2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Adaptive Fusion Multi-View Contrastive Learning With Interest Aggregation for Collaborative Filtering

Runze Feng[1] • Junping Liu[2] • Mingchao Yu[3]

[1] Wuhan Textile University, School of Computer Science and Artificial Intelligence，Hubei Wuhan 430200;

Abstract. In recent years, recommendation systems based on graph neural networks (GNN) have achieved remarkable success. Despite their effectiveness, GNN-based methods are often affected by noisy interactions in user-item data. Consequently, several approaches have adopted graph contrastive learning (GCL)to address this challenge. However, most existing GCL approaches construct contrastive views from the user-item graph, without explicitly leveraging high-order relational information(i.e., user-user and item-item relationships). Moreover, they often adopt a uniform perspective on user-item connections, neglecting the diversity of user interests.To address these limitations, we present a graph contrastive recommendation model that incorporates an adaptive multi-view fusion strategy,named AdaFCL. Specifically,to more explicitly exploit high-order information, we design an adaptive fusion module that fuses edge weights derived from both the user-item interaction graph and the high-order collaborative graph(i.e., user-user and item-item graph).Then this fusion module introduces a learnable generator based on GCN and GAT to generate low-noise contrastive views, as an alternative to traditional random perturbations. Furthermore, we design a interest aggregation module to embed users' personalized preferences into the representation learning process.Extensive experiments on three public benchmark datasets demonstrate the superiority of AdaFCL. Compared to the strongest baselines , our model improves performance by up to 9.27% for NDCG@20 and 8.67% for Recall@20.

Keywords: Information systems, Recommender systems.

1 Introduction

Recommendation systems play a pivotal role in mitigating information overload and enhancing user experience, and have become a critical component of modern information systems. They are extensively applied in e-commerc, social media, and video platforms [1], helping users discover relevant content from an overwhelming volume of information. Collaborative Filtering (CF) is among the most widely used methods in recommendation systems, capturing the preferences of similar users based on implicit feedback such as clicks and Browse history [2][3][4][5]. However, GNNs still contend with several issues in collaborative filtering [6], such as noisy interactions and a heavy reliance on observed interactions as supervisory signals. Drawing inspiration from self-supervised learning, Graph Contrastive Learning (GCL) utilizes view comparison and consistency constraints to learn more robust user and item representations, and has become a principal methodology in recommendation algorithm research. Conventional approaches typically construct contrastive views by applying random perturbations to the interaction graph, such as SGL [7] and SLRec [8], or by perturbing the learned node representations, as in SimGCL [9]. Furthermore, some methods generate self-supervised signals by heuristically constructing views and hypergraphs, thereby implicitly exploiting high-order information to some extent, as exemplified by NCL [10] and HCCF [11].

Although existing GCL-based recommendation methods have demonstrated considerable potential, they are constrained by the following limitations. To begin with, some methods rely predominantly on random perturbations (e.g., node/edge dropping,embedding perturbation) for generating contrastive views . These approaches tend to introduce extraneous noise, undermining key structural information and failing to explicitly exploit high-order relations. Furthermore, existing GCL frameworks typically treat all user-item interactions as equally significant,thereby overlooking degree of user interest in different items and lacking fine-grained interests modeling.

To address these challenges, we propose AdaFCL,a graph-contrastive recommendation method .AdaFCL generates low-noise view and explicitly exploit the use of high-order information by adaptively fusing representations from adaptive multi-view fusion strategy . Specifically, we design a view generator that leverages both Graph Attention Networks (GAT) and Graph Convolutional Networks (GCN), which learns dynamic edge weights to construct contrastive views from distinct perspectives, thereby avoiding the noise introduced by random perturbations . Subsequently, we employ an adaptive multi-view fusion module to fuse the generated representations into a unified view. In addition, we introduce an interest aggregation module to quantify user preferences for interactive items and inject interest signals into the fusion view. To inject signals of user interest intensity into representation learning, we propose an interest aggregation loss to serve as an auxiliary learning objective for the main recommendation task. Finally, AdaFCL effectively improves the performance and robustness of the recommendation system by contrastively learning the initial interaction view and the fused view enhanced by interest. Our primary contributions are as follows：

- We explicitly exploit high-order information from the user–item interaction graph through our view fusion strategy. Moreover, instead of relying on random perturbations, we employs a view generator composed of GAT and GCN to construct contrastive views. As a result, AdaFCL effectively addresses both view noise and the underutilization of high-order information.

- We propose an interest aggregation module to capture the degree of user interest in an item and integrate it into the contrastive view to further enhance the learned representation.

- Extensive experiments on multiple public datasets demonstrate that the AdaFCL framework achieves substantial performance gains over mainstream and state-of-the-art baselines. Further ablation studies and analyses confirm the effectiveness and necessity of its core components.

2 Relatedwork

2.1 GNN-based recommendation systems

Graph neural networks (GNNs) have become effective components for modeling user-item relationships in recommender systems. The principal focus has been on optimizing graph structural modeling, enhancing the efficiency of information interaction, and improving the representational capacity of user-item relationships. Early efforts such as NGCF [12] and LightGCN [5] streamlined non-linear operations within traditional message-passing frameworks. They achieved this by designing lightweight graph convolutional architectures, thereby effectively balancing model performance with computational efficiency. To address the layer depth limitation in GNNs, LR-GCCF [13] introduced residual connections to mitigate the over-smoothing phenomenon. Meanwhile, models like CAGCN [14] optimized the information aggregation process through collaborative signal filtering. Similarly, ApeGNN [15] approached this optimization from the standpoint of node-adaptive diffusion weights.

In the domain of representation learning, models such as DGCF [16] achieved fine-grained preference modeling by disentangling latent user intents. In contrast, HGCF [17] innovatively introduced hyperbolic space to augment the expressive power of embeddings. To contend with complex interaction scenarios, MBGCN [18] constructed multi-relational graph networks to capture the heterogeneity inherent in multi-behavioral interactions. MixGCF [19] proposed a continuous negative sample generation strategy to enhance training robustness. Furthermore, SSNet [20] resolved the issue of scale distortion during node neighborhood information fusion through a scale-aware aggregation mechanism. Collectively, these technical advancements span the dimensions of graph structure optimization, dynamic weight allocation, representation space expansion, and training strategy innovation. They have propelled the evolution of GNN-based recommendation systems, furnishing new theoretical frameworks and practical pathways for modeling complex user-item relationships.

2.2 GCL-based recommendation systems

In recent years, research on recommendation systems based on Graph Contrastive Learning (GCL) has achieved significant progress, with its core focusing on enhancing node representation learning in user-item interaction graphs through multi-view contrastive learning. Existing methods mainly revolve around view generation techniques, which can be roughly categorized into two types: manual view generation constructs contrastive samples by empirically perturbing the original graph structure or features—for example, SGL[7] perturbs the graph structure by randomly dropping nodes or edges, while SimGCL[9] and XSimGCL[21] enhance features by introducing implicit noise; adaptive view generation, on the other hand, dynamically optimizes views through learning mechanisms—for instance, GCARec[22] generates probabilistic connections using Gumbel softmax, and AdaGCL[23] combines graph autoencoders with denoising models to generate semantically enhanced views. Meanwhile, GCL techniques have been extended to various recommendation scenarios, such as NCL[10], which captures high-order collaborative signals through semantic center node alignment. Despite these advances in recommendation accuracy, challenges remain, including insufficient modeling of latent factors and coarse-grained preference representations. How to decouple user interests and uncover implicit interaction motivations through fine-grained contrastive objectives remains an important direction for future research.

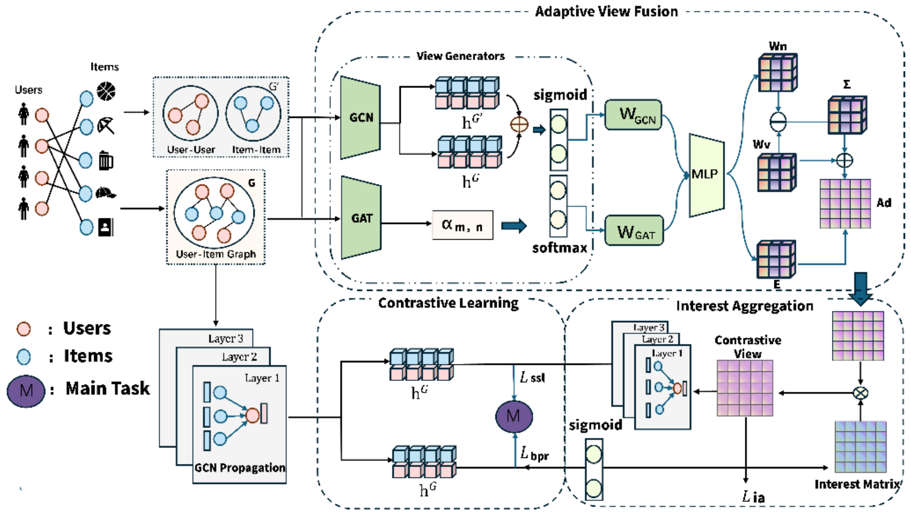

Fig. 1. The overall framework of our proposed AdaFCL model generates information rich contrastive views by adaptively fusing low-noise views from GCN and GAT and integrating a user interest aggregation module

3 Methodology

This section introduces our proposed model AdaFCL. As shown in Figure 1, AdaFCL first utilizes an adaptive view fusion mechanism to generate and fuse multi-view representations. It then employs the preference information from the interest aggregation module to achieve comprehensive multi-view information fusion. Finally, contrastive learning is applied to optimize the generated representations, enhancing recommendation performance.

3.1 Problem description

We define the set of users as $U = \{u\}$ and the set of items as $I = \{i\}$. The observed implicit feedback is represented by an interaction matrix $R \in \{0,1\}^{|U|\times|I|}$, where $R_{u,i} = 1$ indicates that an interaction (e.g., a click or purchase) exists between user u and item i; otherwise, $R_{u,i} = 0$.We use an array E of size $(2, \mathcal{E})$ to represent the connected users and items in the interaction graph, where $\mathcal{E}$ is the number of edges.

Let $\mathcal{G}$ denote the user-item interaction graph, whose adjacency matrix is defined as:

$$A_{\mathcal{G}} = \begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix}, \tag{1}$$

furthermore, we introduce a High-order Collaborative Graph, denoted as $\mathcal{G}'$. This graph incorporates two types of high-order relations: a user-user similarity graph S and an item-item similarity graph K. Its adjacency matrix is given by:

$$A_{\mathcal{G}'} = \begin{bmatrix} S & 0 \\ 0 & K \end{bmatrix}. \tag{2}$$

The primary objective of collaborative filtering is to predict unobserved interactions in the matrix R. Following the formulation in [26], this is typically achieved by estimating the probability that a user u will interact with an item i.

3.2 High-order Collaborative Graph Construction

The High-order Collaborative Graph ($\mathcal{G}'$) is composed of a user-user relation graph and an item-item relation graph. Specifically, we define users with similar preferences as collaboratively similar users, and items that share a common group of interacting users as collaboratively similar items. To quantify this collaborative similarity, we employ the Jaccard Similarity Coefficient [25] to measure the similarity between users and item of the same type, i and j.

Let N_i denote the set of neighbors for a node i in the original interaction graph (i.e., if i is a user, N_j is the set of items i has interacted with; if i is an item, N_i is the set of users who have interacted with it). The Jaccard similarity, $sim_{i,j}$, between two nodes i and j is calculated as follows:

$$sim_{i,j} = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}, \tag{3}$$

where $|N_i \cap N_j|$ is the number of common neighbors between nodes i and j, and $|N_i \cup N_j|$ is the total number of unique neighbors of both nodes. The value of $sim_{i,j}$ ranges from 0 to 1, where a higher value indicates a greater overlap in their neighborhoods and thus a stronger collaborative similarity.

To enhance computational efficiency and filter out noise—such as low-similarity connections resulting from a small number of common neighbors that lack practical significance—we do not retain all non-zero similarity scores. Instead, we employ a filtering strategy that combines Top-K selection with a similarity threshold, δ. as follows:

$$\mathcal{G}'_{i,j} = \begin{cases} sim_{i,j} & \text{if } j \in \text{Top-K and } sim_{i,j} \geq \delta \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where Top-K(i) denotes the set of K nodes with the highest similarity to node i.

3.3 Learnable View Generation

Current Graph Contrastive Learning (GCL) methods primarily rely on heuristic-based perturbations of graph structures or node features to construct contrastive views. Such strategies are prone to introducing noise or discarding critical topology information, thereby undermining the effectiveness of contrastive learning. To address this limitation, we employ a learnable view generation mechanism that adaptively extracts information from the original graph to produce high-quality, information-rich views.

GCN-based View Generation. The structural information of the original interaction graph is crucial, as it can guide the generation of views with less noise. To leverage the structural features captured by Graph Neural Networks (GNNs), we adopt the message-passing strategy proposed in LightGCN[5].

Specifically, we utilize fused node representations from the final (L-th) layer of GNN propagation. This fused representation, h_L^i, integrates information aggregated from both the original graph G and the high-order graph G':

$$h_{G,L}^i = \sum_{j \in \mathcal{N}_G(i)} \frac{1}{\sqrt{|\mathcal{N}_G(i)||\mathcal{N}G(j)|}} hL - 1^j, \tag{5}$$

$$h_{G',L}^i = \sum_{j \in \mathcal{N}_{G'}(i)} \frac{1}{|\mathcal{N}_{G'}(i)|} h_{L-1}^j. \tag{6}$$

The final L-th layer representation h_L^i is defined as the sum of these two components: $h_L^i = h_{G,L}^i + h_{G',L}^i$. These fused embeddings i and an item j, denoted as h_L^i and h_L^j, are then used to compute the edge weight $W_{\text{GCN}_{i,j}}$:

$$W_{\text{GCN}_{i,u}} = \sigma(\mathbf{h}_i^L \cdot \mathbf{h}_j^L), \tag{7}$$

where the edge connects user i and item j. This design enables the integration of long-range dependencies (from G') learned by the GNN, facilitating the generation of more informative and context-aware views.

GAT-based View Generation. Recognizing the inherent heterogeneity in user-item interactions—i.e., different items hold varying degrees of relevance for a user—we adopt an attention-based method for edge weight generation, inspired by Graph Attention Networks (GAT) [26]. This mechanism allows the model to infer context-aware edge importance.

Specifically, for an edge connecting user i and item u, we first compute an unnormalized attention coefficient, $\alpha_{i,u}$. This coefficient acts as a compatibility score derived from the initial node embeddings, $\mathbf{e}_i^u$ and $\mathbf{e}_u^v$. It is calculated by projecting the embeddings with a shared learnable vector $\mathbf{g} \in \mathbb{R}^d$ and then applying a nonlinear activation function σ:

$$\alpha_{u,i} = \sigma((\mathbf{g}^T \mathbf{e}_i^u) \cdot (\mathbf{g}^T \mathbf{e}_u^v)). \tag{8}$$

To ensure the weights are comparable, these raw coefficients are normalized across the neighborhood of user m using the softmax function. The final edge weight, $W_{\mathrm{GAT}_{u,i}}$, is computed for all items k that user m has interacted with (i.e., where $A_{m,k} = 1$):

$$W_{\mathrm{GAT}_{u,i}} = \frac{\exp(\alpha_{u,i})}{\sum_{k|A_{i,k}=1} \exp(\alpha_{i,k})}. \tag{9}$$

3.4 Adaptive Multi-View Representation Fusion.

We design an adaptive multi-view fusion module that enables the flexible integration of representation from different views.

View refinement.View-shared representations are essential for understanding user–item relationships commonly endorsed across different views. This shared representation is denoted as the view-shared edge weights W_v. To compute W_v, we employ a multilayer perceptron (MLP) [27] that directly learns and infers the shared edge weights from the edge weights of individual views. The computation is as follows:

$$\mathbf{h}_j^{(l)} = \sigma(\mathbf{W}^{(l)} \mathbf{h}_j^{(l-1)} + \mathbf{b}^{(l)}), \tag{10}$$

in this formulation, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are shared across all edges e_j in the graph. The set of shared edge weights for all edges is expressed as:

$$w_{vj} = \mathbf{h}_j^{(L)}, \tag{11}$$

$$W_v = \{w_{vj}\}_{j=1}^{M}. \tag{12}$$

In addition, due to their distinct construction paradigms (e.g., W_{GNN} emphasizes topological structure, whereas W_{GAT} focuses on interactions revealed through attention mechanisms), each view is expected to retain information beyond the shared weights W_v.

$$\bar{W}_n = W_n - W_v. \tag{13}$$

To effectively identify and exploit these view-specific signal components, we introduce the notion of view-specific edge weights, denoted as $\bar{W}_n$. The computation of $\bar{W}_n$ follows a differencing logic, aiming to isolate view-specific information by subtracting the shared component W_v from the view edge weights W_n.

View Fusion. Finally, the extracted shared and specific representations are integrated to form W_d. For each edge e_j in the graph, the final aggregated weight w_{dj} is defined as the sum of its shared weight and the corresponding view-specific weights. Accordingly, the aggregated edge weight set W_d is given by:

$$W_d = W_v + \sum_{n \in N} \bar{W_n}, \tag{14}$$

these aggregated edge weights W_d are further combined with the graph topology represented by the edge index E to yield the weighted adjacency matrix A^d:

$$A^d = \Phi(W_d, E), \tag{15}$$

this process generates the contrastive view A^d based on the given edge index E and the associated learned edge weights W_d.

3.5 Interest Aggregation

Most existing graph neural network-based recommendation methods rely on static adjacency matrices for representation learning. While effective, such approaches often overlook the heterogeneity in user interests toward different items, thereby limiting the expressiveness of the resulting representations. To address this, we introduce a learnable interest aggregation module designed to incorporate user interest signals into the contrastive view, enhancing the personalization and robustness of the structural representations.

The core of this mechanism lies in constructing an interest aggregation matrix **P**, which quantifies a user's degree of preference for their interacted items.

Specifically, for a user i and an item j, we compute this preference score using the dot product of their final embedding vectors, normalized by the sigmoid function:

$$P_{ij} = \begin{cases} \text{sigmoid}((\mathbf{h}_L^i)^T \mathbf{h}_L^j), & \text{if } A_{ij} = 1 \\ 0, & \text{otherwise} \end{cases}, \tag{16}$$

$\mathbf{h}_L^i$and $\mathbf{h}_L^j$is from initial view, learning to say, $A_{ij} = 1$said observed the interaction between them. The sigmoid function maps the dot product to a $[0,1]$ interval, providing a probabilistic interpretation of the interaction strength:

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}. \tag{17}$$

Subsequently, we apply element-wise multiplication to inject these interest signals into the aggregated contrastive view $\mathbf{A}^d$, yielding a refined, interest-aware view:

$$\bar{\mathbf{A}}^d = \mathbf{A}^d \odot \mathbf{P}. \tag{18}$$

This operation preserves the graph structure while adjusting edge weights to emphasize high-interest interactions during message passing. To further steer the model, we introduce an interest aggregation loss that encourages higher weights for observed user-item pairs:

$$\mathcal{L}_{IA} = -\sum_{(i,j)|A_{ij}=1} \log(\bar{A}^{d}_{ij}), \tag{19}$$

by maximizing the log-probability of observed interaction edges, this loss term enables the model to encode user interests effectively. This final, refined contrastive view, $\bar{\mathbf{A}}^{d}$, is then used for multi-layer message passing in the graph neural network.

3.6 Contrastive Learning

Existing stochastic perturbation-based view generation strategies may introduce noise into the original graph and typically employ two augmented views for contrastive learning. As a result, the original view does not directly contribute to the contrastive loss. In our method, we utilize adaptively fused contrastive views with integrated interest aggregation, which substantially reduces the risk of noise. Therefore, we perform contrastive learning between the generated contrastive views and the original graph representation.

We adopt the InfoNCE loss [28], denoted as L_{cl}, to maximize the agreement between the same node across different views:

$$L^{u}_{cl} = \sum_{u \in \mathcal{U}} -\log \frac{\exp(\phi(z_u, g_u)/\tau)}{\sum_{u' \in \mathcal{U}} \exp(\phi(z_u, g_{u'})/\tau)}, \tag{20}$$

z_x and g_x denote the representations of node x in the original view Z and the generated contrastive view G, respectively. $\phi(\cdot,\cdot)$ is a similarity function, and τ is a temperature hyperparameter.

3.7 Loss function

The final training objective of the model is a multi-task loss function L, which integrates the primary recommendation loss, contrastive learning loss, interest aggregation loss, and an L2 regularization term. Recommendation loss: We adopt the Bayesian Personalized Ranking (BPR) loss L_{bpr}:

$$L_{bpr} = \sum_{(u,i,j) \in \mathcal{O}} -\log\sigma(\hat{y}_{u,i} - \hat{y}_{u,j}), \tag{21}$$

where $\mathcal{O}$ denotes the set of training triples (user, positive item, negative item), and σ is the sigmoid function.Total loss:

$$L = L_{bpr} + \lambda_1 L_{cl} + \lambda_2 L_{ia} + \lambda_3 \parallel \Theta \parallel^2_2, \tag{22}$$

where Θ denotes all trainable parameters in the model, and $\lambda_1, \lambda_2, \lambda_3$ are hyperparameters that balance the contributions of each loss component. The model is trained end-to-end by minimizing the total loss L.

4 Experimental results and analysis

In this section, we conduct extensive experiments for model evaluation to answer the following key research questions:

1. How effective is our AdaFCL compared to various state-of-the-art (SOTA) recommendation models?

2. Do the designed key components benefit the representation learning of our AdaFCL in achieving performance improvement?

3. How effective is our proposed fusion strategy?

4. How do key parameters affect the model performance?

4.1 Experimental settings

Datasets. We evaluate AdaFCL on three publicly available real-world datasets: Gowalla, Amazon-book, and Tmall. The Gowalla dataset, collected from the Gowalla platform, captures user check-in behaviors at various locations. The Amazon-book dataset consists of user rating behaviors on book-category products from the Amazon platform. The Tmall dataset focuses on customer purchase behaviors observed on the Tmall online retail platform.

Following the preprocessing procedure described [31], we extract user–item interactions from the raw data and split them into training, validation, and test sets with a ratio of 7:2:1. Detailed statistics of the datasets are summarized in Table 1.

Table 1: Statistics of the experimental datasets

Dataset	#Users	#Items	#Interactions	Density
Gowalla	50,821	57,440	1,172,425	$4.0e^{-4}$
Amazon-book	78,578	77,801	2,240,156	$3.7e^{-4}$
Tmall	47,939	41,390	2,357,450	$1.2e^{-3}$

4.2 Baseline methods

To comprehensively evaluate the performance of AdaFCL, we compare it against three categories of state-of-the-art baselines, encompassing a broad range of collaborative filtering techniques.

Traditional Collaborative Filtering Methods:

1) **NCF** [30]: Utilizes multilayer perceptrons (MLPs) to model user–item interactions, demonstrating strong performance on implicit feedback recommendation tasks.

2) **AutoRec** [31]: A compact and easy-to-train autoencoder-based framework designed for collaborative filtering.

Graph Neural Network-Based Methods:

3) **NGCF** [32]: Employs a three-layer autoencoder architecture to learn and enhance user and item embeddings by reconstructing interaction signals.

4) **LightGCN** [5]: Simplifies NGCF by removing nonlinear activations and feature transformations, leveraging neighborhood information via linear propagation.

5) **DisenGCN** [16]: Improves GCN-based CF by eliminating nonlinearities and incorporating a residual architecture to extend LightGCN.

6) **MultiGCCF** [33]: Constructs multiple graphs to explicitly represent user–item, user–user, and item–item relationships.

Contrastive Learning-Based Methods:

7) **DGCF** [34]: Models intent-aware graphs via intention distribution over interactions to learn disentangled representations.

8) **DGCL** [35]: Employs a factor-disentanglement mechanism to learn decoupled node representations; inner product is used for prediction.

9) **SLRec** [38]: Treats node representations as contrastive views to enhance collaborative filtering performance.

10) **SGL-ED/ND** [7]: Constructs views of interaction structures via random walks and edge/node dropout for contrastive learning.

11) **HCCF** [11]: Relies on constructing both global and local hypergraph-based views to perform contrastive learning.

12) **NCL** [10]: A neighborhood-enhanced contrastive learning method using EM-based user clustering, followed by intra-cluster contrastive training.

13) **LightGCL** [36]: A lightweight contrastive learning framework that generates augmented views via singular value decomposition.

14) **DCCF** [29]: A disentangled contrastive CF framework that adaptively separates latent user intents and performs data augmentation.

15) **SimGCL** [9]: Discards traditional graph augmentation and introduces noise injection at each embedding layer to generate contrastive views.

16) **XSimGCL** [21]: Further simplifies contrastive learning by eliminating suboptimal graph augmentations and applying simple noise-based embedding perturbations to learn uniform user and item representations.

4.3 Evaluation

Table 2: Recommendation performance of all compared methods. R and N are abbreviations for Recall and NDCG, respectively.

Data	Gowalla				Amazon-book				Tmall			
Metrics	R@20	R@40	N@20	N@40	R@20	R@40	N@20	N@40	R@20	R@40	N@20	N@40
NCF	0.1247	0.1910	0.0659	0.0832	0.0468	0.0771	0.0336	0.0438	0.0383	0.0647	0.0252	0.0344
AutoRec	0.1409	0.2142	0.0716	0.0905	0.0546	0.0914	0.0354	0.0482	0.0336	0.0611	0.0203	0.0295
NGCF	0.1413	0.2072	0.0813	0.0987	0.0532	0.0866	0.0388	0.0501	0.0420	0.0751	0.0250	0.0365
MultiGCF	0.1458	0.2093	0.0845	0.0999	0.0554	0.0883	0.0402	0.0531	0.0442	0.0788	0.0285	0.0377
LightGCN	0.1799	0.2577	0.1053	0.1255	0.0732	0.1148	0.0544	0.0681	0.0555	0.0895	0.0381	0.0499
DisenGCN	0.1379	0.2003	0.0798	0.0961	0.0481	0.0776	0.0353	0.0451	0.0422	0.0688	0.0285	0.0377
DGCF	0.1784	0.2515	0.1069	0.1259	0.0688	0.1073	0.0513	0.0640	0.0544	0.0867	0.0372	0.0484
DGCL	0.1793	0.2483	0.1067	0.1247	0.0677	0.1057	0.0506	0.0631	0.0526	0.0845	0.0359	0.0469
SLRec	0.1529	0.2200	0.0926	0.1102	0.0544	0.0879	0.0374	0.0490	0.0549	0.0888	0.0375	0.0492
SGL-ED	0.1809	0.2559	0.1067	0.1262	0.0774	0.1204	0.0578	0.0719	0.0574	0.0919	0.0393	0.0513
SGL-ND	0.1814	0.2589	0.1065	0.1267	0.0722	0.1121	0.0542	0.0674	0.0553	0.0885	0.0379	0.0494
HCCF	0.1814	0.2589	0.1065	0.1267	0.0722	0.1121	0.0542	0.0674	0.0553	0.0885	0.0379	0.0494
NCL	0.1831	0.2624	0.1089	0.1293	0.0846	0.1318	0.0656	0.0802	0.0647	0.0986	0.0446	0.0568
LightGCL	0.1825	0.2601	0.1077	0.1280	0.0836	0.1280	0.0643	0.0790	0.0632	0.0971	0.0444	0.0562
DCCF	0.1876	0.2644	0.1123	0.1323	0.0889	0.1343	0.0680	0.0829	0.0668	0.1042	0.0469	0.0598
SimGCL	0.1927	0.2699	0.1139	0.1344	0.0907	0.1365	0.0696	0.0839	0.0680	0.1053	0.0480	0.0609
XSimGCL	0.1933	0.2709	0.1145	0.1350	0.0911	0.1368	0.0701	0.0844	0.0693	0.1072	0.0489	0.0621
AdaFCL	**0.2056**	**0.2869**	**0.1236**	**0.1448**	**0.0990**	**0.1476**	**0.0766**	**0.0922**	**0.0734**	**0.1129**	**0.0518**	**0.0656**
improve	6.36%	5.91%	7.95%	7.26%	8.67%	7.89%	9.27%	9.24%	5.92%	5.32%	5.93%	5.64%

Parameter setting. To implement our proposed model, we adopt the PyTorch framework. For parameter optimization, we use the Adam optimizer with an initial learning rate set to 0.001. Regarding general hyperparameters, we set the embedding dimension to $d = 32$, the temperature coefficient to $\tau = 0.2$, and the batch size to $B = 4096$. The hyperparameters λ_1 and λ_2 are tuned within the ranges [0.1,0.5] and [0,1.0], respectively.

We employ full ranking evaluation: for each user, we rank their positive items in the test set against all items they have not interacted with. When reproducing baseline methods, we follow the hyperparameter settings reported in their original papers, supplemented with further tuning to achieve optimal performance. For GNN-based methods, the number of layers L is selected from $\{1,2,3\}$.

We evaluate recommendation performance using the widely adopted Recall@K and NDCG@K metrics [39], where $K = \{20,40\}$. Recall@K measures the proportion of correctly predicted interactions among the ground truth, while NDCG@K further considers the ranking order of these correct predictions. All experiments are conducted on a 24GB NVIDIA 3090 GPU.

Performance Comparison (RQ1) . We present the performance comparison of recommendation methods in table 2 and draw the following conclusions: Based on a comprehensive evaluation with multiple baseline methods across three datasets, the experimental results demonstrate that AdaFCL consistently outperforms all baselines under both top-20 and top-40 settings. Compared to state-of-the-art models, our method achieves relative improvements of 6.36%, 8.67%, and 5.92% in Recall@20, and 7.95%, 9.27%, and 5.93% in NDCG@20, respectively. Specifically, compared with models employing random perturbations(e.g., SGL[7] and SimGCL[9]) , we achieve improvements of 27.91% and 9.15% in Recall@20 on the Amazon-book dataset, which we attribute this improvement to the view generator's ability to avoid introducing the noise in random perturbation methods. When compared with models that implicitly utilize high-order information(e.g., HCCF[11] and NCL[10]), AdaFCL shows improvements of 37.12% and 17.02% in Recall@20 on the Amazon-book dataset. The results indicate that explicitly leveraging high-order information via adaptive view fusion enables a more accurate capture of complex user-item relations, resulting in richer representations for both users and items.

Moreover, the results show that baseline models incorporating self-supervised learning (SSL)(e.g., LightGCL[36] and DCCF[29]), generally outperform their counterparts without SSL(e.g., NGCF[32] and LightGCN[5]). This is likely due to the inherent sparsity of labeled data in recommendation systems, where SSL helps alleviate the issue by extracting additional supervision signals from limited observed interactions. More importantly, under sparse data conditions, the application of SSL effectively reduces the risk of overfitting commonly found in user representations, especially in GNN-based models, thereby enhancing the quality and generalization of recommendation embeddings. Interestingly, although MultiGCCF[33] also explicitly models high-order relations, its performance is suboptimal. We speculate that this is because it does not employ an interest aggregation module to capture users' personalized interests in items. In addition, MultiGCCFstill applies linear transformations and activation functions within the GNN framework, which may introduce unnecessary complexity and fail to effectively enhance the propagation of collaborative signals, potentially limiting its overall performance.

4.4 Ablation experiment (RQ2)

To further investigate the impact of each module in AdaFCL on overall performance, we conducted extensive ablation studies on several variants of AdaFCL. In these experiments, we constructed different model variants by removing or replacing specific components in AdaFCL to precisely evaluate the effectiveness of each design choice. The variant w/o Specific denotes the use of only shared weights, while w/o Share refers to using only specific weights. The detailed results are presented in Figure 2, from which we observe the following:

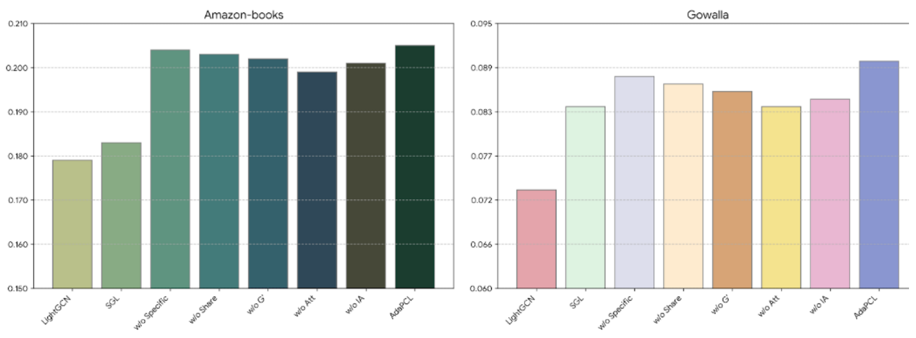

Fig. 2. Performance on datasets under in terms of Recall@20

- **w/o G':** In this setting, we remove G' from the model input, allowing the model to utilize only the original user-item interaction graph G. The experimental results validate our hypothesis: removing the high-order interaction view G' leads to performance degradation. This directly confirms that incorporating high-order interaction information plays a crucial role in capturing complex user behavior patterns, enhancing recommendation quality, and improving overall system effectiveness.

- **w/o GAT:** In this variant, we modify the view generation module by removing the graph attention network (GAT)-based view generator and replacing it with a simple GCN applied to the two input graphs. The results clearly demonstrate that the graph attention mechanism is essential for denoising view generation.

- **w/o IA:** This variant completely removes the interest aggregation module from AdaFCL. The results show that AdaFCL outperforms *w/o IA*, highlighting the benefits of the interest aggregation module in capturing and interpreting users' personalized preferences, enabling more accurate user modeling, and providing finer-grained and dynamic user interest representations for subsequent recommendation generation and interaction.

4.5 Study on Fusion Strategy (RQ3)

We conduct a systematic comparison between our proposed adaptive fusion strategy and two mainstream fusion methods: sum fusion (Sum) and attention-based fusion (Attention). The quantitative results in Table 3 demonstrate that, compared with simple summation, our method explicitly leverages the shared weights between views (W_v), effectively mitigating the risk of over-amplifying common signals caused by redundant information. This mechanism preserves fine-grained features that exist exclusively in specific views.

In contrast to the attention mechanism, which applies global weighting across entire views—potentially leading to a "winner-takes-all" effect that neglects information from certain views—our strategy operates at a finer granularity. It ensures that each view's unique contribution ($\bar{W_n}$) is retained in the final representation, thus preserving representational completeness.Consequently, our approach achieves optimal performance while maintaining model efficiency, validating its effectiveness and robustness in feature representation learning.

Table 3*:* Results from different fusion strategies only

Variants	Metric	Gowalla	Amazon-book	Tmall
Sum	Recall@20	0.2021	0.0976	0.0699
	NDCG@20	0.1217	0.0723	0.0489
Att	Recall@20	0.2013	0.0902	0.0679
	NDCG@20	0.1185	0.0701	0.0426
Ada (Ours)	Recall@20	**0.2056**	**0.0990**	**0.0734**
	NDCG@20	**0.1236**	**0.0766**	**0.0518**

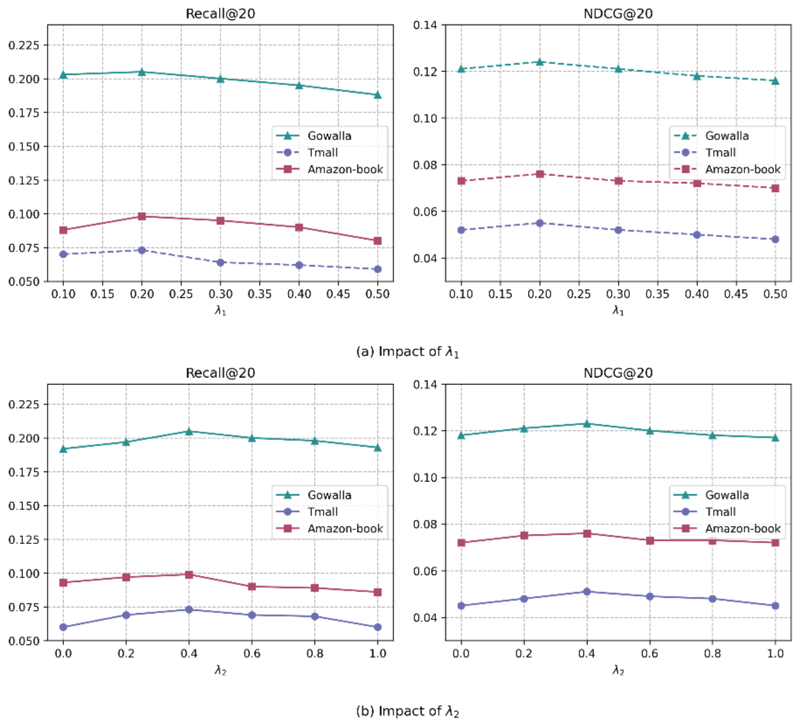

Fig. 3. Analyzing the hyper-parameters for Recall@20 and NDCG@20 on three datasets.

4.6 Hyper-parameter Analysis (RQ4)

This section aims to rigorously examine the sensitivity of the AdaFCL model to two critical hyperparameters: the loss weights λ_1 and λ_2. Specifically, the tuning effects of the contrastive learning loss weight λ_1 and the preference-aware loss weight λ_2 are illustrated in Figure 3. i) Regarding the loss weight λ_1 (for contrastive learning), we observe that the AdaFCL model achieves optimal performance when λ_1 is increased to 0.2. Further increases in λ_1 lead to a decline in performance. This highlights the importance of precisely balancing λ_1, suggesting that the auxiliary contrastive learning task should not be overly emphasized to avoid negatively impacting the primary recommendation task. ii) In addition, a similar pattern is observed for the loss weight λ_2 (for interest aggregation): the model performance peaks when λ_2 reaches 0.5, and declines with further increases. iii) We identify $\lambda_1 = 0.2$ and $\lambda_2 = 0.5$ as the optimal hyperparameter configuration. Under this setting, the model achieves effective coordination and balance among the primary recommendation task, the auxiliary contrastive learning task, and the interest aggregation task, thereby enhancing the overall optimization process in both comprehensiveness and efficacy.

5 Conclusion

We propose AdaFCL, a novel adaptive fusion graph contrastive learning model. This framework adaptively fuses views by integrating multi-graph information (user-item graph, user-user similarity graph, item-item similarity graph) to construct information-rich and less noisy contrastive views, replacing traditional random perturbation strategies. Furthermore, we innovatively design an interest aggregation module. This module fuses user interests and injects this personalized information into the final fused contrastive view. By performing contrastive learning between views, AdaFCL can learn representations that better capture the user's true interests. Extensive experimental results demonstrate that AdaFCL significantly outperforms current state-of-the-art baseline models on multiple real-world datasets, validating its effectiveness in enhancing recommendation performance and robustness.

Acknowledgments. This work was supported by the Ministry of Education Humanities and Social Sciences General project (Grant No. 23YJAZH082); the Hubei Province Education Science Planning Key Project (Grant No. 2022GA046); and the Hubei Provincial Natural Science Foundation (Grant No. 2025AFC097).

References

[1] Covington, P., Adams, J., Sargin, E.: Deep neural networks for YouTube recommendations. In: Proceedings of the 10th ACM conference on recommender systems (2016).

[2] Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. Uncertainty in Artificial Intelligence,Uncertainty in Artificial Intelligence. (2009).

[3] Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer. 30–37 (2009).
[4] Mnih, A., Salakhutdinov, R.: Probabilistic matrix factorization. Neural Information Processing Systems,Neural Information Processing Systems. (2007).
[5] He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: LightGCN: Simplifying and powering graph convolution network for recommendation.
[6] Ju, M., Shiao, W., Guo, Z., Ye, Y., Liu, Y., Shah, N., Zhao, T.: How does message passing improve collaborative filtering? arXiv preprint arXiv:2404.08660. (2024).
[7] Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., Xie, X.: Self-supervised graph learning for recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval (2021).
[8] Yao, T., Yi, X., Cheng, D.Z., Yu, F., Chen, T., Menon, A., Hong, L., Chi, E.H., Tjoa, S., Kang, J., others: Self-supervised learning for large-scale item recommendations. In: Proceedings of the 30th ACM international conference on information & knowledge management. pp. 4321–4330 (2021).
[9] Yu, J., Yin, H., Xia, X., Chen, T., Cui, L., Quoc, N., Hung, V.: Are graph augmentations necessary? Simple graph contrastive learning for recommendation.
[10] Lin, Z., Tian, C., Hou, Y., Zhao, W.: Improving graph collaborative filtering with neighborhood-enriched contrastive learning.
[11] Xia, L., Huang, C., Xu, Y., Zhao, J., Yin, D., Huang, J.: Hypergraph contrastive collaborative filtering. In: Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval (2022).
[12] Piwowarski, B., Chevalier, M., Gaussier, E.: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, (2019).
[13] Chen, L., Wu, L., Hong, R., Zhang, K., Wang, M.: Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. Proceedings of the AAAI Conference on Artificial Intelligence. 27–34 (2020).
[14] Wang, Y., Zhao, Y., Zhang, Y., Derr, T.: Collaboration-aware graph convolutional networks for recommendation systems. (2022).
[15] Zhang, D., Zhu, Y., Dong, Y., Wang, Y., Feng, W., Kharlamov, E., Tang, J.: ApeGNN: Node-wise adaptive aggregation in GNNs for recommendation.
[16] Wang, X., Jin, H., Zhang, A., He, X., Xu, T., Chua, T.-S.: Disentangled graph collaborative filtering. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval (2020).
[17] Sun, J., Cheng, Z., Zuberi, S., Perez, F., Volkovs, M.: HGCF: Hyperbolic graph convolution networks for collaborative filtering. In: Proceedings of the web conference 2021 (2021).
[18] Xia, L., Xu, Y., Huang, C., Dai, P., Bo, L.: Graph meta network for multi-behavior recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval (2021).
[19] Huang, T., Dong, Y., Ding, M., Yang, Z., Feng, W., Wang, X., Tang, J.: MixGCF. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining (2021).
[20] Song, X., Lian, J., Huang, H., Wu, M., Jin, H., Xie, X.: Friend recommendations with self-rescaling graph neural networks. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining. pp. 3909–3919 (2022).
[21] Yu, J., Xia, X., Chen, T., Cui, L., Hung, N., Yin, H.: XSimGCL: Towards extremely simple graph contrastive learning for recommendation. (2022).

[22] Jing, M., Zhu, Y., Zang, T., Yu, J., Tang, F.: Graph contrastive learning with adaptive augmentation for recommendation.
[23] Jiang, Y., Huang, C., Huang, L.: Adaptive graph contrastive learning for recommendation. In: Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining (2023).
[24] Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The adaptive web: Methods and strategies of web personalization. pp. 291–324. Springer (2007).
[25] Hamers, L., Hemeryck, Y., Herweyers, G., Janssen, M., Keters, H., Rousseau, R., Vanhoutte, A.: Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. Inf. Process. Manag. 25, 315–318 (1989).
[26] Liu, Z., Zhou, J.: Graph attention networks. In: Synthesis lectures on artificial intelligence and machine learning,introduction to graph neural networks. pp. 39–41 (2020).
[27] Taud, H., Mas, J.-F.: Multilayer perceptron (MLP). In: Geomatic approaches for modeling land change scenarios. pp. 451–455. Springer (2017).
[28] Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. Cornell University - arXiv,Cornell University - arXiv. (2018).
[29] Ren, X., Xia, L., Zhao, J., Yin, D., Huang, C.: Disentangled contrastive collaborative filtering.
[30] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.-S.: Neural collaborative filtering *. In: Proceedings of the 26th international conference on world wide web (2017).
[31] Sedhain, S., Menon, A.K., Sanner, S., Xie, L.: Autorec: Autoencoders meet collaborative filtering. In: Proceedings of the 24th international conference on world wide web. pp. 111–112 (2015).
[32] Wang, X., He, X., Wang, M., Feng, F., Chua, T.-S.: Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval (2019).
[33] Sun, J., Zhang, Y.: Multi-graph convolutional neural networks for representation learning in recommendation. In: IEEE ICDM (2019).
[34] Wang, X., Jin, H., Zhang, A., He, X., Xu, T., Chua, T.-S.: Disentangled graph collaborative filtering. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp. 1001–1010 (2020).
[35] Li, H., Wang, X., Zhang, Z., Yuan, Z., Li, H., Zhu, W.: Disentangled contrastive learning on graphs.
[36] Cai, X., Huang, C., Xia, L., Ren, X.: LightGCL: Simple yet effective graph contrastive learning for recommendation. (2023).
[37] Xia, L., Huang, C., Zhang, C.: Self-supervised hypergraph transformer for recommender systems. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining (2022).
[38] Yao T, Yi X, Cheng D Z, et al. Self-supervised learning for large-scale item recommendations[C]//Proceedings of the 30th ACM international conference on information & knowledge management. 2021: 4321-4330.