CAFU: Constrained Alignment and Filtered Uniformity for Denoising Recommendation

Xinzhe Jiang[1], Lei Sang[1], Yi Zhang[1], Kaibin Wang[2], Yiwen Zhang[1]*

[1]School of Computer Science and Technology, Anhui University
[2]Department of Computing Technologie, Swinburne University of Technology
jiangxinzhe9@gmail.com, kaibinwang@swin.edu.au, zhangyi.ahu@gmail.com, {sanglei,zhangyiwen}@ahu.edu.cn

Abstract

In recommender systems, recent advances highlight the critical role of alignment and uniformity (AU) in representation learning. Specifically, AU-based methods pull positive user-item pairs closer (alignment) and spread the overall representation distribution (uniformity), typically relying on observed positive samples. Despite their effectiveness, exist methods face two limitations: (1) noise issues have a more severe impact on AU-based methods in the absence of negative samples, leading to the capture of spurious signals such as misclicks or non-preferential behaviors; (2) data sparsity weakens the alignment of user-item representations, hindering reliable representation learning and harming recommendations for sparse users. To tackle these issues, we propose a novel recommendation framework named *Constrained Alignment and Filtered Uniformity (CAFU)*. CAFU enhances robustness through Filtered Uniformity (FU) and improves performance under data sparsity via Constrained Alignment (CA). Specifically, FU adopts a threshold-based strategy to eliminate unreliable samples that degrade embedding quality, thereby strengthening robustness. In parallel, CA mitigates the impact of sparsity by masking low-confidence user-item pairs based on angular distance, leading to better recommendation for sparse users. Extensive experiments on three datasets and three backbones demonstrate the effectiveness and generalization of the proposed framework.

Introduction

One of the most common techniques in recommender systems is collaborative filtering (CF) (Su and Khoshgoftaar 2009; Zhang et al. 2021c), which effectively captures shared patterns of user-item interactions. Traditionally CF methods rely on techniques such as matrix factorization (Koren, Bell, and Volinsky 2009; Zhang et al. 2021b) or autoencoders (Zhang et al. 2023) to learn latent representations of users and items (Xia et al. 2023). However, with the rise of graph neural networks (GNNs) (Wu et al. 2021c), there has been growing interest in leveraging these methods to propagate information along user-item interaction graphs and capture more expressive representations (Wang, Zhao, and Shi 2022; Zhang et al. 2021a). GNN-based methods such as NGCF (Wang et al. 2019) and LightGCN (He et al. 2020) have demonstrated promising results by modeling multi-layer user-item interactions through graph convolutional architectures. In particular, LightGCN simplifies conventional graph convolution by removing feature transformations and non-linearities, instead employing a straightforward weighted sum of neighboring embeddings.

Despite their effectiveness, graph-based CF methods suffer from limited supervision during training. Against this backdrop, many researchers have sought to incorporate self-supervised learning (SSL) techniques into recommender systems. To alleviate the challenge of limited supervision, recent work such as SimGCL (Yu et al. 2022) integrates contrastive learning (Zhang and Zhang 2025) into GNNs, leveraging self-supervised objectives (Wu et al. 2021a) to exploit unlabeled interactions. Graph contrastive learning (GCL) (Yu et al. 2022) enhances representation learning by enforcing consistency across augmented views, typically aligning positive pairs while pushing apart negatives to maximize mutual information, and has shown strong performance in recommendation tasks (Ju et al. 2024).

Meanwhile, some studies have shifted focus to the quality of learned embeddings, identifying two key properties of contrastive loss: alignment and uniformity (Wang and Isola 2020). This insight led to the development of DirectAU (Wang et al. 2022a), which directly optimizes a joint loss to improve both alignment and uniformity. Specifically, DirectAU employs alignment loss to bring observed user-item embeddings closer, capturing semantic similarity. In parallel, it utilizes uniformity loss to encourage user and item embeddings to be evenly distributed on their respective hyperspheres, thereby maintaining a well-structured representation space. By relying on positive interactions, DirectAU avoids the need for negative sampling and auxiliary tasks, offering an elegant and self-contained learning paradigm.

Despite offering a streamlined and effective learning framework, AU-based methods face notable limitations in practical recommendation. More concretely, its reliance on original implicit feedback makes it vulnerable to two main challenges. **Challenge 1: Denoising in the absence of negative samples.** Implicit feedback consists not only of true positives, but also false positives, false negatives, and true negatives(Wang et al. 2021). Existing denoising methods often reweight both positive and negative samples to mitigate noise (Zhao et al. 2024; Yang et al. 2023). For example,

*Corresponding author

	ML-100K		Modcloth	
	R@20	N@20	R@20	N@20
GMF	0.0950	0.0601	0.1277	0.0537
+DeCA	**0.1075**	**0.0631**	0.1528	0.0683
+DeCA(p)	0.1017	0.0599	**0.1551**	**0.0687**
Improvement	+13.2%	+5.0%	+21.5%	+27.9%
LightGCN	0.1100	0.0650	0.1221	0.0515
+DeCA	0.1040	0.0640	0.1289	0.0552
+DeCA(p)	**0.1180**	**0.0750**	**0.1461**	**0.0651**
Improvement	+7.3%	+15.4%	+19.6%	+26.4%
DirectAU	**0.1209**	**0.0768**	**0.1837**	**0.0747**
+DeCA	0.1129	0.0749	0.1506	0.0628
+DeCA(p)	0.1115	0.0727	0.1709	0.0694
Improvement	–6.6%	–2.5%	–6.9%	–7.1%

Table 1: Denoising study on different datasets (measured by Recall@20 and NDCG@20). Bold face denotes the best performance among the backend models. DeCA(p) means the pre-training DeCA.

Wang et al. (Wang et al. 2022b) introduced uncertainty estimation into the training process, significantly improving robustness in models like GMF (He et al. 2017) and LightGCN. However, their reliance on negative samples makes them incompatible with AU-based methods, which learn solely from positives—resulting in limited or even negative gains on DirectAU, as shown in Table 1. This highlights the need for a tailored denoising approach that operates entirely within the scope of positive samples.

Challenge 2: Effective learning from sparse user interactions. Data sparsity is a persistent challenge in recommender systems, where a large portion of users have only a limited number of recorded interactions. For AU-based methods like DirectAU, which rely exclusively on positive feedback, such sparsity reduces the availability of informative training representations. As a result, the method may struggle to optimize alignment and uniformity objectives effectively, particularly for users with few interactions. This imbalance can cause sparse users to be underrepresented in the embedding space. As a result, the method tends to favor popular users and items, which reduces recommendation diversity and personalization. Therefore, alleviating the impact of data sparsity is essential for improving the effectiveness of AU-based methods.

- How to mitigate the impact of noise when AU-based methods rely solely on positive items?
- How to effectively model users with sparse interactions while maintaining model performance?

To address the above issues, we propose the **Constrained Alignment and Filtered Uniformity (CAFU)** framework. Facing the first issue, we retain the characteristic of AU-based methods, using only positive interactions for alignment and uniformity. Specifically, we filter positive samples to reduce noise from user misoperations, minimizing its impact on model training. Furthermore, we filter out users with anomalous behavior or overly broad interests to construct a more representative user embedding distribution. Facing the second issue, we constrain the alignment loss by selectively masking unreliable user-item pairs, thereby preserving only the most semantically consistent interactions. This reduces the influence of low-quality data and guides the model toward more trustworthy signals, while implicitly amplifying the influence of sparse users' informative signals. As a result, sparse users receive higher-quality recommendations despite limited interaction. Overall, CAFU is a simplified yet robust framework that preserves the core advantages of alignment and uniformity, while improving robustness through targeted filtering and alignment constraints. The contributions of our paper are summarized as follows:

- We propose the ***CAFU*** framework, which introduces a filtered uniformity strategy to explicitly suppress noise. This denoising design enhances the methods' robustness while retaining the AU-based characteristic of using only positive samples.
- We design a constrained alignment mechanism that focuses on trustworthy user-item pairs, thereby enhancing performance for users with sparse interactions.
- Extensive experiments on three public datasets and three backbones demonstrate that CAFU effectively mitigates the impact of sparse interactions and noise, leading to improved recommendation performance.

Preliminaries

Alignment and Uniformity

Wang et al. (Wang and Isola 2020) identified two essential properties underlying contrastive learning: the alignment of positive samples and the uniformity of feature distributions. Building upon this, DirectAU (Wang et al. 2022a) introduces these principles into recommendation systems, aiming to jointly optimize user and item embeddings. The alignment loss is calculated as:

$$l_{\text{align}} \triangleq \mathop{\mathbb{E}}_{(x,x^+)\sim p_{\text{pos}}} \left\| f(x) - f(x^+) \right\|^{\alpha}, \tag{1}$$

where $(x, x^+) \sim p_{\text{pos}}$ denotes a positive sample pair, and $f(\cdot)$ represents the l_2-normalized embedding function. Here, x^+ is a positive counterpart of x, sharing semantic similarity. The α controls the sensitivity of the loss to the magnitude of the embedding distances. This alignment loss encourages embeddings of associated users and items to be close in the latent space. The uniformity loss is expressed as:

$$l_{\text{uniform}} \triangleq \log \mathop{\mathbb{E}}_{x,y\sim p_{\text{data}}} e^{-t\|f(x)-f(y)\|^2}, \tag{2}$$

where $x, y \sim p_{data}$ represents randomly sampled instances from the data distribution and t is a parameter that controls the scale of the distance. This term promotes a uniform spread of embeddings across the hypersphere, preventing representational collapse and encouraging diversity (Yang et al. 2023). The overall loss function is a weighted sum:

$$l_{\text{DirectAU}} = l_{\text{align}} + \gamma l_{\text{uniform}}, \tag{3}$$

where γ serves as a balancing hyperparameter that adjusts the emphasis on uniformity.

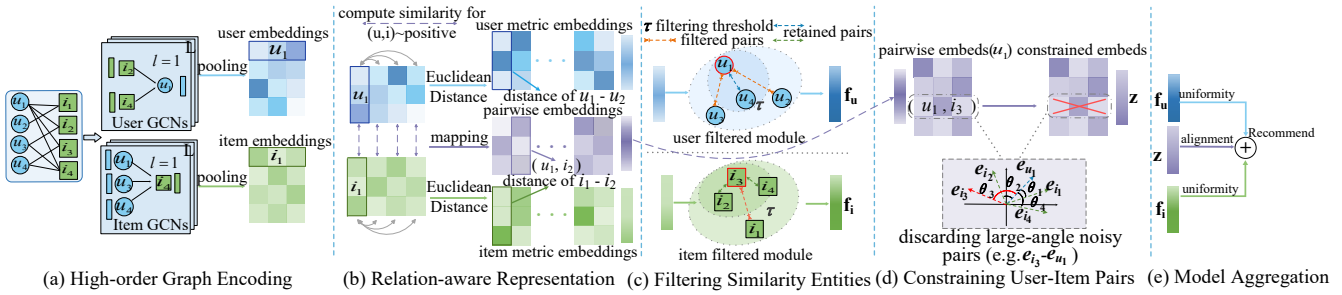


Figure 1: The complete framework of the proposed CAFU. In this framework: **(a)–(b)** constitute the embedding and distance modeling stage, where user and item embeddings are learned and pairwise relations are computed; **(c) Filtering Similarity Entities** selects user–user and item–item pairs with similarity below threshold τ for filtered uniformity loss; **(d) Constraining User-Item Pairs** extracts informative user-item interactions for alignment loss (e.g., u_1 retains only reliable items).

DirectAU demonstrates strong empirical performance by replacing traditional objectives with optimization based on alignment and uniformity. Nevertheless, its reliance on positive-only interactions may lead to suboptimal matching under sparse data and reinforce noisy signals, limiting the expressiveness and robustness of learned representations.

Methodology

In the section, we introduce the CAFU framework, which mainly includes high-order graph encoding, constrained alignment and filtered uniformity. The overall architecture of CAFU is illustrated in Figure 1.

High-order Graph Encoding

Numerous studies have shown that graph convolutional networks (GCNs) (Kipf and Welling 2017) effectively capture collaborative relations in user-item graph $\mathcal{G}$ (Wu et al. 2021b). We adopt LightGCN (He et al. 2020) to generate user and item embeddings:

$$\begin{aligned}\mathbf{e}_u^{(k+1)} &= \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\mathbf{e}_u^{(k)},\\ \mathbf{e}_i^{(k+1)} &= \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\mathbf{e}_u^{(k)},\end{aligned} \tag{4}$$

where $\mathcal{N}_u$ and $\mathcal{N}_i$ denote the items interacted by user u and the users interacted by item i, respectively. The representations $\mathbf{e}_u^{(k)}$ and $\mathbf{e}_i^{(k)}$ refer to the user and item embeddings obtained after k layers of propagation. The symmetric normalization factor $\frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}$ prevents uncontrolled growth of embedding magnitude and improves learning efficiency, enabling effective multi-hop relational encoding.

After K layers of propagation, combining embeddings from all layers to obtain the final representation:

$$\mathbf{e}_u = \sum_{k=0}^{K}\omega_k\mathbf{e}_u^{(k)}, \quad \mathbf{e}_i = \sum_{k=0}^{K}\omega_k\mathbf{e}_i^{(k)}, \tag{5}$$

where α_k is the weight for the k-th layer, typically set uniformly as $\omega_k = \frac{1}{K+1}$ to preserve simplicity and generality.

Constrained Alignment

The alignment loss aims to pulls positive user–item embeddings closer to encourage consistent representations. However, a key limitation of the standard alignment loss is its equal treatment of all positive interactions, regardless of their quality. In practice, some positives may be noisy or uninformative, resulting in suboptimal gradient updates.

To mitigate this issue, we introduce a mechanism that emphasizes informative or reliable positive samples in the alignment process. We hypothesize that removing abnormal interactions reduces embedding distortion and improves optimization. To support this, we analyze the gradient of the alignment loss with respect to users.

$$\begin{aligned}\frac{\partial l_{\text{align}}}{\partial \mathbf{e}_u} &= \frac{\partial}{\partial \mathbf{e}_u}\mathop{\mathbb{E}}_{(u,i)\sim p_{\text{pos}}}\|\mathbf{e}_u-\mathbf{e}_i\|^{\alpha},\\ &= \mathop{\mathbb{E}}_{(u,i)\sim p_{\text{pos}}}\alpha\|\mathbf{e}_u-\mathbf{e}_i\|^{\alpha-2}(\mathbf{e}_u-\mathbf{e}_i),\\ &= \mathop{\mathbb{E}}_{(u,i)\sim p_{\text{pos}}}\alpha\left(\sqrt{2-2\cos\theta_{ui}}\right)^{\alpha-2}(\mathbf{e}_u-\mathbf{e}_i),\\ &= \mathop{\mathbb{E}}_{(u,i)\sim p_{\text{pos}}}\alpha\left(2\sin(\theta_{ui}/2)\right)^{\alpha-2}(\mathbf{e}_u-\mathbf{e}_i),\end{aligned} \tag{6}$$

where $\|\cdot\|$ denotes the Euclidean norm, and $(u,i)\sim p_{\text{pos}}$ represents the distribution over positive user–item pairs. $\mathbf{e}_u$ and $\mathbf{e}_i$ denote the embeddings of user u and item i, respectively, lying on the unit hypersphere. The angle θ_{ui} between these unit vectors directly determines the Euclidean distance $\|\mathbf{e}_u-\mathbf{e}_i\| = 2\sin(\theta_{ui}/2)$, which in turn controls the magnitude of the gradient. To further interpret the update direction, we expand the difference vector:

$$\begin{aligned}\mathbf{e}_u-\mathbf{e}_i &= \mathbf{e}_u-\left(\cos\theta_{ui}\mathbf{e}_u+\sin\theta_{ui}\mathbf{e}_u^{\perp}\right)\\ &= (1-\cos\theta_{ui})\mathbf{e}_u-\sin\theta_{ui}\mathbf{e}_u^{\perp},\end{aligned} \tag{7}$$

where $\mathbf{e}_u^{\perp}$ is the unit vector orthogonal to $\mathbf{e}_u$ toward $\mathbf{e}_i$. The difference vector lies in the tangent space at $\mathbf{e}_u$, pointing to $\mathbf{e}_i$. It has radial term $(1-\cos\theta_{ui})\mathbf{e}_u$ causing contraction and angular term $-\sin\theta_{ui}\mathbf{e}_u^{\perp}$ causing rotation. Together, they define the direction and magnitude of the alignment update.

Consequently, positive pairs with larger angular distances produce stronger gradients that push alignment for similar

embeddings. However, this may also allow noisy or unreliable pairs with large discrepancies to dominate optimization, potentially harming model robustness.

To address this, we apply a top-N constraint that emphasizes retaining closer, more consistent pairs, reducing the effect of unreliable interactions. By restricting gradient computation to the top-N positive samples, this constraint effectively filters out unreliable interactions. Following prior work, we set $\alpha = 2$. The modified gradient is:

$$\frac{\partial l_{\text{align}}^{\text{constrained}}}{\partial \mathbf{e}_u} = \sum_{(u,i)\in S_{\text{top-N}}} \alpha(2\sin(\theta_{ui}/2))^{\alpha-2}(\mathbf{e}_u - \mathbf{e}_i), \quad (8)$$

$$= 2 \sum_{(u,i)\in S_{\text{top-N}}} ((1-\cos\theta_{ui})\mathbf{e}_u - \sin\theta_{ui}\mathbf{e}_u^{\perp}). \quad (9)$$

The top-N constraint is applied by retaining only the N closest positive interactions, based on their angular distance on the unit hypersphere. Formally, the top-N set is defined as:

$$S_{\text{top-N}} = \left\{(u,i) \mid \theta_{ui} \in \text{Top-N}\left(\{\theta_{ui}\}_{(u,i)\in S_{\text{pos}}}\right)\right\}, \quad (10)$$

where θ_{ui} is the angle between user u and item i, and S_{pos} denotes the set of positive pairs. The top-N constraint restricts gradient updates to the N closest pairs, reducing the impact of distant or noisy interactions.

In the original gradient formulation (Eq. (6)), large angular distances θ_{ui} result in large gradients, which may overemphasize unreliable pairs and destabilize training. By focusing only on the most relevant pairs, the top-N strategy highlights informative interactions and improves optimization stability. This is particularly beneficial for sparse users, as limited interactions make them more susceptible to noise. Limiting alignment to their most reliable signals avoids overfitting and ensures more stable gradient flow, which is essential for learning from limited data.

Based on the above strategy, we formulate the constrained alignment loss to quantitatively enforce semantic consistency over the selected user–item pairs:

$$l_{\text{align}}^{\text{constrained}} = \sum_{(u,i)\in S_{\text{top-N}}} \frac{1}{N}\|\hat{\mathbf{e}}_u - \hat{\mathbf{e}}_i\|^2, \quad (11)$$

where $\hat{\mathbf{e}}_u$ and $\hat{\mathbf{e}}_i$ denote the normalized embeddings of user and item, respectively, projected onto the unit hypersphere. The normalization factor $\frac{1}{N}$ ensures equal contribution across the selected pairs. (Garbin, Zhu, and Marques 2020). Embedding normalization, combined with this constrained loss, helps mitigate abnormal gradients and suppress noisy interference during training. By prioritizing semantically consistent interactions, it amplifies the influence of informative signals from sparse users, enabling the model to better capture their limited yet valuable preferences. As a result, it achieves more stable convergence, improved robustness, and enhanced recommendation quality for underrepresented users, while also reducing the risk of overfitting.

Filtered Uniformity

The uniformity loss typically computes a weighted average of distances between all pairs of samples, encouraging the learned representations to be uniformly distributed on the hypersphere (Wang et al. 2023a). However, this assumes all sample pairs are equally informative—an assumption that often fails in real-world recommendation scenarios due to noisy interactions and extreme sparsity.

We argue that in the presence of many noisy samples, cumulative effects from weakly informative pairs can distort gradient estimation, impeding both convergence and structural representation. Furthermore, interaction sparsity in recommendation settings renders average pairwise distances an unreliable approximation of the true distribution, diminishing the utility of standard uniformity loss.

To address this, we propose a filtered uniformity strategy. As shown in Figure 1, we first obtain embeddings via LightGCN and compute user-user and item-item pairwise Euclidean distances. Only pairs with distances below a threshold τ are retained for loss computation. This filtering reduces the impact of noisy or irrelevant pairs and improves the stability of representation learning.

Formally, the filtered uniformity loss is defined as:

$$l_{\text{uniform}}^{\text{filtered}} = \log \mathop{\mathbb{E}}_{x,y\sim p_{\text{data}}} e^{-2\|\mathbf{e}_x - \mathbf{e}_y\|^2} \cdot \mathbb{I}(\|\mathbf{e}_x - \mathbf{e}_y\| \le \tau), \quad (12)$$

where τ is a predefined threshold, and $\mathbb{I}(\|\mathbf{e}_x - \mathbf{e}_y\| \le \tau)$ is an indicator function that filters out distant (potentially noisy) sample pairs. This modified formulation retains the core objective of the original uniformity loss—encouraging representations to spread uniformly over the hypersphere. By incorporating a threshold τ and an indicator function, it selectively includes only those sample pairs whose distances fall within a reasonable range, thereby mitigating the influence of noisy or uninformative interactions.

The rationale behind this filtering strategy is twofold: (1) closer pairs are more likely to encode meaningful structures, while distant pairs often stem from noisy behavior or sparse interaction patterns; (2) excluding unreliable pairs improves gradient estimation by reducing variance, ultimately leading to more efficient and stable optimization.

To further understand the effect of this filtering mechanism, we analyze its influence on the optimization dynamics. Compared to the original uniformity loss, which considers all sample pairs, the filtered version focuses only on nearby embeddings within the distance threshold. This selective focus helps eliminate uninformative or noisy pairs that often introduce high-variance or misleading gradients. Filtering them out reduces gradient variance and yields a more stable optimization direction aligned with the true objective, facilitating more efficient training and faithful structural representations.

In practice, we compute it separately for user and item embeddings, as their distribution characteristics may differ. Let p_{user} and p_{item} denote the sets of users and items, respectively. The overall filtered uniformity loss is defined as:

$$\begin{aligned} l_{\text{uniform}}^{\text{filtered}} = &\log \mathop{\mathbb{E}}_{u,u'\sim p_{\text{user}}} \left[e^{-2\left\|\hat{\mathbf{e}}_u - \hat{\mathbf{e}}'_u\right\|^2} \cdot \mathbb{I}(\|\hat{\mathbf{e}}_u - \hat{\mathbf{e}}'_u\| \le \tau)\right] \\ &+ \log \mathop{\mathbb{E}}_{i,i'\sim p_{\text{item}}} \left[e^{-2\left\|\hat{\mathbf{e}}_i - \hat{\mathbf{e}}'_i\right\|^2} \cdot \mathbb{I}(\|\hat{\mathbf{e}}_i - \hat{\mathbf{e}}'_i\| \le \tau)\right], \end{aligned} \quad (13)$$

	Amazon-Book				Tmall				Yelp2018			
	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20
LightGCN	0.0238	0.0403	0.0250	0.0314	0.0438	0.0718	0.0383	0.0499	0.0373	0.0635	0.0426	0.0521
SimGCL	0.0313	0.0517	0.0334	0.0412	0.0563	0.0883	0.0502	0.0632	0.0426	0.0721	0.0490	0.0596
DiffRec	0.0311	0.0514	0.0341	0.0418	0.0505	0.0792	0.0487	0.0612	0.0393	0.0665	0.0457	0.0556
BIGCF	0.0294	0.0500	0.0320	0.0398	0.0547	0.0876	0.0524	0.0664	0.0431	0.0729	0.0497	0.0602
SIURec	0.0242	0.0407	0.0259	0.0323	0.0448	0.0720	0.0389	0.0502	0.0371	0.0641	0.0407	0.0513
+T-CE	0.0218	0.0376	0.0226	0.0290	0.0417	0.0696	0.0344	0.0461	0.0368	0.0637	0.0406	0.0509
+R-CE	0.0244	0.0410	0.0263	0.0326	0.0451	0.0726	0.0398	0.0512	0.0385	0.0652	0.0434	0.0535
+DeCA	0.0237	0.0399	0.0251	0.0315	0.0450	0.0725	0.0395	0.0508	0.0381	0.0653	0.0428	0.0530
+DCF	0.0244	0.0411	0.0258	0.0324	0.0451	0.0721	0.0398	0.0509	0.0374	0.0636	0.0426	0.0523
+CAFU	**0.0259***	**0.0431***	**0.0279***	**0.0346***	**0.0475***	**0.0759***	**0.0424***	**0.0541***	**0.0390***	**0.0661***	**0.0441***	**0.0542***
MAWU	0.0315	0.0526	0.0345	0.0424	0.0541	0.0851	0.0485	0.0612	0.0404	0.0689	0.0466	0.0568
+T-CE	0.0317	0.0530	0.0346	0.0426	0.0544	0.0855	0.0487	0.0614	0.0405	0.0691	0.0467	0.0570
+R-CE	0.0320	0.0533	0.0348	0.0428	0.0541	0.0854	0.0489	0.0615	0.0402	0.0684	0.0463	0.0564
+DeCA	0.0313	0.0518	0.0340	0.0418	0.0537	0.0843	0.0483	0.0608	0.0398	0.0678	0.0459	0.0559
+DCF	0.0323	0.0540	0.0350	0.0433	0.0547	0.0862	0.0491	0.0618	0.0410	0.0693	0.0473	0.0573
+CAFU	**0.0346***	**0.0568***	**0.0374***	**0.0458***	**0.0582***	**0.0907***	**0.0521***	**0.0653***	**0.0419***	**0.0711***	**0.0482***	**0.0586***
DirectAU	0.0301	0.0508	0.0322	0.0406	0.0526	0.0835	0.0468	0.0593	0.0420	0.0709	0.0481	0.0586
+T-CE	0.0305	0.0510	0.0333	0.0408	0.0540	0.0855	0.0483	0.0611	0.0423	0.0715	0.0490	0.0595
+R-CE	0.0307	0.0512	0.0333	0.0412	0.0540	0.0856	0.0483	0.0612	0.0420	0.0712	0.0485	0.0591
+DeCA	0.0308	0.0518	0.0331	0.0411	0.0535	0.0851	0.0479	0.0607	0.0419	0.0711	0.0486	0.0592
+DCF	0.0318	0.0529	0.0347	0.0427	0.0533	0.0850	0.0476	0.0604	0.0422	0.0716	0.0485	0.0588
+CAFU	**0.0345***	**0.0565***	**0.0373***	**0.0457***	**0.0590***	**0.0923***	**0.0530***	**0.0663***	**0.0446***	**0.0746***	**0.0512***	**0.0619***

Table 2: Recommendation performance on three datasets. Best results are bolded for backbone models, denoising baselines, and CAFU. * indicates significant improvement ($p < 0.001$) via two-tailed paired $t-test$.

where $\mathbb{I}(\|\hat{\mathbf{e}}_u - \hat{\mathbf{e}}'_u\| \leq \tau)$ is an indicator function that filters out user embedding pairs with distances exceeding a predefined threshold. This formulation enables the model to capture the structural patterns of both user and item spaces, improving robustness to noise in both domains.

Model Training

Building on the proposed constrained alignment and filtered uniformity losses, we integrate them into a unified training framework. The overall objective is defined as:

$$\mathcal{L}_{\text{CAFU}} = l_{\text{align}}^{\text{constrained}} + \lambda l_{\text{uniform}}^{\text{filtered}} \tag{14}$$

where λ is a hyperparameter that controls the balance between alignment and uniformity.

To reduce computational cost and improve stability, constraint and filtering are applied every five epochs instead of every iteration. This periodic strategy mitigates early-stage sensitivity to noisy interactions and allows the model to first capture global structural patterns. The design is inspired by the frequency principle in deep learning, which suggests that neural networks tend to learn low-frequency structures before high-frequency details (Luo et al. 2021); applying constraints too early may interfere with this natural progression.

Model Analysis

Space Complexity CAFU introduces no additional memory overhead. Both the constrained alignment and filtered uniformity losses operate directly on existing user and item embeddings, requiring no extra storage beyond standard CF models like LightGCN.

Time Complexity The time complexity of CAFU covers graph convolution and loss computation. Given $|E|$ edges, embedding size d, L layers, batch size B, and M nodes per batch, graph convolution requires $O(L|E|d)$ using a single encoder and a normalized adjacency matrix with $2|E|$ non-zeros. The constrained alignment and filtered uniformity losses incur $O(Bd + Md)$ and $O(M^2 d)$, respectively. Overall, the total complexity is $O(2|E| + L|E|d + Bd + Md + M^2 d)$, matching that of DirectAU.

Experiments

In this section, we evaluate the performance of our proposed CAFU framework compared with state-of-the-art recommendation methods using three real-world datasets and three backbones.

Experimental Settings

Datasets. To evaluate CAFU's recommendation performance, we conduct experiments on three public datasets: **Yelp2018** (Yu et al. 2022), **Amazon-Book** (Zhang, Sang, and Zhang 2024), and **Tmall** (Ren et al. 2023). Table 3 summarizes their statistics. For fairness, we follow the preprocessing steps of prior work (Zhang, Sang, and Zhang 2024). We adopt the full-ranking protocol and report Recall@R and NDCG@R with $R = 10, 20$ (Wang et al. 2019).

Baselines. To verify the effectiveness of CAFU as a denoising module in AU-based methods, we compare CAFU other state-of-the-art denoising methods: **T-CE** (Wang et al.

Datasets	Users	Items	Interactions	Density
Yelp2018	31,668	38,084	1,561,406	0.00130
Amazon-book	52,643	91,599	2,984,108	0.00062
Tmall	47,939	41,390	2,619,389	0.00132

Table 3: Statistics of the experimental datasets.

2021), **R-CE** (Wang et al. 2021), **DeCA** (Wang et al. 2022b) and **DCF** (He et al. 2024). We implement CAFU and the aforementioned denoising baselines to three representative AU-based backend models: **SIURec** (Ma, Lian, and Song 2025), **MAWU** (Yang et al. 2023) and **DirectAU** (Wang et al. 2022a). We also compare CAFU with interaction-based and generative denoising baselines: **LightGCN** (He et al. 2020), **SimGCL** (Yu et al. 2022), **DiffRec** (Wang et al. 2023b), **BIGCF** (Zhang, Sang, and Zhang 2024).

Parameter Settings All experiments are implemented on a GeForce RTX 3090 GPU.For fair comparison, we follow setups from prior works. Adam (Kingma and Ba 2017) is used as the optimizer with Xavier initialization (Glorot and Bengio 2010). Batch size is 2048, and embedding size is 64. The default encoder is LightGCN. For CAFU, we tune weight γ over [0.5, 1.5, 3, 4.5, 6.5, 8.5, 10], constrained weight $top{-}N$ over [0, 0.2, 0.4, 0.6, 0.8], and filtered weight τ over [1.0, 1.2, 1.4, 1.6, 1.8]. Baseline hyperparameters are tuned within ranges from their original papers.

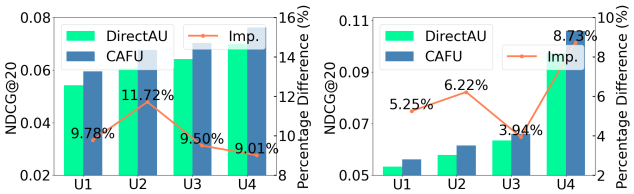
Overall Performance

We evaluate our framework on three AU-based backbone methods across three benchmark datasets, as summarized in Table 2. To ensure robustness, we compare CAFU against severalbaselines, retraining each model five times and reporting corresponding p-values for statistical significance. Based on the results, we draw the following observations:

- CAFU consistently achieves strong performance across all backbones and datasets by capturing the core traits of AU-based methods. It constrains alignment to mitigate noise bias and filters user-user and item-item pairs in the uniformity loss, preserving meaningful structure. These gains are statistically significant under a two-tailed paired $t{-}test$.
- T-CE and DeCA do not consistently outperform their backbone models. Two potential reasons emerge: (1) In T-CE, the threshold gradually increases from zero to an upper bound, which may conflict with the one-shot weighting strategy adopted by SIURec in the first epoch. (2) DeCA jointly optimizes four models, introducing instability and causing performance fluctuations.
- Traditional denoising methods rely heavily on explicitly available negative samples, which are typically unavailable in AU-based methods. For example, DeCA employs distinct denoising strategies for positive and negative pairs, making it incompatible with AU frameworks. As a result, these methods have limited applicability and effectiveness within AU-based settings.

	Amazon-Book		Tmall		Yelp2018	
	R@20	N@20	R@20	N@20	R@20	N@20
DirectAU	0.0508	0.0406	0.0835	0.0593	0.0709	0.0586
CAFU-w/o C	0.0553	0.0445	0.0894	0.0639	0.0737	0.0612
CAFU-w/o F	0.0550	0.0439	0.0905	0.0653	0.0735	0.0609
CAFU	**0.0565**	**0.0457**	**0.0923**	**0.0663**	**0.0746**	**0.0619**

Table 4: Ablation study of CAFU on different datasets.



(a) Tmall Dataset (b) Yelp2018 Dataset

Figure 2: Performance w.r.t interaction sparsity for different user groups, grouped by interaction counts.

Ablation Study

To evaluate the effectiveness of each component, we conduct an ablation study on three datasets by comparing CAFU with its variants, as shown in Table 4. CAFU-w/o C removes the constraint on alignment loss, using standard alignment instead, while CAFU-w/o F replaces the filtered uniformity loss with the original version. Removing the constraint leads to consistent performance drops, highlighting the importance of controlling alignment to avoid distortion from long-distance interactions. Similarly, removing the filtered uniformity module degrades performance, confirming its role in suppressing irrelevant similarities and refining the embedding space. Overall, CAFU consistently outperforms all variants, demonstrating the effectiveness of integrating constrained alignment and filtered uniformity.

Sparse Data Experiment Analysis

To examine the effect of data sparsity, we evaluate performance across user groups with varying interaction counts. As shown in Figure 2, users with more interactions achieve higher accuracy, as expected. CAFU consistently outperforms all baselines across groups. While performance drops for the sparsest users, CAFU maintains strong accuracy, highlighting its robustness.

Overall, CAFU achieves excellent results in user groups with relatively sparse interactions. Notably, on the Tmall dataset, CAFU outperforms DirectAU by 9.78%, 11.72%, and 9.50% in the first three groups, showing substantial gains. Similarly, on Yelp2018, the sparsest group sees significant improvement. We attribute this to the proposed constraints and filtering, which reduce noise and help the model focus on informative patterns—especially crucial for sparse users prone to noise interference.

Data Noise Degree Experiment Analysis

To evaluate the impact of noise on model performance, we perturb the data by modifying interaction edges. Specifi-

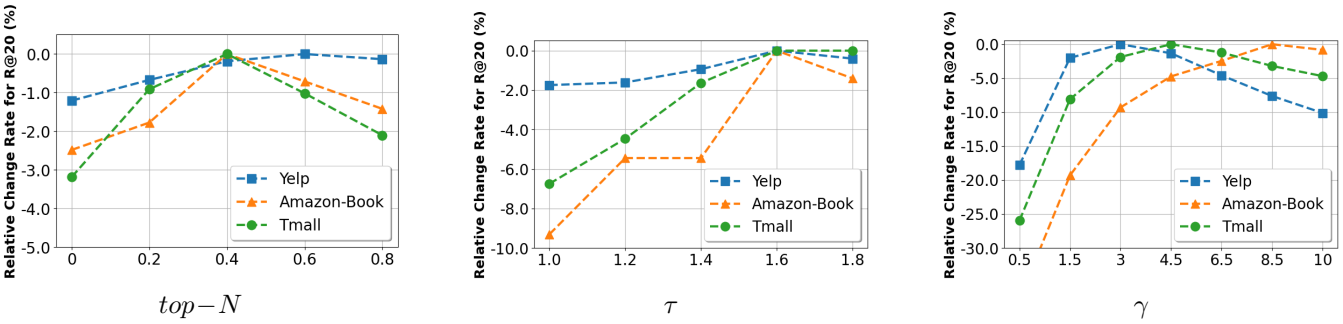


Figure 3: Hyperparameter sensitivities w.r.t Recall: constrained weight $top-N$, filtered weight τ, and uniformity weight γ.

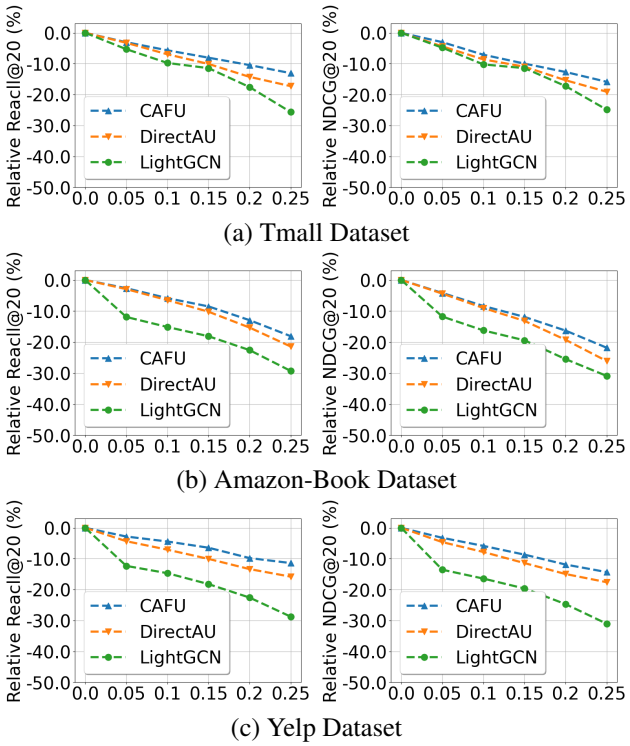


Figure 4: Performance degradation w.r.t. noise ratio. We simulate noise by replacing 5%, 10%, 15%, 20%, and 25% of interaction edges with fake ones.

cally, 5%, 10%, 15%, 20%, and 25% of edges are replaced with fake ones, and the corrupted graph is used for training. We compare CAFU with DirectAU and LightGCN. DirectAU leverages alignment and uniformity in training, while LightGCN is a typical GCN model. As shown in Figure 4, we assess robustness by tracking relative performance drops under noise. CAFU shows smaller degradation than the baselines, indicating stronger robustness.

We attribute this to two reasons: First, CAFU uses alignment and uniformity loss in SSL to capture graph structure information. This approach, like DirectAU, demonstrates stronger robustness compared to LightGCN. Second, CAFU applies constraints on alignment information and filters out irrelevant or noisy pairs. This targeted approach helps mitigate the negative impact of noise on recommendation.

Notably, among all datasets, Amazon-Book shows the largest performance drop under noise. When 25% of edges are replaced, all three models decline by over 20% . This may be due to the fact that Amazon-Book is the sparsest dataset, making it more susceptible to noise interference. Overall, our results suggest that CAFU remains robust and effective for recommendation, even with noisy data.

Parameter Sensitivity Analysis

Effect of Constrained Weights $top-N$**.** We vary N from 20% to 80% to find the optimal constraint level. The best results appear at 40% for Amazon-Book and Tmall, and 60% for Yelp2018. All constrained variants outperform the unconstrained model, confirming that moderate constraints reduce noise and improve alignment.

Effect of Filtered Weights τ**.** Excessive filtering may impair uniformity on the hypersphere, leading to misalignment and reduced discriminability. We performed a grid search for τ in the range of 1 to 2, with the following values [1.0, 1.2, 1.4, 1.6, 1.8]. Figure 3 shows that 1.6 yields the best results across all datasets. Lower τ reduce performance on Amazon-Book and Tmall, due to their higher interaction counts, where too strict filtering confuses sample categories.

Effect of Uniformity Loss Weight γ**.** As shown in Figure 3, low γ leads to excessive alignment, while high γ causes over-separation, both harming performance. On Yelp and Tmall, the optimal γ increases with dataset size, reflecting the need for stronger consistency. For Amazon-Book, sparse interactions require higher uniformity.

Conclusion

This paper revisited the limitations of directly optimizing alignment and uniformity in recommendation. To address these issues, we proposed a novel framework (CAFU). CAFU mitigates noise by filtering user–user and positive item–item pairs, improving the embedding distribution over the hypersphere. Additionally, a positive interaction constraint is introduced to enhance alignment in sparse regions. Extensive experiments on three real-world datasets demonstrate that our denoising strategy significantly improves both effectiveness and generalization in recommendation tasks.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62272001).

References

Garbin, C.; Zhu, X.; and Marques, O. 2020. Dropout vs. batch normalization: an empirical study of their impact to deep learning. *Multim. Tools Appl.*, 79(19-20): 12777–12815.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W.; and Titterington, D. M., eds., *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, volume 9 of *JMLR Proceedings*, 249–256. JMLR.org.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, 639–648. ACM.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, 173–182. ACM.

He, Z.; Wang, Y.; Yang, Y.; Sun, P.; Wu, L.; Bai, H.; Gong, J.; Hong, R.; and Zhang, M. 2024. Double Correction Framework for Denoising Recommendation. In Baeza-Yates, R.; and Bonchi, F., eds., *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, 1062–1072. ACM.

Ju, W.; Wang, Y.; Qin, Y.; Mao, Z.; Xiao, Z.; Luo, J.; Yang, J.; Gu, Y.; Wang, D.; Long, Q.; Yi, S.; Luo, X.; and Zhang, M. 2024. Towards Graph Contrastive Learning: A Survey and Beyond. *CoRR*, abs/2405.11868.

Kingma, D. P.; and Ba, J. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Koren, Y.; Bell, R. M.; and Volinsky, C. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8): 30–37.

Luo, T.; Ma, Z.; Xu, Z.-Q. J.; and Zhang, Y. 2021. Theory of the Frequency Principle for General Deep Neural Networks. *CSIAM Transactions on Applied Mathematics*, 2(4): 771–794.

Ma, R.; Lian, Y.; and Song, C. 2025. Sub-Interest-Aware Representation Uniformity for Recommender System. In Walsh, T.; Shah, J.; and Kolter, Z., eds., *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, 12346–12354. AAAI Press.

Ren, X.; Xia, L.; Zhao, J.; Yin, D.; and Huang, C. 2023. Disentangled Contrastive Collaborative Filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, 1137–1146. ACM.

Su, X.; and Khoshgoftaar, T. M. 2009. A Survey of Collaborative Filtering Techniques. *Adv. Artif. Intell.*, 2009: 421425:1–421425:19.

Wang, C.; Yu, Y.; Ma, W.; Zhang, M.; Chen, C.; Liu, Y.; and Ma, S. 2022a. Towards Representation Alignment and Uniformity in Collaborative Filtering. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, 1816–1825. ACM.

Wang, S.; Zhang, D.; Yan, Z.; Zhang, J.; and Li, R. 2023a. Feature Alignment and Uniformity for Test Time Adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, 20050–20060. IEEE.

Wang, T.; and Isola, P. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, 9929–9939. PMLR.

Wang, W.; Feng, F.; He, X.; Nie, L.; and Chua, T. 2021. Denoising Implicit Feedback for Recommendation. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, 373–381. ACM.

Wang, W.; Xu, Y.; Feng, F.; Lin, X.; He, X.; and Chua, T. 2023b. Diffusion Recommender Model. In Chen, H.; Duh, W. E.; Huang, H.; Kato, M. P.; Mothe, J.; and Poblete, B., eds., *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, 832–841. ACM.

Wang, X.; He, X.; Wang, M.; Feng, F.; and Chua, T. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, 165–174. ACM.

Wang, Y.; Xin, X.; Meng, Z.; Jose, J. M.; Feng, F.; and He, X. 2022b. Learning Robust Recommenders through Cross-Model Agreement. In Laforest, F.; Troncy, R.; Simperl, E.; Agarwal, D.; Gionis, A.; Herman, I.; and Médini, L., eds., *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, 2015–2025. ACM.

Wang, Z.; Zhao, H.; and Shi, C. 2022. Profiling the Design Space for Graph Neural Networks based Collaborative Filtering. In Candan, K. S.; Liu, H.; Akoglu, L.; Dong, X. L.; and Tang, J., eds., *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, 1109–1119. ACM.

Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; and Xie, X. 2021a. Self-supervised Graph Learning for Rec-

ommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, 726–735. ACM.

Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; and Xie, X. 2021b. Self-supervised Graph Learning for Recommendation. In Diaz, F.; Shah, C.; Suel, T.; Castells, P.; Jones, R.; and Sakai, T., eds., *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, 726–735. ACM.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2021c. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.*, 32(1): 4–24.

Xia, L.; Huang, C.; Huang, C.; Lin, K.; Yu, T.; and Kao, B. 2023. Automated Self-Supervised Learning for Recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, 992–1002. ACM.

Yang, L.; Liu, Z.; Wang, C.; Yang, M.; Liu, X.; Ma, J.; and Yu, P. S. 2023. Graph-based Alignment and Uniformity for Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, 4395–4399. ACM.

Yu, J.; Yin, H.; Xia, X.; Chen, T.; Cui, L.; and Nguyen, Q. V. H. 2022. Are Graph Augmentations Necessary?: Simple Graph Contrastive Learning for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, 1294–1303. ACM.

Zhang, Y.; Cui, G.; Deng, S.; Chen, F.; Wang, Y.; and He, Q. 2021a. Efficient Query of Quality Correlation for Service Composition. *IEEE Trans. Serv. Comput.*, 14(3): 695–709.

Zhang, Y.; Sang, L.; and Zhang, Y. 2024. Exploring the Individuality and Collectivity of Intents behind Interactions for Graph Collaborative Filtering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, 1253–1262. ACM.

Zhang, Y.; Wang, K.; He, Q.; Chen, F.; Deng, S.; Zheng, Z.; and Yang, Y. 2021b. Covering-Based Web Service Quality Prediction via Neighborhood-Aware Matrix Factorization. *IEEE Trans. Serv. Comput.*, 14(5): 1333–1344.

Zhang, Y.; Yin, C.; Wu, Q.; He, Q.; and Zhu, H. 2021c. Location-Aware Deep Collaborative Filtering for Service Recommendation. *IEEE Trans. Syst. Man Cybern. Syst.*, 51(6): 3796–3807.

Zhang, Y.; and Zhang, Y. 2025. MixRec: Individual and Collective Mixing Empowers Data Augmentation for Recommender Systems. In *Proceedings of the ACM on Web Conference 2025*, 2198–2208.

Zhang, Y.; Zhang, Y.; Yan, D.; Deng, S.; and Yang, Y. 2023. Revisiting Graph-based Recommender Systems from the Perspective of Variational Auto-Encoder. *ACM Trans. Inf. Syst.*, 41(3): 81:1–81:28.

Zhao, J.; Wang, W.; Xu, Y.; Sun, T.; Feng, F.; and Chua, T. 2024. Denoising Diffusion Recommender Model. In Yang, G. H.; Wang, H.; Han, S.; Hauff, C.; Zuccon, G.; and Zhang, Y., eds., *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, 1370–1379. ACM.