Sub-Interest-Aware Representation Uniformity for Recommender System

Ruijia Ma, Yahong Lian, Chunyao Song*

College of Computer Science, TJ Key Lab of NDST, DISSec, TMCC, TBI Center, Nankai University, Tianjin, China
{rigel.ma, yahong.lian}@mail.nankai.edu.cn, chunyao.song@nankai.edu.cn

Abstract

In today's information-rich era, users rely heavily on recommender systems to identify relevant content. Graph structures, renowned for their ability to model intricate user-content relationships, have become essential to these systems. However, the accuracy of recommendations hinges critically on the quality of node representations within these graphs. Personalized recommendations strive to enhance uniqueness by maximizing the dissimilarity between representations (known as **uniformity**) while simultaneously ensuring that the representations align closely with the content users engage with (dubbed as **alignment**). Nevertheless, balancing these conflicting objectives remains a challenge for optimal recommendation performance. To tackle these challenges, we propose an innovative approach called SIURec, which differs significantly from previous studies. Rather than relying on manual weight selection between uniformity and alignment and optimizing uniformity solely on the final representation, SIURec adopts an adaptive adjustment method that learns the optimal weight between uniformity and alignment automatically. By optimizing uniformity at every convolutional layer, SIURec captures users' sub-interests more effectively, ultimately leading to improved recommendation accuracy. Experimental results on four datasets demonstrate that SIURec achieves superior learning of uniformity (with an average improvement of 4.26% in accuracy compared to eleven SOTA methods) and exhibits robustness across different hyperparameter settings.

Code — https://github.com/xderui/SIURec

1 Introduction

Recommender systems, which mine users' latent preferences from their historical behaviors to more accurately push items of interest to them, have demonstrated considerable success across various domains like e-commerce, social network, and travel planning (Chen et al. 2023; Yu et al. 2023b), etc. Recently, personalized recommendation approaches leveraging Graph Neural Networks (Gao et al. 2023) have significantly enhanced recommendation efficacy by treating users and items as nodes within a graph, capturing intricate higher-order associations (He et al. 2020; Mao et al. 2021; Peng, Sugiyama, and Mine 2022). Among them, the quality of user/item representations is a key ingredient to improve recommendation accuracy. On one hand, to cater to users' personalized preferences, it's crucial to maintain dissimilarity among user representations, ensuring a spatial distribution with distinct embeddings for different users (known as **uniformity**). On the other hand, for accurate recommendations, user representations should closely resemble item representations (positive samples) they've interacted with previously, so as to achieve precise delivery of targeted content (known as **alignment**) (Zhuo et al. 2023; Zhang et al. 2023). **Optimizing both the conflicting goals of uniformity and alignment presents a formidable challenge** (Meng et al. 2021; Wang et al. 2022).

Studies have demonstrated that the incorporation of contrastive learning (CL) methods as well as Bayesian Personalized Ranking (BPR) both have the potential to optimize uniformity and alignment, thereby enhancing the effectiveness of personalized recommendations (Huang et al. 2023; Yang et al. 2023). Based on these findings, recent works (Wang and Isola 2020; Wang et al. 2022) have quantized the alignment and uniformity metrics, and achieved commendable results through manual tuning. This direct optimization of alignment and uniformity is called the AU-based approach.

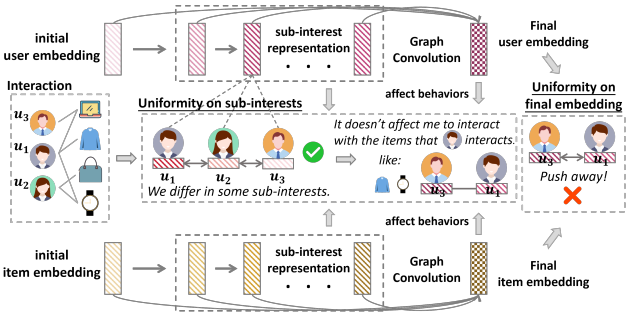


Figure 1: Optimizing uniformity on sub-interests can prevent incorrect uniformity on the final embedding.

However, existing AU-based methods optimize uniformity directly on the ultimate embedding, which may conflict with real-world scenarios in some cases. For instance, in Figure 1, solely optimizing uniformity on the final embedding may inadvertently push u_1 away from u_3, thus hin-

*Corresponding author

dering u_3 who shares interests with u_1 from developing an interest in the jacket and watch that u_1 finds appealing. This scenario deviates from reality. Different from this, we introduce a methodology that optimizes sub-interest uniformity at each convolutional layer, ensuring the final aggregated embedding aligns with real-world intricacies. Furthermore, current AU-based approaches require artificially setting the weights of the two objectives, alignment and uniformity. From the perspective of uniformity, a small weight leads to insufficient uniformity and reduced personalization, while an excessive weight, implying an overemphasis on uniformity, can compromise user interest modeling precision, negatively impacting overall recommendation performance.

Overall, existing AU-based approaches exhibit two significant shortcomings: (1) The direct optimization of the ultimate embedding's uniformity can create a conflict between its intended objective and reality, making it challenging to regulate the achieved uniformity. (2) The manual setting for the uniformity objective significantly affects the recommendation performance, rendering the existing methods less flexible and adaptable to varying scenarios.

In this paper, we propose a novel recommendation model towards **S**ub-**I**nterest-aware **U**niformity, called **SIURec**, to address the issue of inappropriate uniformity in recommender systems. Specifically, to refine the sub-interests of each node at a finer granularity, we generate a pseudo-interaction matrix with preference scores and integrate it with the original preference matrix. Then, based on different order of refined sub-interests, we employ uniformity optimization on the distribution of the refined sub-interests at the same level within each intermediate convolutional layer, instead of optimizing uniformity directly on the ultimate node representations which are composed of multiple sub-interests (in section 4.1). Furthermore, to fine-tune the uniformity, we design a mechanism that leverages gradients from diverse tasks to automatically set the weights, taking into account both the uniformity across different orders and the primary recommendation task. During the weight-solving process, we introduce a time-efficient approach called One-shot Weighting (in section 4.2), which obtains reasonably better weight settings through the optimization information of only one epoch, improving the training performance without additional time overheads.

In summary, our main contributions are as follows:

(1) We devise a novel technique that achieves **layer-wise uniformity optimization**, enabling the extraction of **fine-grained sub-interests** of users/items, and facilitating the learning of representations that better meet the need for uniformity, resulting in higher recommendation accuracy.

(2) We propose an **automated weight adaptation method** to optimize the balance between uniformity and alignment, which is efficient with one-shot solving, enhancing its flexibility and adaptability.

(3) Extensive experiments were carried out using four real-world datasets, comparing our approach to eleven state-of-the-art methods covering contrastive learning, denoising, and AU representation techniques. The results indicate that our method demonstrates significant improvement and is adaptable across various datasets.

2 Related Work

Graph-based recommendation methods have experienced significant growth because of their capacity to model intricate relationships between users and items (Gao et al. 2023; He et al. 2020). In these methods, the construction of high-quality representations for users and items is undeniably crucial for inducing superb recommendation results. To further enhance the accuracy of recommendations, researchers have explored various tactics, including denoising (Ren et al. 2023; Wang et al. 2023), self-supervised learning (SSL) (Yu et al. 2023b; Jing et al. 2023), and intention modeling (Liu et al. 2024). Among them, the denoise-driven methods claim that the observed data carries noise, which will undermine recommendation quality (Wang et al. 2023; Ren et al. 2023). Besides, SSL methods have been extensively utilized in recommendation systems. Among them, contrastive learning (CL)-based approaches have emerged as a mainstream branch (Ren et al. 2024). They construct two or more extra views of representations, whether by injecting noise (Yu et al. 2022) or auxiliary constraint term (Yu et al. 2023a), to obtain more robust user/item representations (Wu et al. 2021; Jiang, Huang, and Huang 2023). Interestingly, recent studies have indicated that the effectiveness of CL-based methods in recommendation systems is partially attributed to their ability to indirectly optimize the alignment and uniformity (AU) of representations (Wang and Isola 2020; Zhou et al. 2022). Instead of using an indirect approach, AU-based recommendation methods explicitly defined the AU loss and optimized it. DirectAU (Wang et al. 2022) was the first work that introduced the optimization objective of AU to the recommendation task. GraphAU (Yang et al. 2023) incorporated higher-order connectivity within user-item bipartite graphs by aligning embeddings with graph representations of higher-order neighborhoods. ProtoAU (Ou et al. 2024) optimized AU by post-node clustering to ensure coherence among different views. AUPlus (Ouyang et al. 2024) identified the label dependency of the user/item representations from the perspective of alignment and uniformity. However, previous AU-based studies are still limited in generalization ability and ignorance of layer-wise uniformity, both may lead to sub-optimal recommendation results.

3 Preliminaries

Problem Statement. Recommender systems aim to learn the preferences of users from their historical behaviors, then recommend the k most relevant items for a given user u. The historical behaviors are represented by an interaction matrix $\mathcal{A} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$, where $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ and $\mathcal{I} = \{i_1, i_2, \ldots, i_{|\mathcal{I}|}\}$ are the set of users and items, respectively. In matrix $\mathcal{A}$, $\mathcal{A}_{ui} = 1$ if user u has interacted with item i and $\mathcal{A}_{ui} = 0$ otherwise.

Graph Convolution and Aggregation. Our model is grounded on LightGCN (He et al. 2020), which simplifies GCN (Chen et al. 2020) by eschewing nonlinear transformations in favor of direct node aggregation. In terms of users, denote $E_l^{(u)} = \left(e_{l,1}^{(u)}, e_{l,2}^{(u)}, \ldots, e_{l,|\mathcal{U}|}^{(u)}\right)$, where $e_l^{(u)} \in \mathbb{R}^d$ represents the d-dimensional vector embedding of the l-th

layer pertaining to a specific user node. For items, $E_l^{(i)} \in \mathbb{R}^{|\mathcal{I}|\times d}$ is denoted analogously. To update each node, an aggregation of information from its neighboring nodes is employed, which is calculated as follows:

$$Z_l^{(u)} = \hat{\mathcal{A}} \cdot E_l^{(i)}, Z_l^{(i)} = \hat{\mathcal{A}}^T \cdot E_l^{(u)}. \tag{1}$$

In equation (1), $\hat{\mathcal{A}} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ indicates the normalized adjacent matrix by $\hat{\mathcal{A}} = D_{(u)}^{-1/2} \cdot \mathcal{A} \cdot D_{(i)}^{-1/2}$, where $D_{(u)} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ and $D_{(i)} \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$ are the diagonal degree matrices of users and items, respectively. In general, the ultimate node representations (i.e., the aggregation of all layer of embeddings) are viewed as the interest of nodes (Tan et al. 2021; Cen et al. 2020). Thus we first designate the layer embedding $Z_l^{(*)}$ (where superscript $(*)$ means either user aspect u or item aspect i, the same is true for the follow-up.) as the l-th order **coarse-grained sub-interest** of nodes. Then the representation of users/items at each layer $E_{l+1}^{(*)}$ can be written as:

$$E_{l+1}^{(*)} = Z_l^{(*)}. \tag{2}$$

The ultimate node representations $E^{(*)}$ are derived from the aggregation across each layer of embeddings, as follows:

$$E^{(*)} = \sum_{l=0}^{L} E_l^{(*)}, \tag{3}$$

where L is the number of layers. And we get both user and item ultimate representation $E^{(u)}, E^{(i)}$ respectively.

Alignment and Uniformity Loss. They have been shown to improve representation quality as an optimization objective on hyperspheres (Wang and Isola 2020; Gao, Yao, and Chen 2021; Wang et al. 2022). Given the user and item representation $e_u \in E^{(u)}, e_i \in E^{(i)}$, the alignment loss is calculated as follows:

$$\mathcal{L}_A = \frac{1}{|\mathcal{P}|} \sum_{(u,i)\in\mathcal{P}} \mathcal{L}_A^{(u,i)} = \frac{1}{|\mathcal{P}|} \sum_{(u,i)\in\mathcal{P}} \|e_u - e_i\|^2, \tag{4}$$

where $\mathcal{L}_A^{(u,i)}$ is the alignment loss between user u and item i, $\mathcal{P}$ is the positive samples of user-item pairs. The objective of alignment is to maximize the similarity of representation between the user and the positive samples.

On the other hand, The uniformity loss is calculated by:

$$\mathcal{L}_U = \frac{1}{2}(\mathcal{L}_U^{(u)} + \mathcal{L}_U^{(i)}), \tag{5}$$

$$\mathcal{L}_U^{(u)} = log \frac{1}{|\mathcal{U}|^2} \sum_{u\in\mathcal{U}} \sum_{u^*\in\mathcal{U}/\{u\}} e^{-2\|e_u - e_{u^*}\|}, \tag{6}$$

$$\mathcal{L}_U^{(i)} = log \frac{1}{|\mathcal{I}|^2} \sum_{i\in\mathcal{I}} \sum_{i^*\in\mathcal{I}/\{i\}} e^{-2\|e_i - e_{i^*}\|}. \tag{7}$$

Uniformity aims to increase the distance of representation among users/items while ensuring their uniform distribution.

4 Methodology

The framework of our proposed SIURec is shown in Figure 2, which consists of sub-interest-aware uniformity, including fine-grained sub-interests extraction and layer-wise uniformity optimization, as well as one-shot weighting.

4.1 Sub-Interest-aware Uniformity

The original interaction matrix $\mathcal{A}$ comprises binary entries. However, this fails to discern the underlying reasons that drive users' preferences for various items. To capture diverse user preferences for items and more **fine-grained sub-interests**, we use the similarity between the coarse-grained sub-interests of users and items as the preferred score to generate a new adjacency matrix similar to (Ren et al. 2023). This procedure is calculated as follows:

$$\widetilde{\mathcal{A}}_l = \frac{Z_l^{(u)} \cdot Z_l^{(i)}}{\|Z_l^{(u)}\| \|Z_l^{(i)}\|}, \tag{8}$$

$$\ddot{\mathcal{A}}_l^{(u)} = \widetilde{\mathcal{A}}_l / \|\widetilde{\mathcal{A}}_l\|, \ddot{\mathcal{A}}_l^{(i)} = \widetilde{\mathcal{A}}_l^T / \|\widetilde{\mathcal{A}}_l^T\|. \tag{9}$$

As the layer of embeddings change during the aggregation process, a new adjacency matrix is derived. To mitigate the influence of node popularity and considering the distinct preference of various perspectives, we employ normalization on rows and columns to represent the users' preference and items' preference respectively, i.e., $\ddot{\mathcal{A}}_l^{(u)} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ and $\ddot{\mathcal{A}}_l^{(i)} \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{U}|}$.

Subsequently, akin to Equation (1), we employ the updated adjacency matrix to aggregate node information, yielding the **fine-grained sub-interests** of nodes $\widetilde{Z}_l^{(*)}$ as follows:

$$\widetilde{Z}_l^{(u)} = \ddot{\mathcal{A}}_l^{(u)} \cdot \widetilde{E}_l^{(i)}, \widetilde{Z}_l^{(i)} = \ddot{\mathcal{A}}_l^{(i)} \cdot \widetilde{E}_l^{(u)}, \widetilde{E}_0^{(*)} = E_0^{(*)} \tag{10}$$

Then we replace the original layer representation in Equation (2) into the new manner to obtain the refined representation $\widetilde{E}_{l+1}^{(*)}$ for each layer as follows:

$$\widetilde{E}_{l+1}^{(*)} = Z_l^{(*)} + \widetilde{Z}_l^{(*)}. \tag{11}$$

Given that similarity scores may not always be accurate, relying solely on $\widetilde{Z}_l^{(*)}$ while neglecting $Z_l^{(*)}$ is not feasible. Hence, we incorporate $\widetilde{Z}_l^{(*)}$ as supplementary information as shown in equation (11). Then, same as pooling Equation (3), we get the refined ultimate node representation $\widehat{E}^{(*)} = \sum_{l=0}^{L} \widetilde{E}_l^{(*)}$. It's important to highlight that we refrained from employing contrastive learning between $Z_l^{(*)}$ and $\widetilde{Z}_l^{(*)}$, as we don't anticipate them to exhibit consistency. This decision lays the foundation for optimizing the uniformity of $\widetilde{Z}_l^{(*)}$ in subsequent stages.

To optimize uniformity, most existing methods conduct on the ultimate node representation $E^{(*)}$ in Equation (3). Nevertheless, it may hinder the model's performance since it is extremely sensitive to the parameters of the uniformity objective function (Wang et al. 2022). In our methodology, we consider the aggregation of neighboring nodes for a given node as an indicator of its sub-interests, denoted by Z (coarse-grained) and $\widetilde{Z}$ (fine-grained). We argue that $\widetilde{Z}$ can express more nuanced sub-interests compared to Z, thus we optimize the uniformity of $\widetilde{Z}$ to disentangle more sophisticated differences among different nodes on each layer. Tak-

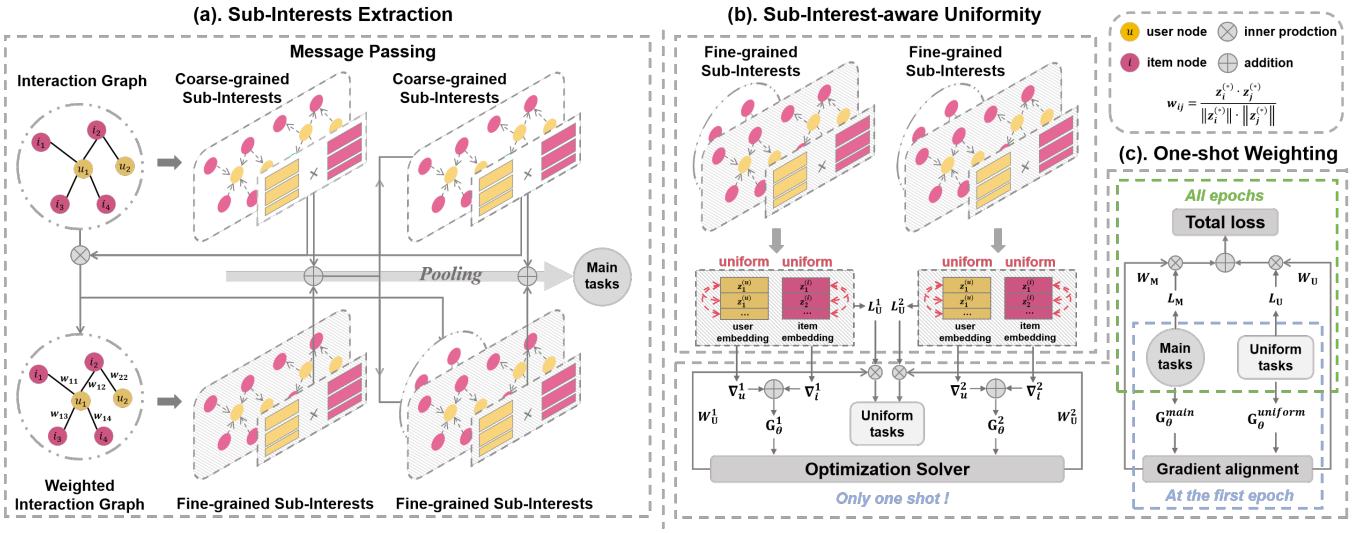


Figure 2: The framework of our proposed SIURec. (a) Basic training model. (b) Optimize the uniformity for each fine-grained sub-interest. (c) Calculate the weights of uniformity and BPR tasks separately to obtain better learning.

ing user embedding representation as an example:

$$\begin{aligned}\mathcal{L}_U^{(u)} &= \sum_{l=1}^{L} log \frac{1}{|\mathcal{U}|^2} \sum_{u\in\mathcal{U}} \sum_{u^*\in\mathcal{U}/\{u\}} e^{-2\|\widetilde{z}_{l,u}-\widetilde{z}_{l,u^*}\|} \\ &= log \prod_{l=1}^{L} \frac{1}{|\mathcal{U}|^2} \sum_{u\in\mathcal{U}} \sum_{u^*\in\mathcal{U}/\{u\}} e^{-2\|\widetilde{z}_{l,u}-\widetilde{z}_{l,u^*}\|}\end{aligned} \tag{12}$$

where $\widetilde{z} \in \mathbb{R}^d$ is a vector in $\widetilde{Z}$, i.e., $\widetilde{Z}_l^{(u)} = \left(\widetilde{z}_{l,1}^{(u)}, \widetilde{z}_{l,2}^{(u)}, \ldots, \widetilde{z}_{l,|\mathcal{U}|}^{(u)}\right)$. For items, $\mathcal{L}_U^{(i)}$ is calculated analogously as equation (12). As per Equation (11), we prioritize uniformity solely in $\widetilde{Z}$, while intentionally overlooking it in Z, to enhance the flexibility of the final $\widehat{E}^{(*)}$ and ensure it aligns more closely with real-world scenarios, as illustrated in Figure 1. Typically, owing to the uniformity of the final representation $\widehat{E}^{(*)}$, AU-based methods utilize alignment loss directly for learning positive samples, thereby eliminating the need for negative sampling. However, alignment using solely positive samples is not feasible here due to the lack of uniformity assurance in $\widehat{E}^{(*)}$, thus we utilize BPR instead of alignment loss as follows:

$$\mathcal{L}_B = \frac{1}{|\mathcal{B}|} \sum_{(u,i,j)\in\mathcal{B}} \sigma(\widehat{E}_u^{(u)}(\widehat{E}_i^{(i)})' - \widehat{E}_u^{(u)}(\widehat{E}_j^{(i)})'), \tag{13}$$

where $\sigma(x) = \log(1+e^x)$, and $\mathcal{B}$ denotes the sampled interactions in each mini-batch, while (u,i,j) are the user u, positive item i, and negative item j, respectively. In the end, the total loss function can be expressed as follows:

$$\mathcal{L} = \alpha_B\mathcal{L}_B + \alpha_U\left(\mathcal{L}_U^{(i)} + \mathcal{L}_U^{(u)}\right) + \alpha_R\mathcal{L}_R, \tag{14}$$

where α_B, α_U, and α_R denote the weight of BPR loss, uniformity loss, and regularization term respectively. Here, $\mathcal{L}_R$ serves as a regularization constraint applied to the initial $\widetilde{E}_0^{(*)}$, specifically defined as $\mathcal{L}_R = \|\widetilde{E}_0^{(u)}\|^2 + \|\widetilde{E}_0^{(i)}\|^2$.

4.2 One-shot Weighting

Typically, GCN consists of at least two layers ($L \geq 2$), leading to multiple uniformity losses that correspond to each of its layers. Owing to the different uniform levels of different fine-grained sub-interests, it is essential to deliberate on establishing the weight relationship among these uniformity losses to deal with the relationship between different orders. In equation (12), additionally, a tunable weight is introduced for the loss of each layer:

$$\mathcal{L}_U = \sum_{l=1}^{L} w_l \left(\mathcal{L}_U^{(u)}(l) + \mathcal{L}_U^{(i)}(l)\right), \tag{15}$$

where $\mathcal{L}_U^{(*)}(l)$ represents the uniformity loss of fine-grained sub-interests $\widetilde{Z}_l^{(*)}$, and w_l denotes the weight associated with its loss. In multi-objective optimization, there exists extensive research on weight assignment, such as Pareto Optimization, which is proven effective in multi-objective optimization tasks, enabling the objectives to be optimized with as little detriment to other objectives as possible (Lin et al. 2019; Xie et al. 2021). The optimal solution is referred to as the Pareto efficient case, and all Pareto efficient cases collectively form the Pareto frontier. In situations where Pareto efficiency is applicable, it must exhibit Pareto stationarity (Sener and Koltun 2018), which can be quantified by a weighted sum of loss gradients, as illustrated below:

$$\begin{aligned} min. &\left\|\sum_{l=1}^{L} w_l \nabla_\theta \left(\mathcal{L}_U^{(u)}(l) + \mathcal{L}_U^{(i)}(l)\right)\right\|_2^2 \\ s.t. &\sum_{l=1}^{L} w_l = 1, \forall w_l \geq 0. \end{aligned} \tag{16}$$

where $\nabla_\theta(\cdot)$ indicates the gradient of the corresponding function. The problem (16) was verified to satisfy the

Karush-Kuhn-Tucker (KKT) conditions (Kjeldsen 2000) and its solution provided directions that can be optimized for multiple tasks concurrently (Désidéri 2012). After solving the problem (16), $w = \{w_1, w_2..w_l...w_L\}$ are obtained and the sum of them is 1. To prioritize the most significant layer objective, we assign a weight of 1 to the layer with the largest weight and determine the practical weights for the remaining layers proportionally, as outlined below:

$$\omega_l = \begin{cases} 1, & l = \text{argmax}\,(w); \\ \frac{w_l}{\max(w)}, & others. \end{cases} \tag{17}$$

Finally, the overall loss function in Equation (14) is rewritten as $\mathcal{L} = \alpha_B \mathcal{L}_B + \alpha_U \sum_{l=1}^{L} \omega_l \left(\mathcal{L}_U^{(u)}(l) + \mathcal{L}_U^{(i)}(l) \right) + \alpha_R \mathcal{L}_R$.

Generally, during the optimization process, BPR loss and uniformity may conflict with each other, thus optimizing one objective may affect the other (Wang and Isola 2020; Wang et al. 2022). Multi-objective optimization like Pareto optimization remains an alternative that maximizes the joint benefit of multiple objectives. However, better overall optimization of the targets does not necessarily lead to better final results for the two special objective tasks, BPR loss, and uniformity. Uniformity targets to achieve a uniform distribution of representations to maximize the preservation of personalized information, which requires optimization across a batch of samples. Meanwhile, the goal of BPR loss is to minimize the distances between users and positive samples locally, potentially sacrificing uniformity in the process. Through Pareto optimization, BPR loss often receives a heavier emphasis than uniformity due to its easier optimization, leading to a reduced focus and more challenging optimization for uniformity, as depicted in Figure 3 (a).

Therefore, different from previous work, to obtain appropriate α_U and greater preservation of personalized information, we design a method utilizing the ratio of the gradient produced in the optimization of uniformity and BPR:

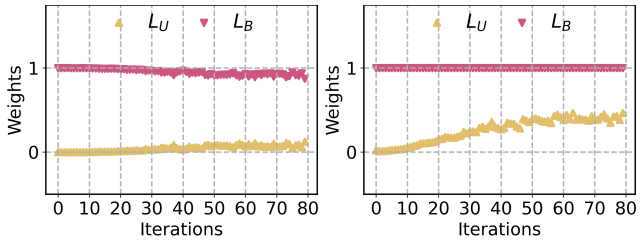
$$\alpha_U = \frac{\left\| \nabla_\theta \widehat{\mathcal{L}}_B \right\|_1}{\left\| \nabla_\theta \left(\widehat{\mathcal{L}}_U^{(u)} + \widehat{\mathcal{L}}_U^{(i)} \right) \right\|_1} \cdot \alpha_B. \tag{18}$$

Given the substantial difference in loss magnitudes across diverse objectives, a bias is inadvertently introduced into the gradient. To counteract this influence, we adopt a specific procedure for deriving gradients, as follows:

$$\widehat{\mathcal{L}}_B = \frac{\mathcal{L}_B - lower(\mathcal{L}_B)}{upper(\mathcal{L}_B) - lower(\mathcal{L}_B)}, \tag{19}$$

$$\widehat{\mathcal{L}}_U^{(*)} = \frac{\mathcal{L}_U^{(*)} - lower(\mathcal{L}_U^{(*)})}{upper(\mathcal{L}_U^{(*)}) - lower(\mathcal{L}_U^{(*)})}, \tag{20}$$

where $upper(\cdot)$ and $lower(\cdot)$ mean the upper and lower bound of the corresponding loss function. For example, the upper and lower bound of BPR loss is $\sigma(2) \approx 2.13$ and $\sigma(-2) \approx 0.13$, respectively, according to equation (13), and the upper and lower bound of uniformity loss is 0 and -8 when the similarity between any two vectors is 1 and -1, respectively, in equation (12). After the normalization, the variation in magnitude will be eliminated. Vary from the combinatorial optimization problem which might expand the disparity among the targets to achieve the optimal overall status, Equation (18) aligns for the gradients, resulting in their consistent effect on the parameters of models thus optimizing both of the two targets better. The weight allocation results calculated by gradient ratio are shown in Figure 3 (b).



(a) Pareto optimization (b) gradient alignment

Figure 3: Two methods for weight allocation between BPR loss and uniformity on Gowalla. For subfigure (b), the weight of $\mathcal{L}_B$(i.e., α_B) is set to 1.

Generally, optimal weights fluctuate with gradient changes, requiring iteration-wise recalibration. However, our goal is to balance alignment with maintaining uniformity for personalized interests, and excessive optimization risks diluting user personalization. As the initial state is unbiased by optimization objectives, weights solved in the first epoch best capture the relationship among objectives. Thus, we calculate weights only once in the first epoch, averaging iterations to determine an accurate uniformity degree. This average weight is then fixed for subsequent processes. For detailed evaluations and analyses, refer to Section 5.4.

5 Experiments

In this section, we evaluate SIURec on different datasets to answer following questions: **RQ1**: How does SIURec perform compared to other competitive methods under different experimental settings? **RQ2**: How does each component of SIURec contribute to performance enhancement? **RQ3**: How robust is SIURec under various parameter settings?

5.1 Experimental Setup

Datasets. We select four commonly used public benchmark datasets in our experiments: MovieLens-1M (ML-1M), Gowalla, Amazon-Beauty (Beauty) and Amazon-Book (Book). The dataset statistics are shown in Table 1. For each dataset, we group them by user and divide them into 8:1:1 ratios for training, validation, and testing.

Evaluation Metrics. Two prevalent metrics are employed to evaluate the performance of all methods, including Recall@K and NDCG@K with K=20/40 (Yang et al. 2023).

Baselines. Based on the relevance to our work, we selected representative SOTA methods from several categories that currently perform best in the recommendation field as comparison models. *(i).Graph-based Recommendation Method:* **LightGCN** (He et al. 2020); *(ii).Contrastive Learning-based Recommendation Methods:* **SGL** (Wu et al.

Datasets	#Users	#Items	#Interactions	Density
ML-1M	6,040	3,416	902,299	3.902%
Gowalla	50,821	57,440	1,196,581	0.037%
Beauty	22,363	12,101	192,244	0.056%
Book	78,578	77,801	2,880,201	0.037%

Table 1: statistics of the datasets

2021), **SimGCL** (Yu et al. 2022), **XSimGCL** (Yu et al. 2023a), and **AdaGCL** (Jiang, Huang, and Huang 2023); *(iii).Denoising-based Recommendation Methods:* **DCCF** (Ren et al. 2023), and **BOD** (Wang et al. 2023); *(iv).AU-based Recommendation Methods:* **DirectAU** (Wang et al. 2022), **GraphAU** (Yang et al. 2023), **ProtoAU** (Ou et al. 2024), and **AUPlus** (Ouyang et al. 2024).

Implementation Details. All experiments are implemented on an Intel(R) Xeon(R) Silver 4110 @ 2.10GHz CPU and an NVIDIA GeForce RTX 2080 Ti GPU. For each baseline, we set the parameters following the suggestions from each individual's work. Regard to SIURec, we set the initial values $\alpha_B = 1$, $\alpha_U = 1$, and $\alpha_R = 2.5 \times 10^{-5}$.

5.2 Performance Comparision (RQ1)

The results are presented in Table 2. According to the table, we can derive the following observations and conclusions:

(1) Our method consistently outperforms all alternatives, uniquely optimizing the uniformity of fine-grained sub-interest representations, rather than the ultimate node representations as other AU-based methods do. In practice, sub-interests consistently exhibit greater uniformity, leading to more effective representation and thus superior performance.

(2) Excluding the Beauty dataset, AU-based models lag behind other types of methods, suggesting direct optimization of ultimate node representations doesn't always enhance recommendation quality. This may stem from unrealistic uniformity in these representations, impacting alignment quality. Conversely, our SIURec, also optimizing uniformity, consistently outperforms across datasets, demonstrating its superior adaptability.

(3) In certain scenarios, CL-based methods significantly outperform other AU-based methods, suggesting that contrastive learning may sometimes adjust the degree of uniformity more effectively, despite being an implicit optimization of alignment and uniformity rather than an explicit one.

5.3 Ablation Study (RQ2)

In this section, we evaluate the effectiveness of each module we designed, including **f**ine-grained **s**ub-**i**nterest extraction (**FSI**), with the alternative of current **c**oarse-grained **s**ub-**i**nterest extraction (**CSI**); **l**ayer-wise **u**niformity optimization (**LU**), with the alternative of current **u**ltimate **u**niformity optimization (**UU**); and **o**ne-shot **w**eighting (**OW**), with the alternative of **c**ontinuous **w**eighting (**CW**) that learns the weights in every training epoch. As shown in Table 3, variant module designs based on LightGCN are added from left to right, with the complete model's results on the far right.

We can see from Table 3 that only add FSI without uniformity optimization offers performance enhancements over LightGCN as reported in Table 2. In addition, we replace each of the three modules in the complete model to obtain three other variants. As for the variants where the FSI module is replaced with the CSI module and the LU module is replaced with the UU module, we can observe a relatively significant decline in performance. Some results are even lower than those achieved by adding the FSI module alone, indicating that the coupling and correlation among the three modules are also crucial. Finally, the comparable outcomes between +CW and SIURec (which is +OW) indicate that solving the weights per epoch might be redundant. While it occasionally offers improvements, its inclusion leads to unnecessary increased computational time.

Setting		Baselines											Ours
Dataset	Metric	LightGCN	SGL	SimGCL	XSimGCL	AdaGCL	DCCF	BOD	DirectAU	GraphAU	ProtoAU	AUPlus	SIURec
ML-1M	R@20	0.1013	0.0947	0.1056	0.0869	0.1035	0.0917	0.0963	0.0851	0.0846	0.1003	0.0896	**0.1099**
	R@40	0.1681	0.1621	0.1814	0.1476	0.1772	0.1544	0.1660	0.1410	0.1417	0.1674	0.1527	**0.1859**
	N@20	0.0826	0.0782	0.0806	0.0658	0.0811	0.0667	0.0798	0.0742	0.0755	0.0811	0.0676	**0.0877**
	N@40	0.1029	0.0987	0.1036	0.0847	0.1037	0.0864	0.1010	0.0910	0.0928	0.1016	0.0871	**0.1108**
Gowalla	R@20	0.1571	0.1787	0.1930	0.1956	0.1072	0.1713	0.1913	0.1817	0.1821	0.1678	0.1691	**0.2048**
	R@40	0.2248	0.2520	0.2673	0.2708	0.1839	0.2413	0.2656	0.2529	0.2563	0.2394	0.2406	**0.2843**
	N@20	0.0922	0.1060	0.1156	0.1165	0.0825	0.1021	0.1143	0.1070	0.1070	0.0985	0.0991	**0.1224**
	N@40	0.1098	0.1253	0.1351	0.1363	0.1063	0.1203	0.1337	0.1257	0.1263	0.1172	0.1177	**0.1432**
Beauty	R@20	0.1158	0.1166	0.1293	0.1116	0.1268	0.1027	0.1348	0.1331	0.1296	0.1193	0.1329	**0.1374**
	R@40	0.1603	0.1604	0.1772	0.1485	0.1748	0.1390	0.1825	0.1804	0.1713	0.1648	0.1780	**0.1896**
	N@20	0.0636	0.0653	0.0730	0.0638	0.0690	0.0575	0.0752	0.0752	0.0758	0.0651	0.0752	**0.0759**
	N@40	0.0749	0.0766	0.0854	0.0733	0.0812	0.0669	0.0875	0.0876	0.0881	0.0766	0.0875	**0.0892**
Book	R@20	0.0506	0.0751	0.0892	0.0893	0.0477	0.0717	0.0847	0.0836	0.0849	0.0705	0.0740	**0.0944**
	R@40	0.0813	0.1147	0.1344	0.1333	0.0737	0.1091	0.1266	0.1246	0.1266	0.1097	0.1121	**0.1408**
	N@20	0.0374	0.0566	0.0682	0.0676	0.0366	0.0545	0.0647	0.0634	0.0648	0.0525	0.0568	**0.0722**
	N@40	0.0476	0.0697	0.0830	0.0822	0.0451	0.0669	0.0785	0.0768	0.0785	0.0655	0.0693	**0.0874**

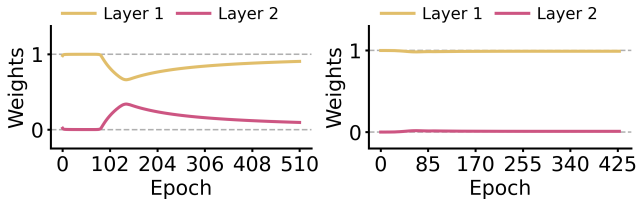
Table 2: Performance comparison on different datasets. The highest scores are in bold, and the second best are with underlines.

Datasets	Metrics	Variants				FSI+LU +OW (SIURec)
		FSI	CSI+LU +OW	FSI+UU +OW	FSI+LU +CW	
ML-1M	R@40	0.1796	0.1785	0.1741	**0.1885**	0.1859
	N@40	0.1079	0.1079	0.1052	0.1093	**0.1108**
Beauty	R@40	0.1781	0.1724	0.1651	0.1787	**0.1896**
	N@40	0.0828	0.0785	0.0833	0.0888	**0.0892**

Table 3: Ablation study with the highest scores highlighted in bold, and the second best indicated with underlines.

5.4 In-Depth Analysis of SIURec (RQ3)

In section 4.2, it has been highlighted that the solution of weights only need to be calculated once. Here, we plot the variation of weights when solving the best weights at each epoch in the situation of two GCN layers, as shown in Figure 4. From this figure, we find that the weight of the first layer is larger than that of the second layer in all situations. This suggests that first-order sub-interests are more individualized compared to those of higher-order. Most importantly, the weight solving results of the first epoch can almost represent the weight allocation throughout the entire process, demonstrating that there is no need to spend a significant amount of time solving for weights in each epoch to achieve minimal or even negative performance gains.



(a) ML-1M (b) Beauty

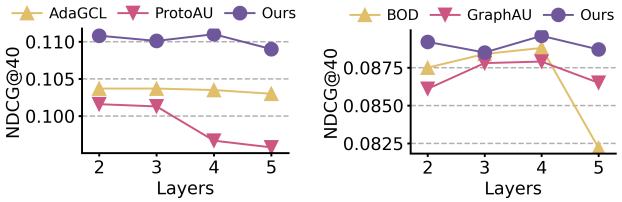
Figure 4: The process of weights variation

Dataset	ML-1M		Beauty		Book	
Setting	R@40	N@40	R@40	N@40	R@40	N@40
AO	0.1846	0.1091	0.1824	0.0884	0.1365	0.0842
OW	**0.1859**	**0.1108**	**0.1896**	**0.0892**	**0.1408**	**0.0874**

Table 4: Effect of sub-interest layer weight allocation on performance. In AO, all layers are equally weighted at 1.

In addition, to validate the significance of **O**ne-shot **W**eighting (OW) among sub-interests in different orders, i.e., the different uniformity of variant sub-interests, the results for **A**ll uniformity targets of fine-grained sub-interests weighted **O**ne (denoted as "AO") are displayed in Table 4. The results show that it is necessary to adapt the weights of different layers and our adjustment is effective.

Furthermore, we set the number of graph neural layers from 2 to 5, thereby analyzing the impact on our model. The results are shown in Figure 5. Compared to the two best models besides ours on respective datasets, our model maintains the best performance in all cases, indicating that our



(a) ML-1M (b) Beauty

Figure 5: Performance *w.r.t* the number of layers

model can tackle the over-smoothing problem as the number of layers increases, thanks to our uniformity optimization. In Figure 6, we present the weight allocation across layers for different total layer counts, demonstrating the consistent dominance of the first layer's sub-interest uniformity.

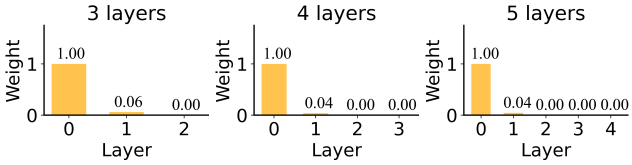
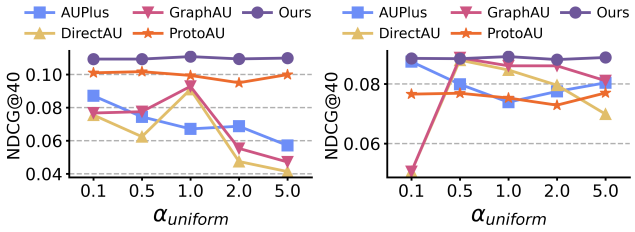


Figure 6: Weight allocation of different sub-interests *w.r.t* the number of layers on Beauty dataset

To verify that SIURec is insensitive to the initial setup parameters, we compare the performance of SIURec with AU-based methods concerning the weight of the uniformity target, as shown in Figure 7. The tendency of SIURec is close to a straight line, demonstrating the robustness of SIURec.



(a) ML-1M (b) Beauty

Figure 7: Performance *w.r.t* the initial weight of uniformity

6 Conclusion

This paper proposes a model named SIURec, which can extract the fine-grained sub-interests of nodes and then learn more appropriate uniformity representations layer-by-layer based on the sub-interest-aware uniformity. In order to control the appropriate degree of uniformity, we design two weighting strategies for different targets, which are Pareto optimization for the uniformity among all fine-grained sub-interests, and gradient alignment for the weights of BPR loss and uniformity. The experiments validate the effectiveness of our proposed SIURec on various scenarios and show the robustness on uniformity representation learning.

Acknowledgments

The authors express their gratitude to the anonymous reviewers for their thorough reviews and valuable feedback. This work was supported in part by National Science Foundation of China (62172237).

References

Cen, Y.; Zhang, J.; Zou, X.; Zhou, C.; Yang, H.; and Tang, J. 2020. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2942–2951.

Chen, J.; Dong, H.; Wang, X.; Feng, F.; Wang, M.; and He, X. 2023. Bias and Debias in Recommender System: A Survey and Future Directions. *ACM Trans. Inf. Syst.*, 41(3).

Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; and Li, Y. 2020. Simple and deep graph convolutional networks. In *International conference on machine learning*, 1725–1735. PMLR.

Désidéri, J.-A. 2012. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6): 313–318.

Gao, C.; Zheng, Y.; Li, N.; et al. 2023. A Survey of Graph Neural Networks for Recommender Systems: Challenges,Methods, and Directions. *Trans. Recomm. Syst.*, 1(1): 1–51.

Gao, T.; Yao, X.; and Chen, D. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

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*, 639–648.

Huang, Z.; Chen, H.; Wen, Z.; Zhang, C.; Li, H.; Wang, B.; and Chen, C. 2023. Model-aware contrastive learning: Towards escaping the dilemmas. In *International Conference on Machine Learning*, 13774–13790. PMLR.

Jiang, Y.; Huang, C.; and Huang, L. 2023. Adaptive graph contrastive learning for recommendation. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, 4252–4261.

Jing, M.; Zhu, Y.; Zang, T.; and Wang, K. 2023. Contrastive self-supervised learning in recommender systems: A survey. *ACM Transactions on Information Systems*, 42(2): 1–39.

Kjeldsen, T. H. 2000. A contextualized historical analysis of the Kuhn–Tucker theorem in nonlinear programming: the impact of World War II. *Historia mathematica*, 27(4): 331–361.

Lin, X.; Chen, H.; Pei, C.; Sun, F.; Xiao, X.; Sun, H.; Zhang, Y.; Ou, W.; and Jiang, P. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, 20–28. New York, NY, USA: Association for Computing Machinery. ISBN 9781450362436.

Liu, H.; Zhou, M.; Mingyang, S.; et al. 2024. Learning Hierarchical Preferences for Recommendation with Mixture Intention Neural Stochastic Processes. *IEEE TKDE*.

Mao, K.; Zhu, J.; Xiao, X.; Lu, B.; Wang, Z.; and He, X. 2021. UltraGCN: ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management*, 1253–1262.

Meng, Y.; Xiong, C.; Bajaj, P.; Bennett, P.; Han, J.; Song, X.; et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems*, 34: 23102–23114.

Ou, Y.; Chen, L.; Pan, F.; and Wu, Y. 2024. Prototypical Contrastive Learning through Alignment and Uniformity for Recommendation. *arXiv preprint arXiv:2402.02079*.

Ouyang, Z.; Zhang, C.; Hou, S.; Zhang, C.; and Ye, Y. 2024. How to Improve Representation Alignment and Uniformity in Graph-based Collaborative Filtering? In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 18, 1148–1159.

Peng, S.; Sugiyama, K.; and Mine, T. 2022. SVD-GCN: A simplified graph convolution paradigm for recommendation. In *Proceedings of the 31st ACM international conference on information & knowledge management*, 1625–1634.

Ren, X.; Wei, W.; Xia, L.; and Huang, C. 2024. A Comprehensive Survey on Self-Supervised Learning for Recommendation. *arXiv preprint arXiv:2404.03354*.

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*, 1137–1146. New York, NY, USA: Association for Computing Machinery. ISBN 9781450394086.

Sener, O.; and Koltun, V. 2018. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31.

Tan, Q.; Zhang, J.; Yao, J.; Liu, N.; Zhou, J.; Yang, H.; and Hu, X. 2021. Sparse-Interest Network for Sequential Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, 598–606. New York, NY, USA: Association for Computing Machinery. ISBN 9781450382977.

Wang, C.; Yu, Y.; Ma, W.; Zhang, M.; Chen, C.; Liu, Y.; and Ma, S. 2022. Towards representation alignment and uniformity in collaborative filtering. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1816–1825.

Wang, T.; and Isola, P. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, 9929–9939. PMLR.

Wang, Z.; Gao, M.; Li, W.; Yu, J.; Guo, L.; and Yin, H. 2023. Efficient bi-level optimization for recommendation denoising. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2502–2511.

Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; and Xie, X. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 726–735.

Xie, R.; Liu, Y.; Zhang, S.; Wang, R.; Xia, F.; and Lin, L. 2021. Personalized Approximate Pareto-Efficient Recommendation. In *Proceedings of the Web Conference 2021*, WWW '21, 3839–3849. New York, NY, USA: Association for Computing Machinery. ISBN 9781450383127.

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*, 4395–4399.

Yu, J.; Xia, X.; Chen, T.; Cui, L.; Hung, N. Q. V.; and Yin, H. 2023a. XSimGCL: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

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 *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, 1294–1303.

Yu, J.; Yin, H.; Xia, X.; Chen, T.; Li, J.; and Huang, Z. 2023b. Self-supervised learning for recommender systems: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, D.; Li, C.; Li, H.; Huang, W.; Huang, L.; and Zhang, J. 2023. Rethinking alignment and uniformity in unsupervised image semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 11192–11200.

Zhou, K.; Zhang, B.; Zhao, X.; and Wen, J.-R. 2022. Debiased Contrastive Learning of Unsupervised Sentence Representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 6120–6130. Dublin, Ireland.

Zhuo, W.; Sun, Y.; Wang, X.; Zhu, L.; and Yang, Y. 2023. WhitenedCSE: Whitening-based Contrastive Learning of Sentence Embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 12135–12148. Toronto, Canada.