v) = \sum_{i\in\mathcal{N}_{u,k}\cap\mathcal{N}_{v,k}} \sum_{j\in\mathcal{N}_{u,k}\cap\mathcal{N}_{v,k}} \frac{1}{\alpha + |\mathcal{N}_{i,k} \cap \mathcal{N}_{j,k}|} \quad (6)$$

where α is a smoothing coefficient. Then the final similarity score $S(u,v)$ is the average of the score in each subgraph $S_{\mathcal{G}_k}(u,v)$.

Then we define the false negatives for u as users with top-N similarity scores, $FN(u) = \{v|S(u,v) \in top(S(u),N)\}$. Accordingly, the inter-behavior contrastive loss between the target behavior K and auxiliary behavior k which eliminates specific false negatives is defined as:

$$\mathcal{L}^{K,k}_{ssl_1,user} = \sum_{u\in\mathcal{U}} -\log\frac{\exp(\phi(\mathbf{e}_{u,K},\mathbf{e}_{u,k})/\tau)}{\sum_{v\in\mathcal{U}\setminus FN(u)}\exp(\phi(\mathbf{e}_{u,K},\mathbf{e}_{v,k})/\tau)}, \quad (7)$$

where τ is the temperature hyperparameter in *softmax* and $\phi(\cdot)$ denotes the inner-product of two vectors. When analogously combined with the contrastive loss of the item side, the inter-behavior contrastive loss between the target behavior K and auxiliary behavior k is $\mathcal{L}^{K,k}_{ssl_1} = \mathcal{L}^{K,k}_{ssl_1,user} + \mathcal{L}^{K,k}_{ssl_1,item}$. Therefore, the ultimate inter-behavior contrastive loss is the sum of each pair of auxiliary behavior and target behavior as: $\mathcal{L}_{ssl1} = \mathcal{L}^{K,1}_{ssl_1} + ... + \mathcal{L}^{K,k}_{ssl_1} + ... + \mathcal{L}^{K,K-1}_{ssl_1}$.

3.2.2 Intra-behavior Self-supervised Learning. For alleviating the skewed data distribution across different behaviors, the inter-behavior self-supervised learning encourages the similarity of node representations under the target behavior and auxiliary behaviors. However, in view of the higher proportion of noisy interactions under auxiliary behaviors, more noises would be implicitly transferred into the target behavior as well, making the learned representations dominated by auxiliary signals while lose the intrinsic semantics under the target behavior. Hence, we devise an intra-behavior self-supervised learning to generate and contrast structurally-augmented views of the target behavior subgraph, in which way we consolidate and amplify the impact of supervision signals within the target behavior itself to counteract the negative transfer towards noise distributions in auxiliary behaviors. Specifically, we first generate two augmented views $\mathcal{G}^1_K, \mathcal{G}^2_K$ from the target behavior subgraph by performing edge dropout introduced in [32]. We denote $\mathcal{G}_K = (\mathcal{V}_K, \mathcal{E}_K)$ as the target behavior subgraph, and then two augmented views are elaborated as:

$$\mathcal{G}^1_K = (\mathcal{V}_K, \mathbf{M}_1 \odot \mathcal{E}_K), \quad \mathcal{G}^2_K = (\mathcal{V}_K, \mathbf{M}_2 \odot \mathcal{E}_K), \quad (8)$$

where $\mathbf{M}_1, \mathbf{M}_2 \in \{0,1\}^{|\mathcal{E}_K|}$ are two random masking vectors controlling the kept edge set with the same dropout out ratio ρ.

After encoding the two augmented views together with auxiliary behavior subgraphs respectively, we obtain the node representations of augmented views, denoted as $\mathbf{e}^1_{u,K}, \mathbf{e}^2_{u,K}$. Similar to Section 3.2.1, we then contrast views at the node scale with positive pairs as $\{(\mathbf{e}^1_{u,K}, \mathbf{e}^2_{u,K})|u \in \mathcal{U}\}$ and negative pairs as $\{(\mathbf{e}^1_{u,K}, \mathbf{e}^2_{v,K})|u,v \in \mathcal{U}, u \neq v\}$. The optimization objective of user side is calculated via InfoNCE loss likewise as:

$$\mathcal{L}_{ssl_2,user} = \sum_{u\in\mathcal{U}} -\log\frac{\exp(\phi(\mathbf{e}^1_{u,K}, \mathbf{e}^2_{u,K})/\tau)}{\sum_{v\in\mathcal{U}}\exp(\phi(\mathbf{e}^1_{u,K}, \mathbf{e}^2_{v,K})/\tau)}. \quad (9)$$

Analogously combining the loss of item side, we obtain the final objective function of intra-behavior self-supervised learning as $\mathcal{L}_{ssl_2} = \mathcal{L}_{ssl_2,user} + \mathcal{L}_{ssl_2,item}$.

3.3 Adaptive Multi-behavior Optimization

In this subsection, we present our optimization objective and a novel adaptive optimization method for multi-behavior setting.

3.3.1 Optimization Objective. For learning model parameters in an effective and stable way, we leverage a recently proposed non-sampling objective for recommendation [6] which has been proved to be superior to the traditional Bayesian Personalized Ranking (BPR) loss. For a specific batch of users $\mathcal{B}$ and the whole item set $\mathcal{I}$, the non-sampling recommendation loss under behavior k is:

$$\begin{aligned}\mathcal{L}^k_{rec} = &\sum_{u\in\mathcal{B}}\sum_{i\in\mathcal{I}^{k+}_u}\left((c^{k+}_i - c^{k-}_i)(\hat{x}^k_{u,i})^2 - 2c^{k+}_i\hat{x}^k_{u,i}\right)\\ &+\sum_{m=1}^{d}\sum_{n=1}^{d}\left((\mathbf{e}^{(m)}_k\mathbf{e}^{(n)}_k)(\sum_{u\in\mathcal{B}}\mathbf{e}^{(m)}_{u,k}\mathbf{e}^{(n)}_{u,k})(\sum_{i\in\mathcal{I}}\mathbf{e}^{(m)}_{i,k}\mathbf{e}^{(n)}_{i,k})\right),\end{aligned} \quad (10)$$

where $\hat{x}^k_{u,i}$ denotes the estimated probability of user u interacting with item i under behavior k; $\mathcal{I}^{k+}_u$ represents the interacted items of user u under behavior k and c^{k+}_i, c^{k-}_i are two hyperparameters. Then the loss of the main supervised recommendation task is the weighted sum of recommendation loss under each behavior with λ_k as the coefficient:

$$\mathcal{L}_{rec} = \sum_{k=1}^{K}\lambda_k\mathcal{L}^k_{rec}. \quad (11)$$

In our framework, we aim to boost the recommendation performance through customized self-supervised learning tasks, so the ultimate learning objective is the combination of main supervised task loss and SSL task loss, which is defined as:

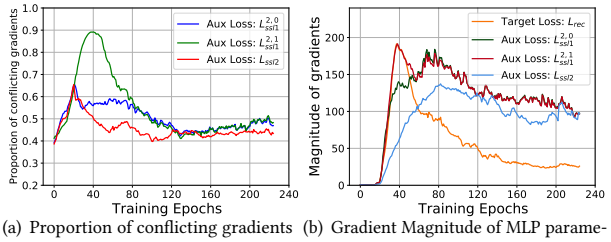
$$\begin{aligned}\mathcal{L} &\Rightarrow \mathcal{L}_{rec} + \mathcal{L}_{ssl1} + \mathcal{L}_{ssl2}\\ &\Rightarrow \mathcal{L}_{rec} + \mu_1\mathcal{L}^{K,1}_{ssl1} + ... + \mu_{K-1}\mathcal{L}^{K,K-1}_{ssl1} + \gamma\mathcal{L}_{ssl2},\end{aligned} \quad (12)$$

where $\mu_k (k = 1, 2, ..., K-1)$ and γ are hyperparameters to control the strength of the corresponding SSL task.

For one batch of the training data, the computational cost of $\mathcal{L}_{rec}$ is $O\big((|\mathcal{B}|+|\mathcal{I}|)Kd^2+|\mathcal{E}|d\big)$, where $|\mathcal{E}|$ is the total number of positive interactions. Besides, all of the SSL task losses share the same time complexity, which equals $O\big(|\mathcal{B}|d(2+|\mathcal{V}|)\big)$. Therefore, the overall complexity of $\mathcal{L}$ is $O\left(|\mathcal{B}||\mathcal{V}|Kd + |\mathcal{E}|d\right)$, which is comparable to SOTA multi-behavior recommendation models, (*e.g.*, CML, GHCF).

3.3.2 Hybrid Manipulation on Gradients. Similar to Equation (12) where SSL tasks are viewed as auxiliary tasks to improve recommendation, existing SSL recommendation models [21, 31, 32, 38] jointly optimize the main supervised recommendation task along with the SSL task. However, they suffer from two limitations. On one hand, they neglect the potential for a significant optimization imbalance across tasks, which could deteriorate the performance of the target task. This problem becomes particularly prominent when utilizing SSL tasks to serve as auxiliary tasks because SSL pretext tasks barely coincide with the target task perfectly under a manually-designed manner. Take our MBSSL model as an example, Figure 2(a) and 2(b) highlight two examples from Beibei, which respectively demonstrate the imbalance phenomenon of gradient direction and gradient magnitude between SSL tasks and the target task.

On the other hand, tuning the weights of multiple auxiliary task losses (*i.e.*, μ_k, γ in Equation (12)) is time-consuming, and fixed weights do not apply to all the batches throughout the dynamic training process. Towards this end, we develop an adaptive optimization method for SSL recommendation models with hybrid manipulation on both the magnitude and direction of gradients.



(a) Proportion of conflicting gradients (b) Gradient Magnitude of MLP parameters

Figure 2: The illustration of large differences of gradient direction and magnitude between auxiliary tasks and target task.

In MBSSL, each SSL task loss (*i.e.*, $\mathcal{L}_{ssl1}^{K,1}, ...\mathcal{L}_{ssl1}^{K,K-1}, \mathcal{L}_{ssl2}$) can be viewed as an auxiliary task loss denoted as $\mathcal{L}_{aux,i}$ while the main supervised task loss can be rewritten as the target task loss $\mathcal{L}_{tar}$. Let θ denote the set of bottom shared parameters; t denotes the t-th training iteration within one epoch; $\mathbf{G}_{tar}^t, \mathbf{G}_{aux,i}^t$ respectively denote the gradient of the target task and auxiliary task with respect to θ, *i.e.*, $\mathbf{G}_{tar}^t = \nabla_\theta \mathcal{L}_{tar}^t$ and $\mathbf{G}_{aux,i}^t = \nabla_\theta \mathcal{L}_{aux,i}^t$. Hence, our goal is to balance $\mathbf{G}_{aux,i}^t$ and $\mathbf{G}_{tar}^t$ during each iteration via modifying the direction and magnitude of $\mathbf{G}_{aux,i}^t$ adaptively.

Intuitively, gradient with larger magnitude will dominate the optimization trend, so in the case where $||\mathbf{G}_{aux,i}^t|| \geq ||\mathbf{G}_{tar}^t||$, the optimizer prefers to approach the i-th auxiliary task rather than the target task which leads to the performance degradation. As a solution, we dedicate to alter the direction and magnitude of auxiliary gradient $\mathbf{G}_{aux,i}^t$ which embraces larger magnitude than the target task $\mathbf{G}_{tar}^t$, so as to guide the optimization towards the target task. In terms of auxiliary gradients with smaller magnitude as well as conflicting directions, we keep them unaltered for preventing overfitting. To be more specific, we first modify the gradient direction via projecting the auxiliary gradient to the normal plane of the target gradient if they conflict with each other, *i.e.*, their cosine similarity is negative. The projection strategy is formulated as:

$$\mathbf{G}_{aux,i}^t = \mathbf{G}_{aux,i}^t - \frac{\mathbf{G}_{aux,i}^t \cdot \mathbf{G}_{tar}^t}{||\mathbf{G}_{tar}^t||^2} \mathbf{G}_{tar}^t. \tag{13}$$

Although the amount of destructive gradient interference has been reduced via direction modification, large gradient magnitudes of auxiliary tasks still hinder the optimization towards the target task. Therefore, we further balance the gradient magnitude based on [16, 22] with a relax factor r to control the magnitude proximity between $\mathbf{G}_{aux,i}$ and $\mathbf{G}_{tar}$:

$$\mathbf{G}_{aux,i}^t = r * \frac{||\mathbf{G}_{tar}^t||}{||\mathbf{G}_{aux,i}^t||} \mathbf{G}_{aux,i}^t + (1-r) * \mathbf{G}_{aux,i}^t. \tag{14}$$

Through such a hybrid manipulation altering both the gradient direction and magnitude of auxiliary tasks, the learning process could readily optimize towards the target task, thus boosting the recommendation performance of target behavior. The complete update procedure of hybrid manipulation is shown in Algorithm 1.

3.4 Model Analysis

In this section, we compare the proposed SSL paradigm with that of existing representative work on multi-behavior recommendation.

Algorithm 1: The Hybrid Manipulation on Gradients.

Input: Initial shared parameters θ_0; Relax factor r; Learning rate η; Auxiliary task loss $\mathcal{L}_{aux,i} \in \{\mathcal{L}_{ssl1}^{K,1}, ..., \mathcal{L}_{ssl1}^{K,K-1}, \mathcal{L}_{ssl2}\}$; Target task loss $\mathcal{L}_{tar} = \mathcal{L}_{rec}$;
Output: Updated shared parameters θ_T

for $t = 1$ *to* T **do**
  $\mathbf{G}_{tar}^t = \nabla_\theta \mathcal{L}_{tar}^t$
  for $i = 1$ *to* K **do**
    $\mathbf{G}_{aux,i}^t = \nabla_\theta \mathcal{L}_{aux,i}^t$
    if $||\mathbf{G}_{aux,i}^t|| > ||\mathbf{G}_{tar}^t||$ **then**
      $\mathbf{G}_{aux,i}^t \leftarrow \mathbf{G}_{aux,i}^t - \frac{\mathbf{G}_{aux,i}^t \cdot \mathbf{G}_{tar}^t}{||\mathbf{G}_{tar}^t||^2} \mathbf{G}_{tar}^t$
      $\mathbf{G}_{aux,i}^t \leftarrow r * \frac{||\mathbf{G}_{tar}^t||}{||\mathbf{G}_{aux,i}^t||} \mathbf{G}_{aux,i}^t + (1-r) * \mathbf{G}_{aux,i}^t$
    end
  end
  $\mathbf{G}^t \leftarrow \mathbf{G}_{tar}^t + \mathbf{G}_{aux,1}^t + ... + \mathbf{G}_{aux,K}^t$
  $\theta_t \leftarrow \theta_{t-1} - \eta * \mathbf{G}^t$
end
return θ^T

Comparison with CML. Likewise, the cross-behavior SSL in CML is performed between each auxiliary and target behavior pair to capture cross-type behavior dependency. Specifically, the SSL paradigm follows the conventional rule, *i.e.*, the views of any different users will be regarded as negative pairs. However, we can conclude that users may share similar preferences based on rich semantics and huge data volume of behaviors, which means the common practice may lead to many false negative pairs. Therefore, in our inter-behavior SSL, we selectively construct negative pairs based on the calculated structural node similarities to facilitate the knowledge transfer between auxiliary and target behaviors.

Comparison with S-MBRec. The star-style SSL in S-MBRec constructs additional positive samples by finding similar users based on the data under target behavior. However, the data is so sparse that the calculated node similarities are not reliable. What's worse, the negative transfer will be further amplified under the current SSL paradigm which encourages the alignment between unreliable positive samples. Accordingly, we aim to make full use of data under all the behaviors to select potential similar users with a high confidence level, and refuse to augment positive samples accounting for the robustness against interaction noises [7, 17].

All of the existing work solely rely on the inter-behavior SSL to handle the data sparsity issue, which is not enough though. And the inter-behavior SSL is likely to introduce noises from auxiliary behaviors. As a solution, we conduct intra-behavior SSL within the target behavior itself with an aim to counteract the auxiliary noises via amplifying the impact of the target behavior.

4 EXPERIMENTS

In this section, we conduct extensive experiments on five real-world datasets to evaluate our proposed model. In particular, we aim to figure out the following research questions:

- **RQ1**: How effective is MBSSL in multi-behavior recommendation scenarios compared to existing methods?
- **RQ2**: How do the sub-modules of MBSSL affect the recommendation performance?
- **RQ3**: How robust is MBSSL to data sparsity under the target behavior and to noisy interactions from auxiliary behaviors?

Table 1: The statistics of datasets.

Dataset	#User	#Item	#Interactions	Interaction Behavior Type
Beibei	21,716	7,977	3,338,068	{Page View, Cart, Purchase}
Taobao	48,749	39,493	1,952,931	{Page View, Cart, Purchase}
Tmall	31,882	31,232	1,451,219	{Page View, Favorite, Cart, Purchase}
IJCAI-Contest	17,435	35,920	799,368	{Page View, Favorite, Cart, Purchase}
Videos	29,197	23,251	2,002,201	{Click, Like, Comment, Download }

• **RQ4**: Which strategy of combining the manipulation on gradient direction and magnitude is the best?
• **RQ5**: What is the influence of different hyperparameter settings on recommendation performance?

4.1 Experimental Settings

4.1.1 Datasets. To evaluate the performance of MBSSL, we experiment on four public datasets *i.e.*, Beibei, Taobao, Tmall, IJCAI-Contest which are publicly available split datasets and one internal dataset, Videos. Table 1 summarizes the statistics of all the datasets.

- **Beibei**. This is the dataset collected from Beibei, the largest infant product e-commerce platform in China. It involves three types of interaction behavior, *i.e.*, *page view, cart, purchase*, among which *purchase* is the target behavior.
- **Taobao**. This is an open dataset obtained from the largest e-commerce site Taobao, which contains the same interaction type with Beibei. We directly use the processed dataset in GHCF[4].
- **Tmall**. This is another processed dataset on Taobao provided by CML [31], which contains one additional behavior *favorite*.
- **IJCAI-Contest**. This dataset is provided in IJCAI15 Challenge. It is collected from a business-to-customer retail system, which shares the same behavior types with the Tmall data.
- **Videos**. This is a dataset collected from an online short video platform. There are totally four types of interaction behavior between users and videos, *i.e.*, *click, like, comment, download*. In this dataset, we regard *download* as the target behavior.

4.1.2 Baselines. We compare our method with the following state-of-the-art methods from three types: Single-Behavior, Self-supervised Learning and Multi-Behavior recommendation models. For Single-Behavior and Self-supervised Learning methods, we normally utilize target behavior data in the same way as CML [31] or GHCF [4] to build the model. However, in order to eliminate the performance gap resulting from different volumes of training data, we additionally conduct experiments on these models by treating all the behaviors as the same type, which leads to stronger baselines.

Single-Behavior Recommendation Models.

- **NGCF** [29]: It is a neural collaborative filtering method utilizing GNNs, with an aim to capture high-order connections.
- **LightGCN** [13]: It simplifies the GCN structure to improve training efficiency and generalization ability for recommendation.

Self-supervised Learning Recommendation Models.

- **SGL** [32]: This method explores SSL on graph structure and accordingly devises three unified augmentation operators including node dropout, edge dropout and random walk.
- **SimGCL** [39]: It is a simple yet effective graph-augmentation-free contrastive learning method that can regulate the uniformity in a smooth way.

Multi-Behavior Recommendation Models.

- **MATN** [34]: It proposes an attention-based transformer encoder to help preserve cross-type behavior collaborative signals and type-specific behavior contextual information.
- **MBGMN** [36]: It enhances multi-behavior modeling with Graph Meta Network which incorporates the meta learning paradigm.
- **CML** [31]: It designs a contrastive learning framework for multi-behavior recommendation and further utilizes meta learning to learn the customized weights for each user.
- **EHCF** [6]: It conducts knowledge transfer among behaviors and proposes a novel non-sampling objective for multi-behavior recommendation.
- **S-MBRec** [10]: It is another SSL-based model which considers the discrepancies and commonalities of multiple behaviors.
- **GHCF** [4]: It is an improvement over EHCF which relies on the GNNs to model the complex high-hop user-item correlations.

4.1.3 Evaluation Methodology. For a fair comparison with various models on recommendation, we adopt the widely-used leave-one-out evaluation and two ranking metrics, *Recall@K* and *NDCG@K*. Note that we utilize the same evaluator as EHCF [4] or GHCF [6], *i.e.*, we rank all the items except positive ones for each user, which is more persuasive than randomly sampling a subset of non-interactive items for each user (*e.g.*, in MBGMN [36] or CML [31], pairing each positive item instance with 99 randomly sampled items).

4.1.4 Parameter Settings. We implement our MBSSL model with Pytorch and the model is optimized using the Adam optimizer with learning rate 0.001 during the training phase. We set α to 0.5 for swing algorithm. The batch size ranges from {256, 512, 1024}. By default, the size of the latent dimension is set as 64 and the number of propagation layers is 4. In addition, the negative weight of the non-sampling loss is chosen from {0.1, 0.01} for all the datasets. For preventing overfitting, we set the embedding dropout ratio as 0.3 and the edge dropout ratio ρ as 0.5.

4.2 Performance Comparison (RQ1)

In the evaluation, we first perform experiments to make recommendations on five datasets where we set the recommendation length K as 10, 50. From Table 2, we summarize the following observations:

- Our model MBSSL generally outperforms all the baselines on all datasets, with the improvements over the best baseline ranging from 18.64% to 19.71% in terms of Recall@10 and NDCG@10. The significant improvements can be attributed to two main reasons: i) Through inter-behavior and intra-behavior SSL, our model effectively addresses the data sparsity under the target behavior and noisy interactions from auxiliary behaviors, which are two key problems of multi-behavior recommendation. ii) The proposed hybrid manipulation method on gradients empowers the capability of balancing the optimization between the SSL task and the main supervised recommendation task, which further preserves the recommendation performance.
- Interestingly, we find that some multi-behavior models (*e.g.*, MATN, MBGMN) may perform worse than some single-behavior models which treat all the behaviors as the same type. (*e.g.*, LightGCN). The poorer performance of MBGMN and MATN may suggest their disadvantages of differentiating behavior types. Whereas, methods like EHCF, GHCF and MBSSL are superior to single-behavior models in general, which highlights the significance of behavior modeling.

Table 2: The performance comparison on five datasets. Note that baselines with the "all" suffix use data from all the behaviors to build the single-behavior model. The best results are illustrated in bold and the number underlined is the runner-up. And the number with a star (∗) indicates the result is statistically evaluated with $p < 0.05$ under t-test compared to other baselines.

	Beibei				Taobao				Tmall			
	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50
NGCF [29]	0.0268	0.1008	0.0124	0.0281	0.0135	0.0330	0.0073	0.0115	0.0155	0.0399	0.0078	0.0131
LightGCN [13]	0.0383	0.1366	0.0190	0.0399	0.0163	0.0477	0.0085	0.0152	0.0172	0.0479	0.0078	0.0144
NGCF_all [29]	0.0286	0.1084	0.0139	0.0308	0.0144	0.0391	0.0075	0.0127	0.0341	0.0752	0.0183	0.0272
LightGCN_all [13]	0.0552	0.1626	0.0284	0.0514	0.0405	0.0991	0.0223	0.0350	0.0328	0.0821	0.0174	0.0279
SGL [32]	0.0418	0.1383	0.0201	0.0405	0.0431	0.0775	0.0257	0.0332	0.0340	0.0700	0.0185	0.0264
SimGCL [39]	0.0479	0.1548	0.0231	0.0460	0.0395	0.0772	0.0226	0.0308	0.0347	0.0775	0.0193	0.0286
SGL_all [32]	0.0597	0.1699	0.0310	0.0545	0.0506	0.1214	0.0278	0.0430	0.0281	0.0779	0.0148	0.0255
SimGCL_all [39]	0.0603	0.1697	0.0311	0.0545	0.0502	0.1242	0.0274	0.0433	0.0327	0.0903	0.0166	0.0290
MATN [34]	0.0466	0.1224	0.0261	0.0422	0.0259	0.0760	0.0132	0.0239	0.0154	0.0558	0.0075	0.0161
MBGMN [36]	0.0427	0.1439	0.0227	0.0441	0.0485	0.1314	0.0250	0.0428	0.0263	0.0741	0.0136	0.0239
CML [31]	0.0588	0.1953	0.0292	0.0584	0.0299	0.0914	0.0150	0.0282	0.0061	0.0242	0.0027	0.0065
EHCF [6]	0.1523	0.3316	0.0817	0.1213	0.0717	0.1618	0.0403	0.0594	0.0234	0.0642	0.0116	0.0204
S-MBRec [10]	0.1697	0.3708	0.0872	0.1313	0.0814	0.1878	0.0446	0.0677	0.0261	0.0771	0.0125	0.0235
GHCF [4]	0.1922	0.3794	0.1012	0.1426	0.0807	0.1892	0.0442	0.0678	0.0353	0.0954	0.0175	0.0303
MBSSL	**0.2229**∗	**0.3806**∗	**0.1277**∗	**0.1626**∗	**0.1027**∗	**0.2120**∗	**0.0576**∗	**0.0813**∗	**0.0400**∗	**0.1101**∗	**0.0201**∗	**0.0352**∗

	IJCAI-Contest				Videos			
	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50
NGCF [29]	0.0041	0.0115	0.0026	0.0042	0.0178	0.0666	0.0084	0.0187
LightGCN [13]	0.0143	0.0323	0.0076	0.0113	0.0307	0.0803	0.0158	0.0265
NGCF_all [29]	0.0102	0.0254	0.0056	0.0089	0.0808	0.1995	0.0449	0.0704
LightGCN_all [29]	0.0274	0.0669	0.0156	0.0244	0.0887	0.2060	0.0491	0.0745
SGL [32]	0.0157	0.0323	0.0086	0.0120	0.0337	0.0832	0.0179	0.0420
SimGCL [39]	0.0156	0.0360	0.0083	0.0127	0.0371	0.0964	0.0199	0.0326
SGL_all [32]	0.0269	0.0644	0.0151	0.0232	0.1002	0.2179	0.0575	0.0830
SimGCL_all [39]	0.0283	0.0710	0.0163	0.0254	0.1053	0.2282	0.0595	0.0860
MATN [34]	0.0113	0.0347	0.0053	0.0102	0.0053	0.0373	0.0023	0.0089
MBGMN [36]	0.0305	0.0647	0.0172	0.0246	0.0365	0.0786	0.0220	0.0311
CML [31]	0.0249	0.0483	0.0142	0.0192	0.0011	0.0056	0.0005	0.0020
EHCF [6]	0.0264	0.0552	0.0152	0.0215	0.1036	0.2947	0.0530	0.0941
S-MBRec [10]	0.0292	0.0805	0.0160	0.0268	0.0911	0.1944	0.0461	0.0690
GHCF [4]	0.0305	0.0830	0.0168	0.0281	0.1687	0.3512	0.0808	0.1220
MBSSL	**0.0390**∗	**0.0952**∗	**0.0201**∗	**0.0321**∗	**0.1835**∗	**0.3768**∗	**0.0869**∗	**0.1306**∗

- SSL based models (*i.e.*, SGL and SimGCL) yield better performance than single-behavior models, which indicates that exploiting SSL on graph-structure could empower the generalization ability of GNN-based recommender models. However, the optimization imbalance of the SSL task and the recommendation task remains unexplored and this may lead to a performance decline.

4.3 Ablation Study (RQ2)

In this subsection, we evaluate the rationality of each designed module in our model, with four variants as follows:

- ***w/o CDM***: The cross-behavior dependency modeling is removed in the behavior-aware graph neural network.
- ***w/o*** SSL_{inter}: We do not perform inter-behavior SSL in the multi-behavior self-supervised learning part.
- ***w/o*** SSL_{intra}: We remove intra-behavior SSL in the multi-behavior self-supervised learning part.
- ***w/o HMG***: We disable the hybrid manipulation on gradients, *i.e.*, we do not tackle the optimization imbalance. Instead, we assign equal weights for each auxiliary loss.

The ablation study results are shown in Table 3. Note that due to space limitations, we only show the results of Recall@10 and NDCG@10 on two datasets, Beibei and Taobao. For the other datasets, the observations are similar. From Table 3, we can find:

- The cross-behavior dependency modeling plays a vital role in performance, which indicates that the self-attention mechanism has a strong capability on capturing the implicit pair-wise dependencies across behaviors.

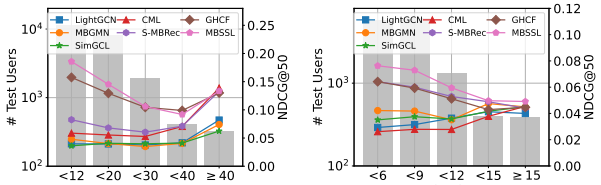
Table 3: The experimental results of ablation study.

Data	Beibei		Taobao	
Metric	Recall@10	NDCG@10	Recall@10	NDCG@10
w/o CDM	0.1923	0.1059	0.0936	0.0520
w/o SSL_{inter}	0.2025	0.1108	0.0902	0.0515
w/o SSL_{intra}	0.1994	0.1027	0.0943	0.0526
w/o HMG	0.2086	0.1131	0.0936	0.0530
MBSSL	**0.2229**	**0.1277**	**0.1027**	**0.0576**

- Removing either part of the multi-behavior self-supervised learning will undermine the performance. Specifically, the performance gap between *MBSSL* and *w/o* SSL_{inter} demonstrates the effectiveness of the inter-behavior SSL on narrowing the gap of skewed representations and alleviating the data sparsity of the target behavior. In addition, the intra-behavior SSL contributes to the performance further, indicating the necessity to address the noisy interactions from auxiliary behaviors.
- When we replace the HMG by assigning equal weights for all the auxiliary losses, the performance experiences a great decline which verifies the existence of the optimization imbalance. This also suggests that during the multi-behavior learning process, HMG adaptively rectifies the gradients to balance the auxiliary tasks, which improves the target task's performance significantly.

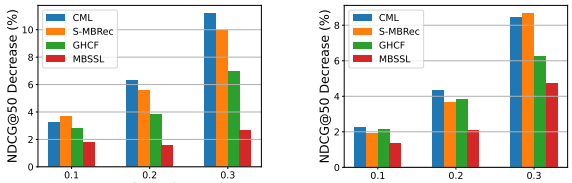
4.4 Robustness Analysis (RQ3)

4.4.1 Robustness to data sparsity. In recommender systems, the recommendation towards inactive users with few available interactions is quite challenging, so we aim to illustrate the effectiveness of our model in alleviating the data sparsity. Specifically, we split the test users into five groups based on sparsity degrees, *i.e.*, the



(a) Beibei NDCG@50 (b) Taobao NDCG@50

Figure 3: Performance comparison *w.r.t* different data sparsity degrees on Beibei and Taobao.



(a) Beibei (b) Taobao

Figure 4: Performance *w.r.t* noise ratio. The bar denotes the decrease percentage of the performance reported in Table 2.

number of interactions under the target behavior (*i.e.*, purchase), then we compute the average NDCG@50 on each group of users.

Due to space limitations, we present the comparison results with representative baselines on two public datasets Beibei and Taobao, results on other datasets would be later attached if needed. In Figure 3, the x-axis denotes the different data sparsity degrees, the left side of y-axis displays the number of test users in the corresponding group quantified by bar while the right side of y-axis is the averaged metric value quantified by line.

Based on the results, we have the following observations: i) Our model generally obtains a better performance compared to other SOTA methods on these two datasets. On Beibei dataset, despite the slight inferiority to GHCF and CML on part of active users, our model manifests a good capability of recommendation for inactive users who occupy a considerable amount of user populations. ii) The performance will encounter a slight descent when the number of interactions increases, and this may be caused by different amounts of auxiliary data. For example, on Taobao dataset, the number of auxiliary behavioral records for users who have more than 12 purchase records is much fewer than for users who have less than 9 purchase records.

4.4.2 Robustness to noisy interactions. As mentioned above, the inter-behavior SSL would inevitably introduce auxiliary noises to the representations under the target behavior, under which circumstance we propose intra-behavior SSL. In order to study the capability of intra-behavior SSL to relieve the impact of noises, we artificially add a certain proportion of noisy interactions into the auxiliary data (*i.e.*, 10%, 20%, 30%) and then compare the performance decline percentage of MBSSL with that of several representative SOTA baselines (*i.e.*, CML, S-MBRec, GHCF). Figure 4 shows the results on Beibei and Taobao.

We can observe that i) It is obvious that adding noises into the auxiliary data reduces the performance of all the methods. Generally, CML is more sensitive to the interaction noises while MBSSL presents the least performance degradation. What's more, the degradation gap between MBSSL and each baseline is more apparent as the noise ratio increases. This suggests the capability of MBSSL to figure out informative graph patterns and to relieve the over-reliance on certain interactions. ii) The performance decrease of GHCF is lower than that of CML and S-MBRec in most of the cases which indicates that the SSL paradigm in CML and S-MBRec would amplify the negative impact of noises. However, the intra-behavior in MBSSL aims to counteract the noises via consolidating the target self-supervised signals, which consequently makes MBSSL obtain a more stable performance in terms of different noise ratios.

4.5 Impact of Hybrid Strategies (RQ4)

Given the heuristic observation that there exists an obvious discrepancy between auxiliary tasks and the target task in both gradient direction and magnitude, we propose to manipulate the gradients of auxiliary tasks in these two dimensions to balance the optimization. Besides, to equip our model with a good generalization ability, we only alter auxiliary gradients with larger magnitudes than the target gradient while keeping auxiliary gradients with small magnitudes unaltered. In this subsection, with an aim to verify the rationality of our gradient manipulation method proposed in Section 3.3.2, we compare three different strategies as follows:

- **Strategy A**: In each iteration, we only alter the direction of auxiliary gradients following [40]. We manipulate the directions of all the eligible gradients regardless of their magnitudes.
- **Strategy B**: In each iteration, we only alter the magnitude of auxiliary gradients following [16], *i.e.*, we reduce the magnitude of the auxiliary gradient if it is larger than that of the target gradient and vice versa.
- **Strategy C**: In each iteration, we first alter the direction using Strategy A, and then we alter the magnitude based on Strategy B. Similarly, we conduct the manipulation on all the gradients regardless of their magnitudes.

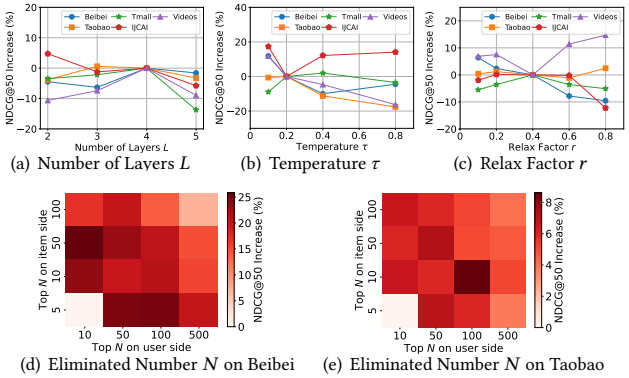
Table 4: The comparison results of different strategies.

Data	Beibei		Taobao	
Metric	Recall@10	NDCG@10	Recall@10	NDCG@10
Strategy A	0.2081	0.1155	0.0950	0.0543
Strategy B	0.2196	0.1247	0.0994	0.0567
Strategy C	0.2114	0.1217	0.0942	0.0543
MBSSL	**0.2229**	**0.1277**	**0.1027**	**0.0576**

We show the comparison results on Beibei and Taobao in Table 4, the results on the other datasets are similar. From the table, we find that i) Solely utilizing direction-based or magnitude-based methods obtains suboptimal performance, which corresponds to our observation that the SSL task yields great disparity with the target task in both direction and magnitude. ii) The performance gap between Strategy C and our model suggests that deliberately keeping the conflict between the target gradient and auxiliary gradients with small magnitudes could improve the generalization ability and ease the overfitting issue.

4.6 Hyperparameter Study (RQ5)

In this subsection, we conduct extensive experiments to examine the effects of several key hyperparameters, which include the number of propagation layers L, the temperature τ, the relax factor r and the number of eliminated false negatives N. Figure 5 shows



(a) Number of Layers L (b) Temperature τ (c) Relax Factor r

(d) Eliminated Number N on Beibei (e) Eliminated Number N on Taobao

Figure 5: Hyperparameter study of MBSSL.

the estimated performance decrease/increase percentage *w.r.t* a randomly selected datum point.

4.6.1 Effect of Propagation Layer Numbers. From Figure 5, we observe that more embedding propagation layers yield better performance due to the strengthened capability of capturing high-hop signals. However, when the number of layers exceeds 4, the performance suffers from a huge decline, which is resulted from the over-smoothing effect.

4.6.2 Effect of Temperature. The temperature is tuned carefully in {0.1, 0.2, 0.4, 0.8}. According to the curves shown in Figure 5(b) and 5(e), we find that the best selection of τ varies by dataset. However, either too small (*e.g.*, 0.1) or too large (*e.g.*, 0.8) is not appropriate which suggests that a large temperature value will undermine the ability to distinguish between negative samples. Conversely, a too small value would excessively exaggerate the effects of some negative samples while making others useless.

4.6.3 Effect of Relax Factor. Since relax factor controls the magnitude proximity between auxiliary gradients and target gradient, its selection varies according to the task. Therefore, a big relax factor is appropriate on Videos while Beibei prefers a smaller one, and the others perform best with a moderate relax factor value.

4.6.4 Effect of Eliminated False Negatives Numbers. Based on the average interaction numbers on each dataset, we determine the number of eliminated false negatives of user side and item side from {10, 50, 100, 500} and {5, 10, 50, 100} respectively. Figure 5(g) and 5(h) show the results on Beibei and Taobao where darker color means better performance. We conclude that either too small or too big value of N degrades the performance, which proves the necessity of sufficient negative samples and the effectiveness of the selective inter-behavior SSL.

5 RELATED WORK

5.1 Graph-based Recommendation

In terms of graph-based models for recommendation, recent years have witnessed the effectiveness of Graph Neural Networks (GNNs) due to the powerful capability on modeling high-order connectivity. For example, NGCF [29] is one collaborative architecture which decides the message propagation on both the graph structure and the affinity with the central node. LightGCN [13] distills a more concise and accurate GCN model for recommendation by omitting two burdensome operations, *i.e.*, feature transformation and non-linear activation. On top of methods focusing on model designs, another line of methods are dedicated to enriching the user-item bipartite graph via fusing various side information, ranging from social influences [5, 33], item-item relatedness [27, 28, 30, 41], to user and item attributes [2, 19] .

5.2 Multi-behavior Recommendation

Recent studies have attempted to explore the multiplicity via various deep learning techniques. NMTR [9] extends the neural collaborative filtering (NCF [14]) framework to multi-behavior settings, which performs a joint optimization on cascading prediction tasks. To avoid the sampling bias issue, EHCF [6] efficiently correlates the prediction of each user behavior in a transfer way without negative sampling. Later on, researchers take advantages of GNNs to explore the high-hop user-item interactions. MBGCN [18] and GHCF [4] learn discriminative behavior representations using GCNs while MBGMN [36] utilizes graph meta network to capture interaction diversity and behavior heterogeneity. To alleviate data sparsity of the target behavior, CML [31] and S-MBRec [10] incorporate self-supervised learning into multi-behavior recommendation.

5.3 Self-supervised Learning on Graphs

Inspired by the advances in CV and NLP domain, self-supervised learning (SSL) on graphs has recently been explored. InfoGraph [25] and DGI [26] learn node representations based on mutual information maximization between a node and the local structure while GRACE [42], GCA [43] and GraphCL [12] conduct node-level same-scale contrast. When it comes to the recommendation scenario, recent studies have adopted SSL to achieve better recommendation performance. Yao et al. [38] propose a two-tower DNN architecture with uniform feature masking and dropout for self-supervised item recommendation. Wu et al. [32] further devise a unified SSL framework with three augmentation operators for graph-based recommendation.

6 CONCLUSION

In this work, we develop a novel self-supervised learning framework with an adaptive optimization method for enhancing multi-behavior recommendation. Our framework effectively captures the behavior semantics and correlations via a graph neural network incorporating the self-attention mechanism. To alleviate the data sparsity and noisy interactions issue, we contrast nodes via inter-behavior SSL and intra-behavior SSL respectively. In addition, we take the initial step to study the optimization imbalance of the SSL task and recommendation task, and design a hybrid manipulation method on gradients accordingly, which has been proved effective on five real-world datasets. In the future, we plan to design a more elaborated SSL by fully capturing the structural information based on the various interaction data.

ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (No. 61872207) and Kuaishou Inc. Chaokun Wang is the corresponding author.

REFERENCES

[1] Kristen M Altenburger and Daniel E Ho. 2019. Is Yelp actually cleaning up the restaurant industry? A re-analysis on the relative usefulness of consumer reviews. In *The World Wide Web Conference*. 2543–2550.

[2] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1358–1368.

[3] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential recommendation with graph neural networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 378–387.

[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 *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3958–3966.

[5] Chong Chen, Min Zhang, Chenyang Wang, Weizhi Ma, Minming Li, Yiqun Liu, and Shaoping Ma. 2019. An efficient adaptive transfer neural network for social-aware recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 225–234.

[6] Chong Chen, Min Zhang, Yongfeng Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Efficient heterogeneous collaborative filtering without negative sampling for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 19–26.

[7] Tsai-Shien Chen, Wei-Chih Hung, Hung-Yu Tseng, Shao-Yi Chien, and Ming-Hsuan Yang. 2021. Incremental false negative detection for contrastive learning. *arXiv preprint arXiv:2106.03719* (2021).

[8] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*. PMLR, 794–803.

[9] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2019. Neural multi-task recommendation from multi-behavior data. In *2019 IEEE 35th international conference on data engineering (ICDE)*. IEEE, 1554–1557.

[10] Shuyun Gu, Xiao Wang, Chuan Shi, and Ding Xiao. 2022. Self-supervised Graph Neural Networks for Multi-behavior Recommendation. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

[11] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical user profiling for e-commerce recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 223–231.

[12] Hakim Hafidi, Mounir Ghogho, Philippe Ciblat, and Ananthram Swami. 2020. Graphcl: Contrastive self-supervised learning of graph representations. *arXiv preprint arXiv:2007.08025* (2020).

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

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[15] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 549–558.

[16] Yun He, Xue Feng, Cheng Cheng, Geng Ji, Yunsong Guo, and James Caverlee. 2022. MetaBalance: Improving Multi-Task Recommendations via Adapting Gradient Magnitudes of Auxiliary Tasks. In *Proceedings of the ACM Web Conference 2022*. 2205–2215.

[17] Tri Huynh, Simon Kornblith, Matthew R Walter, Michael Maire, and Maryam Khademi. 2022. Boosting contrastive self-supervised learning with false negative cancellation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2785–2795.

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

[19] Zekun Li, Zeyu Cui, Shu Wu, Xiaoyu Zhang, and Liang Wang. 2019. Fi-gnn: Modeling feature interactions via graph neural networks for ctr prediction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 539–548.

[20] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).

[21] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479* (2021).

[22] Itzik Malkiel and Lior Wolf. 2020. Mtadam: Automatic balancing of multiple training loss terms. *arXiv preprint arXiv:2006.14683* (2020).

[23] 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.

[24] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World Wide Web*. 111–112.

[25] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2019. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000* (2019).

[26] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. *ICLR (Poster)* 2, 3 (2019), 4.

[27] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge graph convolutional networks for recommender systems. In *The world wide web conference*. 3307–3313.

[28] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 950–958.

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

[30] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the Web Conference 2021*. 878–887.

[31] 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.

[32] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 726–735.

[33] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. 2019. A neural influence diffusion model for social recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 235–244.

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

[35] 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.

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

[37] Xiaoyong Yang, Yadong Zhu, Yi Zhang, Xiaobo Wang, and Quan Yuan. 2020. Large Scale Product Graph Construction for Recommendation in E-commerce. *arXiv preprint arXiv:2010.05525* (2020).

[38] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2021. Self-supervised Learning for Large-scale Item Recommendations. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4321–4330.

[39] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1294–1303.

[40] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[41] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 179–188.

[42] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131* (2020).

[43] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*. 2069–2080.